Received April 10, 2021, accepted April 20, 2021, date of publication May 3, 2021, date of current version May 13, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3077117

Dynamics Learning With Object-Centric Interaction Networks for Robot Manipulation

JIAYU WANG, CHUXIONG HU, (Senior Member, IEEE), YUNAN WANG, AND YU ZHU, (Member, IEEE)
State Key Laboratory of Tribology, Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China
Beijing Key Laboratory of Precision/Ultra-Precision Manufacturing Equipments and Control, Tsinghua University, Beijing 100084, China

Corresponding author: Chuxiong Hu (cxhu@tsinghua.edu.cn)

This work was supported in part by the National Nature Science Foundation of China under Grant 51922059 and Grant 51775305, in part by the Beijing Natural Science Foundation under Grant JQ19010, in part by the State Key Laboratory of Tribology under Grant SKLT2020D22, in part by the National Youth Talent Support Program, and in part by the State Key Laboratory of Mechanical System and Vibration under Grant MSV202007.

ABSTRACT Understanding the physical interactions of objects with environments is critical for multi-object robotic manipulation tasks. A predictive dynamics model can predict the future states of manipulated objects, which is used to plan plausible actions that enable the objects to achieve desired goal states. However, most current approaches on dynamics learning from high-dimensional visual observations have limitations. These methods either rely on a large amount of real-world data or build a model with a fixed number of objects, which makes them difficult to generalize to unseen objects. This paper proposes a Deep Object-centric Interaction Network (DOIN) which encodes object-centric representations for multiple objects from raw RGB images and reasons about the future trajectory for each object in latent space. The proposed model is trained only on large amounts of random interaction data collected in simulation. The learned model combined with a model predictive control framework enables a robot to search action sequences that manipulate objects to the desired configurations. The proposed method is evaluated both in simulation and real-world experiments on multi-object pushing tasks. Extensive simulation experiments show that DOIN can achieve high prediction accuracy in different scenes with different numbers of objects and outperform state-of-the-art baselines in the manipulation tasks. Real-world experiments demonstrate that the model trained on simulated data can be transferred to the real robot and can successfully perform multi-object pushing tasks for previously-unseen objects with significant variations in shape and size.

INDEX TERMS Deep learning in robotic manipulation, model learning, representation learning, visual learning.

I. INTRODUCTION

Humans possess a natural physical intuition that can predict the effects of actions and the state of things in the future. The ability to interpret the physical world can be acquired and improved through experience. Understanding the efforts of physical interactions is essential for planning actions in robotic manipulation tasks. If a robot could reason about the effects of actions and the future states of objects to manipulate, it would plan feasible actions to navigate objects to their goal configurations. Nonetheless, the ability to reason physical interactions has been a longstanding challenge in robotics. Although many predictive models that leverage fully observable states have shown impressive results for various mechatronic systems [1]–[5], the system states, such as object poses, may not be directly accessible in many real scenarios. In particular, learning a predictive model only from high-dimensional sensory inputs, such as raw RGB images, remains significantly challenging [6], [7]. Furthermore, for multi-object manipulation tasks, predictive models require to consider not only the interaction between the robot and the manipulated objects, but also the interaction between objects, such as collisions between objects [8]. Thus, an effective and robust prediction model should accurately predict the object states in the future conditioned on applied actions, which allows a robot to perform multi-object manipulation tasks for unseen objects with variations in shape and size.

The associate editor coordinating the review of this manuscript and approving it for publication was Yingxiang Liu.

Recently, video prediction models have been successfully applied to perform various manipulation tasks [9]–[11], which directly predict future frames [12] or 2D flow of pixels [13] from raw visual observations. However, these models typically require large amounts of real data, hindering many potential real-world applications. Although such prediction models combined with a model-predictive control (MPC) are applied for predicting future trajectories of object states [14], they are limited to short-term tasks due to the accumulation of model error over prediction time [11]. On the other hand, from a robotic perspective, there is no need to generate the raw pixels of future frames. Directly predicting raw pixels containing a lot of noise, such as background and shadows, can increase the difficulty of predicting the each entity state in the image. Moreover, the pixel-based methods discard the structured knowledge about the world, such as object positions and visual shapes, resulting in inefficient and inaccurate predictions. Therefore, instead of directly predicting raw pixels in image space, the key idea in this paper is to represent each entity with an object-centric representation that contains the object positions and visual features in latent space. Such compact representation allows for an accurate and efficient prediction model for planning plausible actions.

This paper proposes a Deep Object-centric Interaction Network (**DOIN**), a deep neural network that can learn system dynamics of physical interaction and predict object-centric representations of multiple objects through visual observations. The proposed model can abstract object-centric representations for multiple objects in latent space from raw images, and reason about future representations conditioned on the interactions between objects. The object-centric representation for each object contains the explicit position in image space and implicit visual feature that encodes the object information (such as shape, size, and rotation) and the context of the environment. The proposed DOIN model contains two major components: 1) an encoder module that abstracts object-centric representation for each entity, and 2) an interaction prediction network that predicts future object states. DOIN model is trained purely on data collected in simulation through self-supervised random interactions. The learned model combined with a model-predictive control framework can allow a robot to perform multi-object manipulation tasks, such as planar pushing tasks. Simulation experiments demonstrate that the proposed method successfully performs multi-object pushing tasks, and outperforms state-of-the-art baselines. Real-world experiments validate that the model trained only on simulated data can accurately reason the effect of physical interactions and allows the robot to successfully manipulate previously-unseen objects to achieve goal configurations in real world.

The contributions of this paper are presented as follows. First, a novel interaction network is proposed to leverage object-centric representations to predict the effect of multi-object interactions only from RGB observations. Second, DOIN is trained only on simulated data collected through self-surprised random interactions and can be directly transferred to the real-world experiments without any extra effort. Third, extensive experiments in simulation and the real-world demonstrate that the learned model successfully performs multi-object manipulation tasks, substantially outperforming the baselines. Furthermore, the proposed method can generalize to manipulate novel unseen objects with variations in shape.

II. RELATED WORK

A. LEARNING PHYSICAL INTERACTIONS

Learning models that can predict the future states of objects is an active research area. Many approaches on the prediction of physical interactions have been applied from video prediction [15] to robotic manipulation tasks [16]. Some methods build the system dynamics by explicitly modeling the state transitions, leveraging ground-truth poses [17], [18] or known physical properties [19]. However, access to the ground-truth states and physical properties may be difficult for most real-world applications. Recently, some data-driven methods have been proposed to reason about physical interactions from visual observations via predicting the objects' center of mass [20], predicting objects' physical properties via external multi-step interactions [21], or predicting 3D volumetric scene flow [22]. However, most of them only consider isolated objects and cannot be easily extended to multi-object manipulation tasks. On the other hand, some generic approach has been proposed to leverage graph neural network [23] to capture the relations between objects in a scene [24], [25]. For example, Janner *et al.* [8] learn an object-centric dynamics model trained by a physics simulator and show that such models can be utilized for block stacking tasks. However, these methods often rely on a predefined number of objects. Furthermore, they only consider simple scenes like falling blocks, and do not involve applied actions for robotic manipulation tasks, In contrast, the proposed method learns multiple interactions without limiting the number of objects, and can be used for planning action sequences to perform robotic pushing tasks.

B. LEARNING VIDEO PREDICTION

Recently, pixel-based prediction models have shown impressive results in video prediction applications [12], [13]. Some approaches have utilized a transformation-based model to generate pixels by predicting the flow of pixels [12], [26], and have successfully performed robotic manipulation tasks [27]. For example, *VisualMPC* [10] has been proposed to combine a video prediction model with a model predictive control framework to predict the flow of the designated pixels, and has achieved to manipulate unseen objects, even multiple objectives, in the real world. However, such action-conditioned video prediction models typically require a large amount of real data, making it difficult to generalize to different scenes efficiently. Instead of predicting raw pixels, the proposed method leverages object-centric representations for multiple objects, enabling an efficient and accurate prediction for object states. Furthermore, the proposed model combines the object-centric representations with a interaction network to reason about the effect of physical interactions.

