Joint Discovery of Object and Action Symbols through Effect Prediction for Robotic Manipulation Planning

Burcu Kilic[1], Berke Kartal[1], Fatih Dogangun[1], Erhan Oztop[2,3], Emre Ugur[1]

***Abstract*—To perform complex manipulation planning, autonomous robots are required to abstract continuous, high-dimensional sensorimotor interactions into discrete object and action representations. Earlier work either categorized objects based on visual appearances, which fails to distinguish objects that appear similar but behave differently, or based on effects under interaction, but was limited to predefined actions. To address these limitations, we propose a model that jointly discovers high-level manipulation primitives and object categories through a binary bottleneck layer, trained to predict multi-modal outcomes, including object motion, contact, and force feedback, from random interaction data. Building on these discovered binary representations, we leverage a discrete planning method that uses intermediate steps in the predicted effect trajectory to enable partial action executions for precise low-level control. Additionally, we evaluate our framework's generalization capabilities on novel objects by assigning object categories through comparing a small number of interaction effects with the predicted effects of learned object symbols, enabling few-shot generalization based on behavior rather than visual similarity. We conduct experiments on tabletop repositioning and stacking tasks, and confirm that our effect-driven planning approach outperforms both a state-of-the-art method and a visual-based alternative in planning precision across seen and novel objects.**

***Index Terms*—Robot learning, Representation learning, Autonomous mental development, Few shot learning**

I. Introduction

Autonomous robot development aims to build intelligent agents capable of learning through own experience, reasoning about their surroundings, and adapting to novel scenarios [1]. However, a fundamental challenge is that reasoning over high-dimensional continuous sensorimotor spaces becomes unmanageable in long-horizon manipulation tasks. Operating in such complex sensorimotor spaces requires a layer of abstraction on both percepts and actions [2], [3]. For example, an infant that stacks multiple blocks on top of each other does not plan low-level joint movements directly from environmental pixels; rather, it thinks in terms of high-level intentions like picking a block and placing it on top of another one.

Prior studies either manually constructed such abstractions [4] or used reconstruction-based clustering [5], ignoring the role of interaction. However, Sun [6] argued that the concepts of objects and skills emerge directly from the agent's interaction with the world and are linked to their goals and needs. Following this principle, an autonomous robot must discover its own representations by continuously exploring and manipulating objects, while learning to make sense of the physical dynamics. For this physical exploration to be effective, the robot's interactions cannot be limited to purely visual observations. Multi-modal feedback such as force and contact can convey complementary information to visual feedback about action outcomes, an observation also supported by developmental studies showing that young infants learn action properties through both visual and tactual interactions with objects [7]. Motivated by this, our approach allows a robot to autonomously categorize objects and learn high-level manipulation skills directly from multi-modal physical interactions.

Through continuous physical interactions, children learn to categorize objects not only by static visual appearance, but primarily by the effects of actions performed on them [8]. This concept is defined by the term *affordances*, first introduced in [9] as the action possibilities the environment provides to an agent relative to its own abilities. In this context, objects are learned not by visual appearance alone but by the actions the agent can perform on them and by the consequences of those actions [10]. Following this, Ugur and Piater [11] proposed a bottom-up approach to generate symbolic abstractions by grouping the similar effects of the robot's own continuous interaction with the objects. Ahmetoglu et al. [12] learned symbolic object affordances through a deep neural network architecture that predicts effects and showed that abstracting based on effects provides better planning performance than reconstruction-based clustering. Kilic et al. [13] extended this method to learn object and parameterized action abstractions jointly. However, learning action primitives based solely on the final outcomes fails to capture the temporal structure of robotic trajectories. For example, if actions are categorized only by their end-effects, an agent cannot distinguish between pushing an object and picking-and-placing it to the exact same spot. This limitation motivates the use of temporally-extended actions and effects to capture the full temporal dynamics of an interaction, which also allows the use of intermediate points of a trajectory for precise planning.

This deep understanding of physical dynamics is especially critical when encountering the unknown. Similar to an infant that applies familiar movements to new toys, an autonomous agent must be able to adapt its learned motor skills to handle novel settings with minimal interactions. Recent studies have utilized foundation models for one-shot object affordance grounding [14] or used geometric similarity to categorize new objects [15]. However, these methods rely only on static appearance-based similarity and fail to capture the physical dynamics of interactions. Furthermore, foundation model-based approaches require computationally expensive training and lack data efficiency. We show that after learning functional

This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) ARDEB 1001 Program (124E227) and by the INVERSE Project under Grant 101136067, funded by the European Union.
[1]Bogazici University, [2]Ozyegin University, [3]Osaka University.

object categories and high-level skills, generalizing to unseen objects based on behavioral similarity by matching multi-modal effects under minimal interactions allows more effective skill transfer than visual matching alone.