C. LEARNING OBJECT-CENTRIC REPRESENTATIONS

Object-centric representations have been widely applied in robotic manipulation applications from pick-and-place [25] to non-prehensile pushing [28]. Recently, a number of learning-based method have leveraged object-centric representations to plan actions, including 6-DOF poses estimator [29], keypoint-based methods [30], [31], and visual descriptors [32], [33]. However, most of these methods are task-specific data collection procedures and are not sufficient to generalize to unseen objects with large variations in shape. The work most related to ours is [34], which also combines an object-centric representation with MPC for pushing objects. However, instead of using a video prediction model, this paper proposes a novel object-centric representation, which combines object pixel positions with the visual features produced by a modified CVAE architecture. The proposed method also shows stronger generalization capability for unseen objects than prior work.

III. OVERVIEW

This paper considers the problem of multi-object robotic manipulation tasks only from visual observations, such as RGB images. Robotic manipulation can be formulated as a Markov Decision Process (MDP) with a high-dimensional state space $\mathcal{S}$ and an action space $\mathcal{A}$, where the dynamics model, i.e., $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$, defines a transition probability of the next state s_{t+1} given the current state-action pair (s_t, a_t). In this paper, the state s_t is processed from raw RGB image I_t and a_t is a planner pushing primitive command of the robot end-effector. The goal of this paper is to learn an interaction prediction model that, given an initial scene and a goal image I_g, is used for planning action sequences such that the objects match the specified goal configurations after executing the push actions.

As illustrated in Fig. 1, the proposed approach for multi-object robotic manipulation tasks contains three phases: self-supervised data collection, latent interaction predictive model training, and planning control via the model at test-time. First, to train DOIN, large amounts of data are collected autonomously by applying random actions in the simulation. A variety of objects with a different number of objects in a scene are used for data collection. The synthetic data does not require any human annotation or extra supervision. Second, during the training phase, the model is trained on the collected data. The model takes as input the current image and abstracts an object-centric representation for each entity. Using this representation, the model can predict the future states under applied actions. Last, at test time, an MPC method is used to search the action sequences that can minimize the task-specific cost function. The robot executes the action a_t^* each step after receiving the current image I_t in a closed-loop manner until the task goal is achieved.

IV. METHOD

This section first presents a description of the proposed DOIN that uses object-centric representations to perform interaction reasoning and predicts the object states. Then, a sampling-based planner combined with the learned model is proposed to perform robotic multi-objects manipulation tasks.

A. DEEP OBJECT-CENTRIC INTERACTION NETWORK

1) LATENT OBJECT-CENTRIC REPRESENTATIONS

In order to accurately predict the effect of physical interaction between objects and applied actions, object representations require an implicit understanding of the dynamics between the object and the robot and how they affect the future object motion. To this end, an object-centric representation is applied to denote an individual entity in a scene, as shown in Fig. 1. Concretely, given N objects in a scene, an object-centric representation is denoted as $\{x_t^i\}_{i=1}^N$. The representation for i-th object at the t time step consists of the explicit spatial location $p_t^i \in \mathbb{R}^2$ and the implicit visual feature $f_t^i \in \mathbb{R}^{d_f}$ that encodes the object information, such as shape and orientation, where d_f is the dimension of visual feature. In this paper, an advanced instance segmentation algorithm [35] is implemented to extract the region of interest (ROI) for each object from a raw RGB image. The p_t^i is assumed to be the center of the corresponding ROI, denoted as a 2-dimensional xy-coordinate in image space. The visual feature f_t^i can be extracted from the corresponding ROI of the current image I_t and the context image (i.e., initial image I_0) through the encoder modules. The context image provides a description of different scenes so that the representations not only contain the own object information but also the environmental information. Therefore, i-th object is represented as $x_t^i \equiv (p_t^i, f_t^i) \in \mathbb{R}^{2+d_f}$, where N denote the number of objects in a scene.

2) ENCODER MODULE

The encode modules consist of two encoder networks to extract object-centric representations for multiple objects in latent space. Given the current image I_t and the initial image I_0 with size $3 \times H \times W$, each object's corresponding ROIs, i.e, $\{^{\mathrm{ROI}}I_0^i\}_{i=1}^N$ and $\{^{\mathrm{ROI}}I_t^i\}_{i=1}^N$, can be extracted from an off-the-shelf segmentation algorithm. The two encoder networks take in the ROIs and output the latent features $\{z_c^i\}_{i=1}^N$ and $\{z_t^i\}_{i=1}^N$ in latent space, respectively. These latent features are concatenated to the final visual features f_t^i, i.e., $f_t^i \equiv (z_t^i, z_c^i)$. Besides the visual features, the object-centric representation is the concatenation of visual features and XY coordinates in image space. In this work, the dimension of visual features is $d_f = 96$, and the representation for each object is a 98-dimension vector, i.e., the dimension of the x_t^i is d = 98.

3) INTERACTION NETWORK

The interaction network predicts the future representations for each object by considering physical interactions between objects. The future state of an object depends not only on

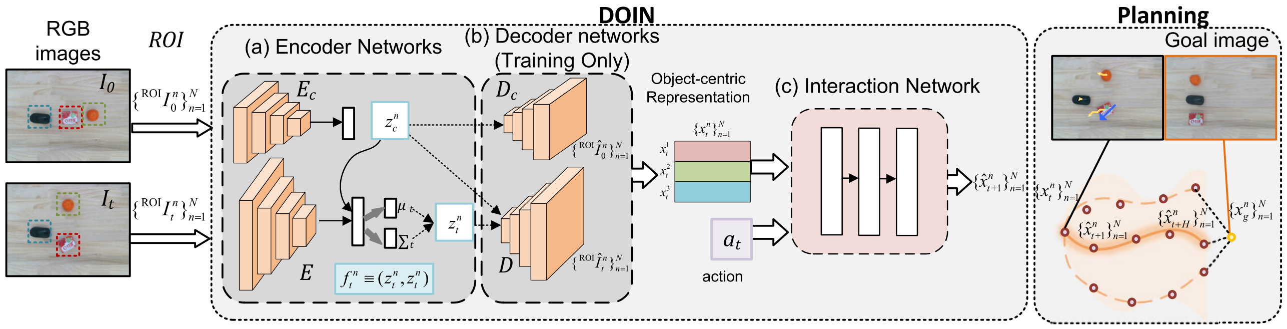
RGB images
ROI
I_0
I_t
$\{^{ROI}I_0^n\}_{n=1}^N$
$\{^{ROI}I_t^n\}_{n=1}^N$
DOIN
(a) Encoder Networks
(b) Decoder networks (Training Only)
E_c
E
z_c^n
z_t^n
μ_t
Σ_t
$f_t^n \equiv (z_c^n, z_t^n)$
D_c
D
$\{^{ROI}\hat{I}_0^n\}_{n=1}^N$
$\{^{ROI}\hat{I}_t^n\}_{n=1}^N$
Object-centric Representation
$\{x_t^n\}_{n=1}^N$
(c) Interaction Network
a_t
action
$\{\hat{x}_{t+1}^n\}_{n=1}^N$
Planning
Goal image
$\{x_t^n\}_{n=1}^N$
$\{\hat{x}_{t+1}^n\}_{n=1}^N$
$\{\hat{x}_{t+H}^n\}_{n=1}^N$
$\{x_g^n\}_{n=1}^N$

FIGURE 1. System Overview. For N objects, the proposed DOIN takes in the corresponding ROIs of the current image I_t and the initial image I_0 for each object as input, and outputs object-centric representations through the encoder networks (a). Then the model can predict the future representations conditioned on applied actions by interaction networks (c). During the planning phase, an MPC planner is used to plan an action sequence that minimizes the cost function to achieve the manipulation task. Additionally, the decoder networks are used for training (b).

the applied actions but also on the physical interaction with the surrounding objects. Assuming that there exist N objects in a scene, with object-centric representations as $\{x_t^i\}_{i=1}^N = \{x_t^1, x_t^2, \ldots, x_t^N\}$. The interaction model of each object and its surrounding object can be characterized as a single-layer fully connected neural network

$$\mathcal{F}_I\left(x_t^i, x_t^j, a_t\right) = \text{ReLU}\left(W_I^T\left[x_t^i, x_t^j, a_t\right]\right) \tag{1}$$

where W_I^T is a learnable weight and $\mathcal{F}_I$ is a single-layer fully network with ReLU activations. For object i, its own representation x_t^i and interaction reasoning with surrounding objects are forwarded to a multi-layer fully connected network to predict the future object state:

$$\hat{p}_{t+1}^i = \mathcal{F}_p\left(\mathcal{F}_h\left(x_t^i\right) + \sum_{j\in\mathcal{N}(i)} \mathcal{F}_I\left(x_t^i, x_t^j, a_t\right)\right) + p_t^i \tag{2}$$

$$\hat{f}_{t+1}^i = \mathcal{F}_f\left(\mathcal{F}_h\left(x_t^i\right) + \sum_{j\in\mathcal{N}(i)} \mathcal{F}_I\left(x_t^i, x_t^j, a_t\right)\right) \tag{3}$$

where $\mathcal{F}_h$, $\mathcal{F}_f$ and $\mathcal{F}_f$ are two fully connected layers with 64 hidden units and ReLU activation. $\mathcal{N}(i)$ stands for a set of the neighboring objects. It should be noted that the interaction model $\mathcal{F}_I$ is only used when the Euclidean pixel distance between the two objects in image space is smaller than a certain threshold, i.e.,

$$\mathcal{N}(i) = \{j \mid \|p_j - p_i\|_2 \leq \zeta, j \neq i\} \tag{4}$$

where ζ is the pixel threshold.

The interaction module uses fully connected networks to predict the change of the position state Δp_t and the next visual feature f_t. And the next position $\hat{p}_{t+1}$ is computed by adding the current state p_t to the predicted change of the position state Δp_t. For similarity, the interaction modules are denoted as

$$\{\hat{x}_{t+1}^i\}_{i=1}^N = \mathcal{F}\left(\{x_t^i\}_{i=1}^N, a_t\right). \tag{5}$$

The learned interaction module can be used for predicting future trajectories of object states with a length of H in latent space via applying it recurrently over H time steps conditioned on a sequence of actions $a_{t:t+H-1}$.

4) LOSS FUNCTIONS

The loss function used for training uses a weighted combination of three components: 1) a reconstruction loss $\mathcal{L}_{\text{recon}}$, 2) a Kullback-Leibler (KL) loss $\mathcal{L}_{\text{KL}}$, and 3) a dynamics loss $\mathcal{L}_{\text{dyn}}$. The reconstruction loss is similar to a conditional variational autoencoder (CVAE) [36]:

$$\mathcal{L}_{\text{recon}} = \sum_{i=1}^N \left\|\mathcal{D}(z_t^i) - {}^{ROI}I_t^i\right\|_2^2 + \sum_{i=1}^N \left\|\mathcal{D}_c(z_0^i) - {}^{ROI}I_t^i\right\|_2^2 \tag{6}$$

where z_0^i and z_t^i are latent features of the i-th object extracted from the E_c and E, respectively. In addition, the network is also learned by minimizing the KL divergence between the latent distribution and Gaussian distribution. Importantly, to improve the perdition accuracy for future object states, a dynamics loss is used. The dynamics loss can be expressed as:

$$\begin{aligned}\mathcal{L}_{\text{dyn}} = &\sum_{i=1}^N \lambda_2 \left\|\hat{p}_{t+1}^i - p_{t+1}^i\right\|_2^2 \\ &+ \sum_{i=1}^N \lambda_1 \left\|\mathcal{D}(\hat{z}_{t+1}^i) - {}^{ROI}I_{t+1}^i\right\|_2^2 \\ &+ \sum_{i=1}^N \lambda_3 \left\|E({}^{ROI}\hat{I}_{t+1}^i) - z_{t+1}^i\right\|_2^2 \\ &+ \sum_{i=1}^N \lambda_4 \left\|\mathcal{D}(\hat{z}_{t+1}^i) - {}^{ROI}I_{t+1}^i\right\|_2^2\end{aligned} \tag{7}$$

where λ_1, λ_2, λ_3, and λ_4 are weighted factors. The first term minimizes the average Euclidian distances between the predicted and ground-truth positions of the multiple objects for each step. The second term encourages minimizing the ℓ_2 loss between predicted visual images and the ground-truth images. The third item minimizes the average ℓ_2 loss between the prediction z^i_{t+1} and the next encoded states $E(^{ROI}\hat{I}^i_{t+1})$ in latent space. The fourth item minimizes the average ℓ_2 loss between the reconstructed image $\mathcal{D}(\hat{z}^i_{t+1})$ and the next ground-truth image $^{ROI}I^i_{t+1}$. Using a weighted combination of these loss functions, i.e, $\mathcal{L} = \mathcal{L}_{\text{recon}} + 0.1\mathcal{L}_{\text{KL}} + \mathcal{L}_{\text{dyn}}$, the whole training process of the proposed model can be fully self-supervised without any human intervention.

Algorithm 1 Planning via DOIN

```
Input: predictive model $\mathcal{F}$, cost function $C$, goal image $I_g$,
    and initial action Gaussian distribution $\mathcal{N}(0, I)$
1: for $t \in [0, H-1]$ do
2:     for $i \in [0, n_{iter}-1]$ do
3:         Sample $M$ action sequences $\{a^i_t, \ldots, a^i_{t+H-1}\}^M_{i=1}$
           from the action Gaussian distribution.
4:         Predict states $\{x^i_{t+1}, \ldots, x^i_{t+H}\}^N_{i=1}$ with sampled
           actions and current representation $\{x^i_t\}^N_{i=1}$ by recur-
           sively using DOIN model as Equation 5.
5:         Evaluate action sequences using a cost function $C$.
6:         Refit the action Gaussian distribution to the $K$ action
           samples with lowest cost.
7:     end for
8:     Execute the first action of the best action sequences $a^*_t$
9: end for
```

B. PLANNING VIA DOR-NN IN ROBOT MANIPULATION

During planning, the goal of the manipulation task is to find a sequence of actions such that executing them to manipulate multiple objects to the desired goal configurations. A model predictive control framework uses the learned model to predict future states of multiple objects and plans plausible actions. Given the learned interaction model $\mathcal{F}$ and a sequence of sampled actions, future position trajectories of the objects with a length can be recursively predicted by the model $\mathcal{F}$ using Equation 5. Each trajectory has length H, i.e., $\hat{x}^i_{t+1:t+H}$. A sampling-based optimizer is used to plan action sequences $a_{1:T}$ that minimize the sum of the costs $C_{1:T}$ along the planning horizon H. Although there exists a variety of trajectory optimization methods in the literature, this paper uses a cross entropy method (CEM) [37], a simple stochastic optimization, gradient-free optimization procedure. The process often includes randomly resampling actions sequences and refitting Gaussian distributions to selected actions according to the results of the cost function. The whole planning process is illustrated in Algorithm 1. First, M action sequences are sampled from a Gaussian distribution. Then the future position trajectories of the objects $\{x^i_{t+1}, \ldots, x^i_{t+H}\}^N_{i=1}$ is predicted by the interaction model $\mathcal{F}$ conditioned on the sampled actions. Second, the action sequences are evaluated by computing the task-specific cost function C. The Gaussian distribution of the actions is refit based on the best K action sequences. Finally, only the current best action a^*_t is applied to the robot, and then the robot receives a new observation and acts in a closed-loop manner.

The cost function is defined as a weighted combination of average Euclidean distance to the goal configurations for multiple objects in latent space. The cost function consists of two parts: position distance and visual feature distance, as follows:

$$C=\sum_{t=1}^{T} c_t=\sum_{t=1}^{T}\sum_{i=1}^{N}[\left\|\hat{p}^i_{t+H}-p^i_g\right\|_2^2+\lambda_c\left\|\hat{f}^i_{t+H}-f^i_g\right\|_2^2] \quad (8)$$

where λ_c is weighting factor; p^i_g and f^i_g are two parts of the object-centric representation of the goal configuration. The expected position distance to the goal provides effective information that enables the robot to manipulate the object toward the target direction. The latent feature distance encourages the objects to match the goal image and implicitly encodes the object positions in latent space, which can be considered as a supplement to the position distance.

V. EXPERIMENTS

In this section, both simulation and the real-world experiments are conducted to assess the perception and manipulation performance. Videos for the quantitative examples can be found in the supplementary material.

The goal of the experimental evaluation is to answer four primary questions: (1) How accurately does the proposed model DOIN predict future states of multiple objects? (2) Does the model using the interaction network outperform alternative methods *without* interaction network? (3) Compared to the benchmarked methods, can the proposed approach improve the generalization capability for unseen multi-object manipulation tasks? (4) Can the proposed approach be transferred to the real world and enable a robot to perform manipulation tasks?

The question (1) is addressed through both simulation and real-world evaluations (Sec. V-C), questions (2)(3) are addressed through qualitative experiments compared with baseline methods(Sec. V-D), and the last question is addressed through real-world experiments (Sec. V-E).

A. EXPERIMENTAL SETUP

The proposed method is evaluated both in simulation and real-world experiments. The experiments are evaluated on a series of tabletop multi-object pushing tasks. The objective of tasks is to push multiple previously-unseen objects such that the objects achieve the goal configurations. The experimental setup of manipulation tasks includes a Kinova Jaco2 7DOF robot arm with an attached cylinder tool and an overhead RGB camera both in simulation and the real world, as shown in Fig. 2. In the simulation, the robot and camera layout is consistent with the real-world experiments, according to the real calibration results. In the real world, an Intel RealSense D435i camera is used to capture only RGB images. The object ROI is obtained by a segmentation algorithm Detectron2 [35]. For observation space, the raw images with a resolution 480 × 640 are fed to the model. For action space, each action is defined as a series of primitive movements that can

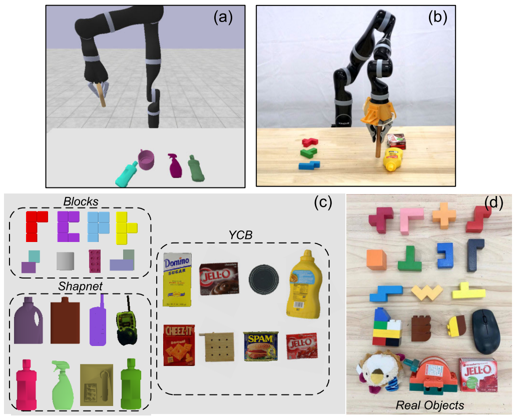

FIGURE 2. Experimental Setting and Testing Objects. (a)(b) Simulation and real-world experimental setting. (c) Simulation test objects: *ShapeNet*, *Blocks* and *YCB*. Except for *ShapeNet*, the others have never been seen during training. (d) Real-world objects. These datasets are used to evaluate manipulation performance of the proposed method.

achieve a straight line pushing motion along x-and y-axes. The action is represented as a 4-dimension tuple consisting of the initial XY coordinates and the relative movements of the end-effector.

1) NETWORK ARCHITECTURE

The encoder modules contain two encoder networks. Two encoder networks have the same architecture and take as input the cropped image with a size of 64×64. Each network contains three conventional layers with 16, 32, 64 channels, respectively, and kernels size of 5, 3 and 3, followed 3 fully connected layers of size $[512, N_z]$ where N_z is the size of the latent space (mean and variance). Different N_z for two encoder networks are used, i.e., $N_{z_t} = 64$ and $N_{z_c} = 32$. The decoder network D takes in a concatenation of z_t^i and z_c^i and feeds it through 3 fully connected layers [98, 256, 512], followed by three de-conventional layers with 32, 16, 16s channels, respectively, and kernels size of 2×2. The stride sizes of the first and second de-convolutional layers are 2×2. Another layer has a stride of 4×4 and a kernel size of 4×4. All layers except the final layer use batch normalization and ReLU activation. For the interaction network, pixel threshold $\zeta = 96$ corresponds about 15cm.

2) IMPLEMENTATION DETAILS

The proposed model is implemented using PyTorch. All the networks in the experiments are trained for 100 epochs with the batch size 2048. Optimization is carried out using ADAM with $\beta_1 = 0.9$ and $\beta_2 = 0.95$. An initial learning rate of 10^{-3} with a decay of 0.5 after 30 steps is used. During the planning, $M = 200$, $H = 5$, $K = 10$, $n_{iter} = 3$ and $\lambda_c = 0.1$ are used. During test time, the proposed model is run on a laptop with Intel i7 2.2 GHz CPU and an NVIDIA GTX 1060 Ti GPU, which is efficient enough for real-time control.

B. DATA COLLECTION

In order to collect large amounts of interaction data for training and evaluating the model, a self-supervised data collection framework is developed to generate synthetic training data automatically. The simulation environment is implemented on an open-source physics simulator PyBullet as described in Sec. V-A. During collection, multiple objects are randomly placed on the workspace, and then the robot executes random pushing actions. The raw images are recorded before and after applying each action. The data are collected in the form of triplets (I_t, a_t, I_{t+1}), where the a_t denotes the random action after receiving the image I_t.

To collect meaningful interaction data, a heuristic random policy like [20] is utilized to generate random actions. The policy first uniformly samples a pixel inside the object mask and sets the pixel as the action's end position. A line segment with one end being the pixel position is then randomly sampled. If the segment's starting position does not fall in the mask of all objects, then it deems as the feasible action. Otherwise, random sampling is repeated until a feasible action is obtained. This policy makes the end position of each push fall inside the object mask, ensuring that each action can change the object states.

1) TRAINING DATASET

A subset of ShapeNet dataset [38] is used for data collection during training. 38 objects of 5 categories with variations in shape are collected: mug, bottle, can, sofa and phone. For each scene, three objects are randomly sampled from the dataset and are randomly placed on the table. To further improve robustness, many aspects of the objects are randomized: color, size, initial positions, and orientations. Each episode has a length of 12 time steps. An episode is early terminated when the object is outside the boundary of the workspace, or the object is flipped. In total, 30K episodes (approximately 360K pushes in total) are collected for training.

2) TESTING DATASET

To evaluate the perception performance of the proposed method, three simulated test sets are collected: *ShapeNet*, *Blocks*, and *YCB* [39], as shown in Fig 2. For *Blocks* and *YCB*, each test set contains 8 previously-unseen objects with different shapes, where the objects in *YCB* have larger sizes and are closer to common everyday objects. For each test set, the interaction data are collected from 4 different scenes with a different number of objects (i.e., 2 to 5). Each scene has 100 episodes with approximately 1200 interactions in the simulation. In total, ~14.4K synthetic interaction data are collected for testing.

C. PREDICTION EVALUATION

1) BASELINES

To evaluate the prediction performance of multi-object states, the proposed approach is compared with the two baselines: 1) **SNA** [9]: An advanced video prediction model directly predicts the 2D flow of the pixels from raw images, which has been successfully applied in manipulation tasks in the

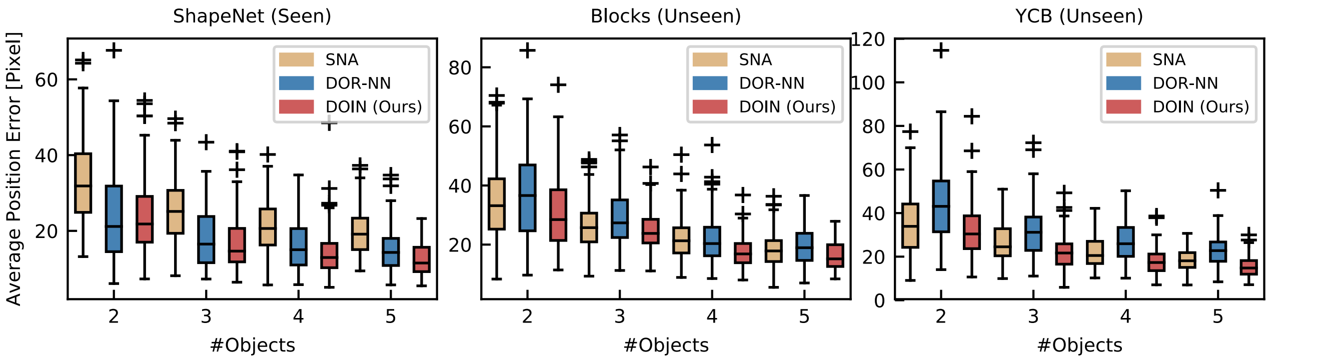

FIGURE 3. Qualitative Results for Three Test sets Box-plots of the average position error (APE) for the predicted object states under 12 scenes with four different numbers of objects from 2 to 5 and three test sets. The crosses represent outliers. Units are pixels in the 480 × 640 images. 20 pixel corresponds to around 3.5cm.

real-world [10]. 2) **DOR-NN**: A dynamic object-centric representation network *without interaction modules* is similar to the proposed model, which uses the same object representations to estimate future object states. 3) **DOIN (Ours)**: The proposed object-centric method *with interaction networks* predicts future object representations conditioned on the applied actions and interaction relations.

2) METRIC

All methods are evaluated in terms of the perception accuracy of the object positions. Given the future predicted positions for N objects $\hat{p}$ in image space, **Average Position Error (APE)** is used as metric: average Euclidean distance between the predicted and the ground-truth positions over the prediction horizon T:

$$\text{APE} = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\left\|\hat{p}_t^i - p_t^i\right\|^2 \tag{9}$$

For comparison, all the compared methods are trained on the same simulated dataset. Three simulated test sets are used for testing. All the methods use the first 2 ground-truth frames and predict the next 9 frames conditioned on the same actions.

3) PERCEPTION RESULTS

The qualitative results of the perception performance are shown in Fig. 3. The median and third quartile (Q3) of APE metric are summarized in Table 1. From the results, compared with the other baselines, the proposed model DOIN has relatively small median errors and small outliers (Q3) in terms of APE for overall test sets and the most scenes with varying numbers of objects. For seen *ShapeNet* objects that are used for training, the average position errors of DOR-NN are slightly worse than the proposed method. However, for previously-unseen objects (*Blocks* and *YCB*), the prediction accuracy of the proposed model with interaction networks substantially outperforms the alternative methods that do not consider physical interactions. Moreover, as the number of objects increases, the proposed method achieves better object position perdition in image space than baselines.

TABLE 1. Prediction Results for multi-object interactions. The numbers in the table show median and third quartile (Q3) of the average position error (pixel 480 × 640 image).

Datasets	Methods	Number of Objects #2	#3	#4	#5
Blocks	SNA	33.2 (42.3)	25.7 (30.6)	21.3 (25.7)	17.9 (21.4)
	DOR-NN	36.6 (46.9)	27.4 (35.1)	20.3 (25.9)	18.9 (23.8)
	DOIN	**28.5 (38.6)**	**23.8 (28.6)**	**16.8 (20.4)**	**15.2 (20.0)**
Shapenet	SNA	31.9 (40.4)	25.1 (30.7)	20.6 (25.8)	19.1 (23.4)
	DOR-NN	**21.2 (31.8)**	16.5 (23.8)	15.0 (20.6)	14.3 (18.0)
	DOIN	21.8 (29.1)	**14.7 (20.7)**	**13.0 (16.7)**	**11.5 (15.7)**
YCB	SNA	33.9(44.2)	24.5 (32.8)	20.5 (27.0)	18.1 (21.8)
	DOR-NN	43.1 (54.7)	31.2 (38.2)	25.9 (33.4)	22.8 (26.7)
	DOIN	**30.4 (38.7)**	**21.7 (25.9)**	**17.3 (21.2)**	**14.8 (18.2)**

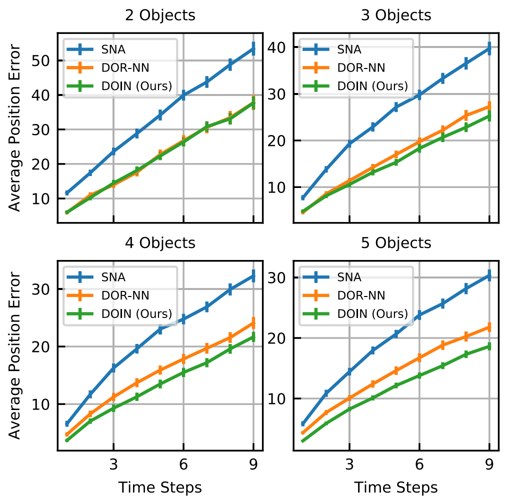

FIGURE 4. Average Position Error over prediction steps on *ShapeNet*. Units are pixels in the 480 × 640 images.

For example, for *YCB* objects with larger shapes, the proposed method outperforms the second-best method by a large margin of at least 18.2% in APE metric. These results show that the proposed method using interaction networks can significantly improve perception accuracy for multi-object scenes.

Fig. 4 illustrates the prediction results of baselines over time steps. Although when the prediction step increases the

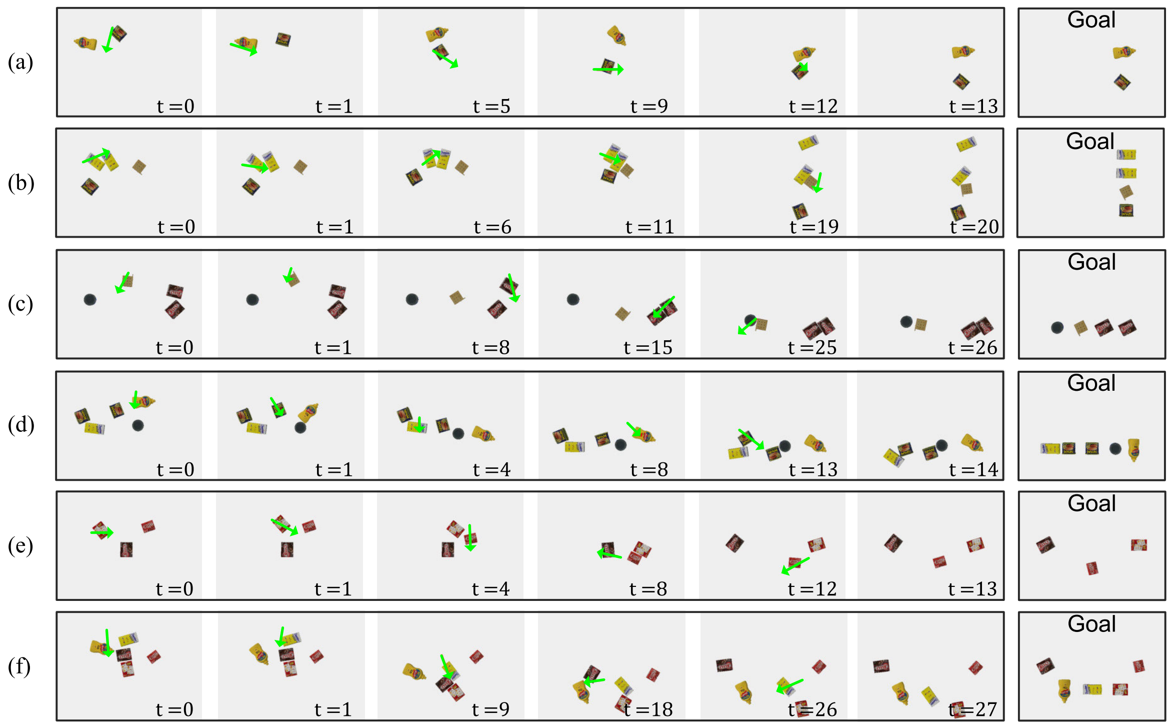

FIGURE 5. **Snapshots of Multi-object Pushing Manipulation in Simulation. The robot executes the planned actions (green arrows) and pushes the unseen objects to match the human-specified goal images. Simulation experiments show that the proposed model can enable the robot to push multiple unseen objects for different tasks with different numbers of objects: *Task 1* (a,b), *Task 2* (c,d), and *Task 3* (e,f).**

position error compounds, the prediction accuracy of the proposed method for the whole episode is significantly better than compared methods. Moreover, when the number of objects increases, the probability of collisions between objects increases, which leads to deterioration of the prediction accuracy of the compared methods that do not reason about the physical interactions between objects. In contrast, it can be seen that the proposed method can also achieve higher prediction accuracy than alternatives when the number of objects increases. These results show the proposed method can accurately predict object positions for unseen objects via reasoning about the physical interactions between objects.

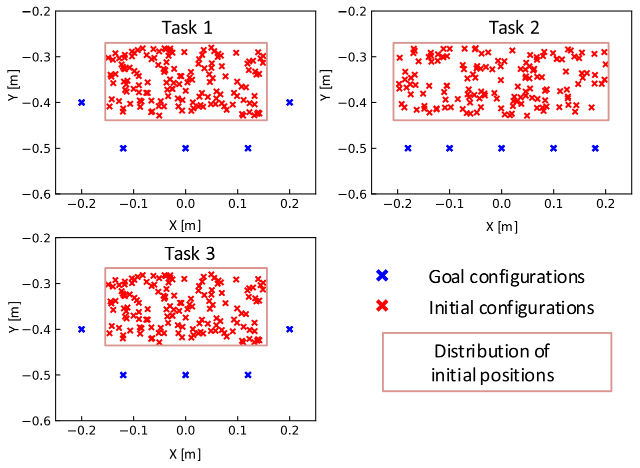

FIGURE 6. **Tasks. Three pushing tasks are defined in simulation experiments. For each trial, the initial positions of objects are randomly sampled from the red region and their goal configurations are set to the blue points.**

D. SIMULATION EXPERIMENTS

To assess the manipulation performance of the proposed method, extensive experiments on multi-object pushing manipulation tasks are conducted using the simulated setup described previously. In the setting, multiple objects are randomly placed on the specified initial area, and their goal positions are specified. The goal image is captured by placing the objects at the goal positions. The objective of the experiments is to push multiple objects to match the target configurations given the goal image.

1) TASKS

According to the different target configuration distributions as shown in Fig. 6, three tasks are designed to evaluate all baselines: *Task 1*, rearrange multiple objects into a column on the right; *Task 2*, rearrange multiple objects into a row on the bottom; *Task 3*, rearrange multiple objects on both sides of the workspace. Since the workspace's width is less than the length, it is more likely to collide among multiple objects during the execution of *Task 2*, making it more difficult than *Task 1*. Similarly, *Task 3* requires manipulating multiple objects to reach left and right sides of the workspace, leading to increasing the difficulty of the task.

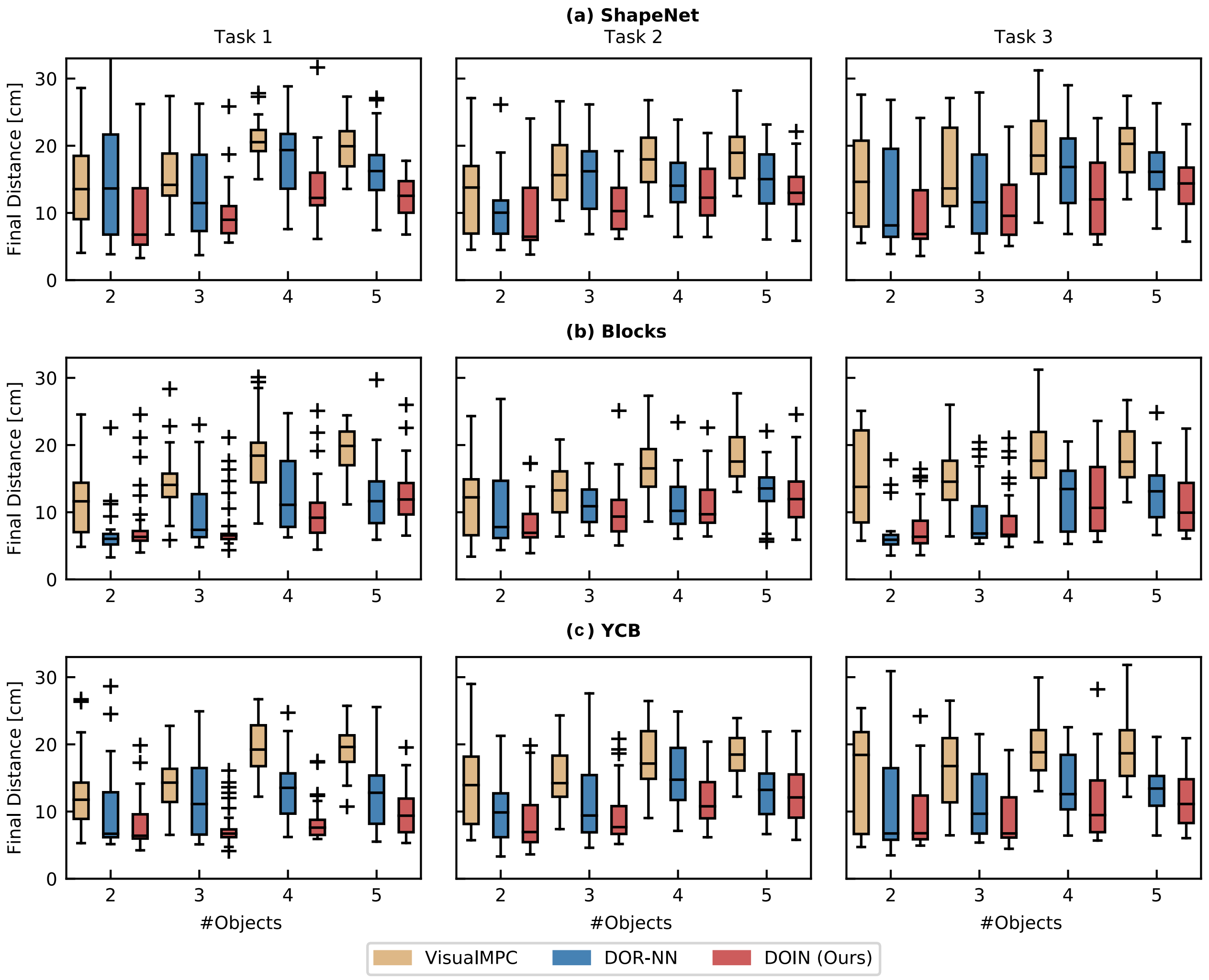

FIGURE 7. Quantitative Results of Simulated Experiments. Box-plots of the average final distance to goals for all scenes. We compare performance separately on three different pushing tasks using three object categories as shown in Figure 2: *ShapeNet*, *Blocks* and *YCB*. 30 trials for each scene are run with a different number objects for each method, for a total of 1080 trials. The proposed model outperforms all baselines using the three object categories in both tasks. In the plot, the crosses represent outliers.

2) BASELINES

Three baselines are used to showcase the effectiveness of the proposed approach.

- **VisualMPC** [10]: A state-of-the-art model-based method uses a flow-based video prediction model (SNA), as described in Sec. V-C, to predict the 2D flow of the designated object pixels. For each step, the designated object pixel is assumed to be the center of the ROI, so it is actually stronger than the previous method [10] that does not capture the ground-truth pixels. The video model is trained on the same simulated data as ours and also leverages a CEM method to plan a sequence of actions.
- **DOR-NN**: An object-centric dynamics network *without interaction modules* uses the same object-centric representations to predict future object states as ours. A similar MPC palnner is used to plan actions.
- **DOIN (Ours)**: The proposed multi-object dynamic network with interaction modules predicts the object-centric representations for multiple objects and a CEM planner to plan the actions given a goal image.

3) METRIC

To evaluate the performance for multi-object manipulation tasks, **Average Final Distance** is defined as the metric to measure the average distance between the final positions of all objects and the target positions like the previous work [10], [20].

To evaluate generalization of the proposed method, all baselines are evaluated using the three test sets: *ShapeNet*, *Blocks* and *YCB*. Except for *ShapeNet*, the rest of the test sets have never previously been seen during training. Each test set contains 8 different objects with variations in shape. Each test has 12 scenes containing four different numbers of objects from 2 to 5 and three different tasks. In total, 36 different scenes are used for evaluating all methods. For a fair comparison, the initial positions and goals of all baselines are the same for each scene. For each scene, 30 trails are run with different random seeds.

TABLE 2. Quantitative performance of simulation experiments on YCB test set. The median and third quartile (Q3) of average final distances to goals are reported in the table. All the units are expressed in centimeter.

#Objects	Task 1			Task 2			Task 3		
	VisualMPC	DOR-NN	DOIN	VisualMPC	DOR-NN	DOIN	VisualMPC	DOR-NN	DOIN
2	12.3 (5.2)	10.3 (6.1)	**7.9 (3.7)**	13.8 (6.1)	10.3 (5.1)	**8.6 (4.0)**	15.7 (7.3)	10.8 (7.0)	**9.2 (4.9)**
3	14.2 (3.9)	12.5 (6.1)	**7.7 (3.0)**	14.9 (4.3)	11.8 (6.2)	**9.7 (4.3)**	16.1 (5.8)	11.3 (5.3)	**8.9 (4.3)**
4	19.7 (3.7)	13.4 (4.8)	**8.5 (3.0)**	17.8 (4.8)	15.5 (4.7)	**11.6 (4.2)**	19.5 (4.6)	13.6 (4.8)	**11.2 (5.4)**
5	19.5 (3.5)	12.4 (5.2)	**10.0 (3.5)**	18.5 (3.3)	12.8 (3.9)	**12.2 (3.9)**	19.0 (4.8)	13.6 (3.8)	**11.7 (4.2)**

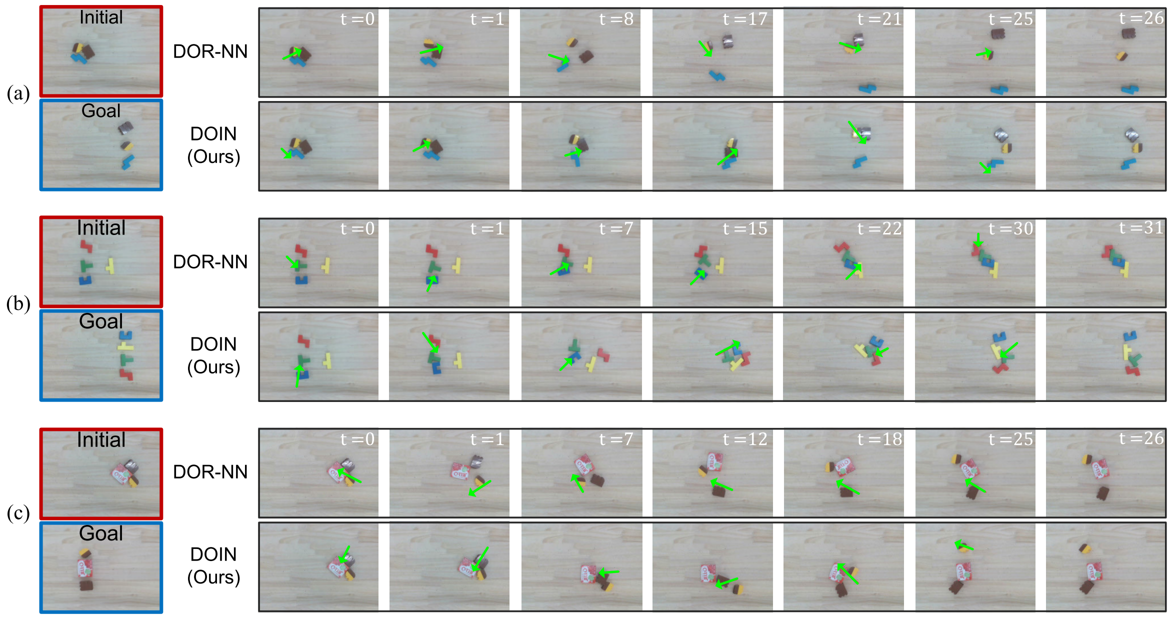

FIGURE 8. Real-world experiments Given the initial configuration (in red box) and the goal configuration (in blue box), the plot shows the results of two baselines at different time steps.

4) QUANTITATIVE RESULTS

Fig. 5 shows some examples for pushing multiple objects in the simulation. The visualization suggests the proposed method can successfully push multiple objects to reach designated goal configurations and generalize to unseen objects with variations in shape and color. The quantitative result of all baselines in terms of the average final distance is illustrated in Fig. 7. The results show that the proposed model with the interaction network achieves lower average final distances for most scenes of all test sets, significantly outperforming the state-of-the-art video prediction *VisualMPC* and the alternatives without reasoning about physics interaction. Moreover, as the number of objects or the difficulty of the task increases, the overall performance of the compared methods deteriorates. In contrast, although the final distance of the proposed method increases as the number of objects increases, the final distances are significantly lower than the other baselines, especially when there are more objects in a scene. In particular, the statistical results of *YCB* objects are summarized in Table 2. For *Task 3* using YCB objects with relatively large shapes, it can be seen from the table that the final distance of the proposed method is lower than DOR-NN by approximately 14%. These extensive experiments demonstrate that the proposed method can generalize to perform multi-object pushing tasks for unseen shape variations, achieving better performance than baselines.

E. REAL-WORLD EXPERIMENTS

To validate the performance of the proposed method in the real world, multiple trials are conducted in different scenes on the real robot Kinova Jaco2. The experimental setting is consistent with the simulation. The experiments involve 18 real objects including 11 relatively small building blocks and 7 everyday common objects with large shapes (Fig. 2). Two scenes with different numbers of objects (i.e., 3 and 4) are used for evaluating. 10 trials are run for each scene. For each trial, multiple objects are randomly placed in the workspace, and the goals are set at positions that are at least 20cm away from their initial positions.

1) RESULTS

Fig. 8 shows several examples of pushing real objects. The quantitative results are illustrated in Fig. 9. These results show that the proposed model trained only on simulated data allows the robot to perform previously-unseen object pushing tasks in the real world successfully. For comparison, *VisualMPC* and DOR-NN are implemented as baselines trained on the same simulated data as ours. From the Fig. 9, the proposed

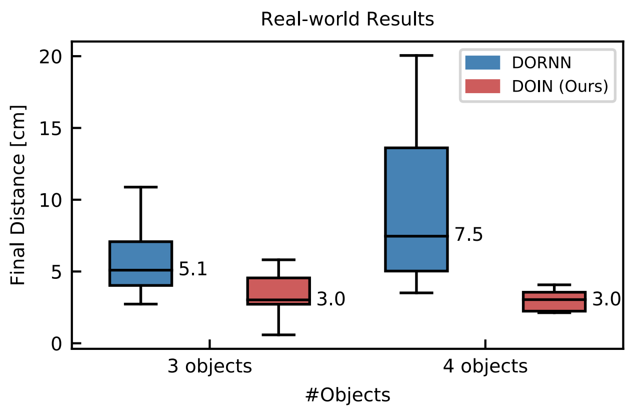

FIGURE 9. **Performance of real-world experiments. We compare our model with other baseline in terms of average final distance. Two scenes with 3 and 4 objects are used for evaluating. For each scene, 10 trials are run. The numbers show the medians.**

method substantially outperforms the baselines in term of average final distance for all scenes. In particular, for the 4-object scene, compared to DOR-NN that does not consider the physical interactions, the average final distance of the proposed method is about 60% lower than the one of DOR-NN (3.0cm versus 7.5cm). Moreover, it can be found that the future object states generated by *VisualMPC* nearly had large deviations in most cases, which led to a complete failure to perform the tasks. The explanation is that the video prediction model trained on simulated data is insufficient to generalize to predict states of real objects. Thus, a significant prediction error compounds as the planning step increases, resulting in planning invalid actions. These experiments demonstrate that the proposed model can accurately predict the effect of the physical interactions between objects and applied actions, and can provide sufficient generalization to plan plausible actions for unseen objects with shape variations in the real world.

2) FAILURE ANALYSIS

The primary failure modes of real-world experiments are analyzed. First, most failure cases result from the objects out of the workspace boundary. Due to the workspace limitation, as the number of objects increases, the possibility of objects out of the boundary increases. This issue could be alleviated by expanding the workspace of the robot end-effector. Second, when a large object has been pushed to the vicinity of the target position, the robot may not plan an effective action to reach the object due to prediction errors. This problem can be alleviated by adding real data to the train dataset to improve the prediction accuracy. Alternative solution is to leverage 3D information to build a dynamic prediction model [22]. Finally, several failure cases result from the undesired collisions between the tool at the end-effector and the object, like directly poking on the object. This problem can be solved by adding an action constraint during the planning phase, ensuring the sampled actions keep a certain safe distance between the starting position and the object.

VI. CONCLUSION

This paper has presented an approach that learns a deep interaction network to predict the effect of physical interactions between objects and the robot using object-centric representations. The proposed model DOIN can achieve accurate and efficient predictions, enabling the robot to perform unseen objects pushing manipulation tasks successfully. The model is trained on simulated interaction data through the self-supervised procedure without any human annotations and can be transferred to the real world. Extensive simulation experiments have shown that the proposed model outperforms the state-of-the-art baselines in terms of perception and manipulation performance. Furthermore, the experiments have validated that the proposed approach can successfully perform multiple multi-objects pushing tasks in the real world and generalize to never-before-seen objects with variations in shape.

REFERENCES

[1] M. Andrychowicz, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," *Int. J. Robot. Res.*, vol. 39, no. 1, pp. 3–20, 2020.

[2] C. Hu, T. Ou, Y. Zhu, and L. Zhu, "GRU-type LARC strategy for precision motion control with accurate tracking error prediction," *IEEE Trans. Ind. Electron.*, vol. 68, no. 1, pp. 812–820, Jan. 2021.

[3] C. Hu, T. Ou, H. Chang, Y. Zhu, and L. Zhu, "Deep GRU neural network prediction and feedforward compensation for precision multiaxis motion control systems," *IEEE/ASME Trans. Mechatronics*, vol. 25, no. 3, pp. 1377–1388, Jun. 2020.

[4] M. Q. Mohammed, K. L. Chung, and C. S. Chyi, "Review of deep reinforcement learning-based object grasping: Techniques, open challenges, and recommendations," *IEEE Access*, vol. 8, pp. 178450–178481, 2020.

[5] T. Ou, C. Hu, Y. Zhu, M. Zhang, and L. Zhu, "Intelligent feedforward compensation motion control of maglev planar motor with precise reference modification prediction," *IEEE Trans. Ind. Electron.*, early access, Aug. 7, 2020, doi: 10.1109/TIE.2020.3013795.

[6] B. Ichter and M. Pavone, "Robot motion planning in learned latent spaces," *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 2407–2414, Jul. 2019.

[7] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 2555–2565.

[8] M. Janner, S. Levine, W. T. Freeman, J. B. Tenenbaum, C. Finn, and J. Wu, "Reasoning about physical interactions with object-oriented prediction and planning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–12.

[9] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 64–72.

[10] F. Ebert, C. Finn, S. Dasari, A. Xie, A. Lee, and S. Levine, "Visual foresight: Model-based deep reinforcement learning for vision-based robotic control," 2018, *arXiv:1812.00568*. [Online]. Available: http://arxiv.org/abs/1812.00568

[11] S. Nair and C. Finn, "Hierarchical foresight: Self-supervised learning of long-horizon tasks via visual subgoal generation," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020.

[12] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, "Video frame synthesis using deep voxel flow," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4473–4481.

[13] A. X. Lee, R. Zhang, F. Ebert, P. Abbeel, C. Finn, and S. Levine, "Stochastic adversarial video prediction," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[14] F. Ebert, S. Dasari, A. X. Lee, S. Levine, and C. Finn, "Robustness via retrying: Closed-loop robotic manipulation with self-supervised learning," in *Proc. Conf. Robot Learn. (CoRL)*, 2018.

[15] P. Agrawal, A. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to poke by poking: Experiential learning of intuitive physics," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 5074–5082.

[16] F. Paus, T. Huang, and T. Asfour, "Predicting pushing action effects on spatial object relations by learning internal prediction models," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 10584–10590.
[17] F. Li, Q. Jiang, W. Quan, S. Cai, R. Song, and Y. Li, "Manipulation skill acquisition for robotic assembly based on multi-modal information description," *IEEE Access*, vol. 8, pp. 6282–6294, 2020.
[18] D. Liu, Z. Wang, B. Lu, M. Cong, H. Yu, and Q. Zou, "A reinforcement learning-based framework for robot manipulation skill acquisition," *IEEE Access*, vol. 8, pp. 108429–108437, 2020.
[19] K. Fragkiadaki, P. Agrawal, S. Levine, and J. Malik, "Learning visual predictive models of physics for playing billiards," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015.
[20] J. K. Li, D. Hsu, and W. S. Lee, "Push-Net: Deep planar pushing for objects with unknown physical properties," in *Proc. Robot., Sci. Syst.*, 2018, pp. 1–9.
[21] Z. Xu, J. Wu, A. Zeng, J. Tenenbaum, and S. Song, "DensePhysNet: Learning dense physical object representations via multi-step dynamic interactions," in *Proc. Robot., Sci. Syst.*, 2019.
[22] Z. Xu, Z. He, J. Wu, and S. Song, "Learning 3D dynamic scene representations for robot manipulation," in *Proc. Conf. Robot Learn. (CoRL)*, 2020.
[23] N. Watters, D. Zoran, T. Weber, P. Battaglia, R. Pascanu, and A. Tacchetti, "Visual interaction networks: Learning a physics simulator from video," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 4539–4547.
[24] A. E. Tekden, A. Erdem, E. Erdem, T. Asfour, and E. Ugur, "Object and relation centric representations for push effect prediction," 2021, *arXiv:2102.02100*. [Online]. Available: http://arxiv.org/abs/2102.02100
[25] R. Li, A. Jabri, T. Darrell, and P. Agrawal, "Towards practical multi-object manipulation using relational reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 4051–4058.
[26] T. Xue, J. Wu, K. L. Bouman, and W. T. Freeman, "Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 91–99.
[27] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 2786–2793.
[28] K. Fang, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Dynamics learning with cascaded variational inference for multi-step manipulation," in *Proc. Conf. Robot Learn. (CoRL)*, 2019, pp. 42–52.
[29] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," in *Proc. Conf. Robot Learn. (CoRL)*, vol. 87, 2018, pp. 306–316.
[30] J. Wang, S. Lin, C. Hu, Y. Zhu, and L. M. Zhu, "Learning semantic keypoint representations for door opening manipulation," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 6980–6987, Oct. 2020.
[31] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "KPAM: KeyPoint affordances for category-level robotic manipulation," 2019, *arXiv:1903.06684*. [Online]. Available: http://arxiv.org/abs/1903.06684
[32] P. R. Florence, L. Manuelli, and R. Tedrake, "Dense object nets: Learning dense visual object descriptors by and for robotic manipulation," in *Proc. Conf. Robot Learn. (CoRL)*, 2018, pp. 373–385.
[33] P. Florence, L. Manuelli, and R. Tedrake, "Self-supervised correspondence in visuomotor policy learning," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 492–499, Apr. 2020.
[34] Y. Ye, D. Gandhi, A. Gupta, and S. Tulsiani, "Object-centric forward modeling for model predictive control," in *Proc. Conf. Robot Learn. (CoRL)*, 2019, pp. 100–109.
[35] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. (2019). *Detectron2*. [Online]. Available: https://github.com/facebookresearch/detectron2
[36] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*. [Online]. Available: http://arxiv.org/abs/1312.6114
[37] R. Y. Rubinstein, "The cross-entropy method for combinatorial and continuous optimization," *Methodol. Comput. Appl. Probab.*, vol. 1, no. 2, pp. 127–190, Sep. 1999.
[38] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An information-rich 3D model repository," 2015, *arXiv:1512.03012*. [Online]. Available: http://arxiv.org/abs/1512.03012
[39] B. Calli, A. Singh, J. Bruce, A. Walsman, K. Konolige, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Yale-CMU-Berkeley dataset for robotic manipulation research," *Int. J. Robot. Res.*, vol. 36, no. 3, pp. 261–268, 2017.

JIAYU WANG received the B.S. degree in automation from the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China, in 2014. He is currently pursuing the Ph.D. degree in mechanical engineering with the Department of Mechanical Engineering, Tsinghua University, Beijing. His research interests include deep reinforcement learning, visual learning, and robotic manipulation.

CHUXIONG HU (Senior Member, IEEE) received the B.S. and Ph.D. degrees in mechatronic control engineering from Zhejiang University, Hangzhou, China, in 2005 and 2010, respectively. From 2007 to 2008, he was a Visiting Scholar in mechanical engineering with Purdue University, West Lafayette, IN, USA. In 2018, he was a Visiting Scholar in mechanical engineering with the University of California at Berkeley, Berkeley, CA, USA. He is currently an Associate Professor with the State Key Laboratory of Tribology, Department of Mechanical Engineering, Tsinghua University, Beijing, China. His research interests include precision motion control, high-performance multiaxis contouring control, precision mechatronic systems, intelligent learning, adaptive robust control, neural networks, iterative learning control, nonlinear systems, and precision metrology/measurement calibration. He was a recipient of the Best Student Paper Finalist at the 2011 American Control Conference, the 2012 Best Mechatronics Paper Award from the ASME Dynamic Systems and Control Division, the 2013 National 100 Excellent Doctoral Dissertations Nomination Award of China, the 2016 Best Paper in Automation Award, and the 2018 Best Paper in AI Award from the IEEE Internal Conference on Information and Automation. He is currently a Technical Editor for the IEEE/ASME TRANSACTIONS ON MECHATRONICS.

YUNAN WANG is currently pursuing the B.S. degree in mechanical engineering with Tsinghua University, Beijing, China. His research interests include deep learning and robotic manipulation.

YU ZHU (Member, IEEE) received the B.S. degree in radio electronics from Beijing Normal University, Beijing, China, in 1983, and the M.S. degree in computer applications and the Ph.D. degree in mechanical design and theory from the China University of Mining and Technology, Beijing, in 1993 and 2001, respectively.

He is currently a Professor with the State Key Laboratory of Tribology, Department of Mechanical Engineering, Tsinghua University, Beijing. He has published more than 180 technical papers. He is also the holder of more than 100 warranted invention patents. His research interests include precision measurement and motion control, ultra-precision mechanical design and manufacturing, two-photon micro-fabrication, and electronics manufacturing technology and equipment.