In this study, we present a neuro-symbolic framework that enables learning high-level skills and object abstractions jointly from continuous random interaction data in a developmental setting and uses these learned concepts for planning and categorizing novel objects. Our contributions are as follows:

- A deep encoder-decoder network with a binary bottleneck that jointly discovers object and action symbols through a two-stage learning procedure optimized on multi-modal (spatial-haptic) effect trajectory predictions.
- A planning framework that builds a discrete effect library from the learned model and performs search with partial action executions, combined with symbol-conditioned low-level controllers and drift-based replanning for plan executions.
- A few-shot object generalization method, where novel objects are categorized according to behavioral similarity through effect matching, enabling manipulation planning on unseen objects.

We evaluate the proposed framework in a tabletop environment on manipulation tasks. Our results demonstrate that the proposed method, which uses the discovered symbolic actions, achieves significantly better planning performance compared to the Diffusion Policy [16] baseline across all object types in both object repositioning and stacking tasks. We further conduct ablation studies to validate our design choices (a transformer-based action encoder and two-stage learning) and to analyze the impact of the action symbol dimension on planning performance. Moreover, we show that our effect-driven object categorization generalizes better to novel objects with a few demonstrations, outperforming the baselines that rely on visual conditioning of object depth maps.

II. Related Work

It is necessary to discover object categories and high-level actions in order to perform discrete planning. [5], [17], [18] used an autoencoder with a binary bottleneck trained on the reconstruction of states, allowing symbolic plan generation. [12], [19], [20] abstracts object affordances by an effect predictor encoder-decoder deep neural network with a binary bottleneck layer; however, in these studies, predefined actions were used, and symbolic action discovery was not addressed. [13] built upon a similar architecture and jointly discretized both actions and objects by spatial effect prediction in a curiosity-driven exploration setting. We perform a similar joint discretization that aims to obtain high-level skills and object categories, but we leverage partial action executions by predicting spatio-tactual effect trajectories and demonstrate that our framework can be beneficial in few-shot object generalization.

Learning the dynamics of the environment by effect prediction has been studied with different architectures. [21] used Graph Neural Networks to model the effects of push actions on multi-part objects. [22] realized a model for single-action push and grasp effect prediction, and incorporated partial effects into their planning module. [23] integrated parameterized skills and learned continuous skill-effect models. [24] proposed a model-based planner over the learned latent action space. [25] learned multi-object affordances by effect prediction and utilized the network for multi-step planning; however, only a given place primitive was used as a skill. In addition to that, all of these architectures require model inference during planning, which is computationally heavy, whereas we create a discrete effect library to utilize in the planning.

Recently, generalization to novel objects by one- or few-shot demonstrations has been explored. [15] categorized object affordances based on geometric similarity and generalized to novel object categories. [26] proposed a neuro-symbolic imitation learning framework; however did not explore autonomous discovery of skills and required a human labeler. [27] realized a network to generate affordance maps for objects and performed active interactions to adapt a prior affordance map of a novel object. [14] used foundation models for object-to-object affordance grounding, which is useful in one-shot object generalization, and leverages LLMs for creating constraint functions during planning. Most of these methods are effective at identifying geometric similarity in novel objects, but we move beyond the visual affordances. We categorize objects based on their effects on the same actions; thus, we can differentiate objects that appear visually similar, but behave differently to the same applied action trajectory.

III. Method

We propose a framework that autonomously discovers symbolic representations of objects and actions through predicting the effect of interactions, and utilizes these symbols for multi-step manipulation planning and few-shot generalization to novel objects. In this section, we first formulate the problem, then present the overview of the proposed framework, and detail its components in the subsequent sections.

A. Problem Definition

We consider a robotic manipulation setting in which an agent interacts with objects on a tabletop environment and learns high-level object and action representations that enable multi-step discrete planning from its own exploration. In this section, we provide object, action, and effect formulations, and the objective for symbol discovery and planning.

1) Object Representation: An object $o \in \mathbb{R}^{W \times H}$ is represented by its depth image with width $W = 64$ and height $H = 64$.

2) Action Representation: An action $a \in \mathbb{R}^{T \times D}$ is a trajectory of robot's joints' angular positions over T timesteps, where D is the degree of freedom of the robot:

$$a = \{\theta_1, \theta_2, \theta_3, ..., \theta_T\}, \quad \theta_t \in \mathbb{R}^D. \tag{1}$$

3) Effect Representation: At each timestep $t \in \{1, 2, ..., T\}$ during the action execution, the following are recorded:

- The absolute position of the object, $\mathbf{p}_t \in \mathbb{R}^3$.
- The force vector $\mathbf{f}_t \in \mathbb{R}^3$ measured from the force sensor on the robot's wrist.

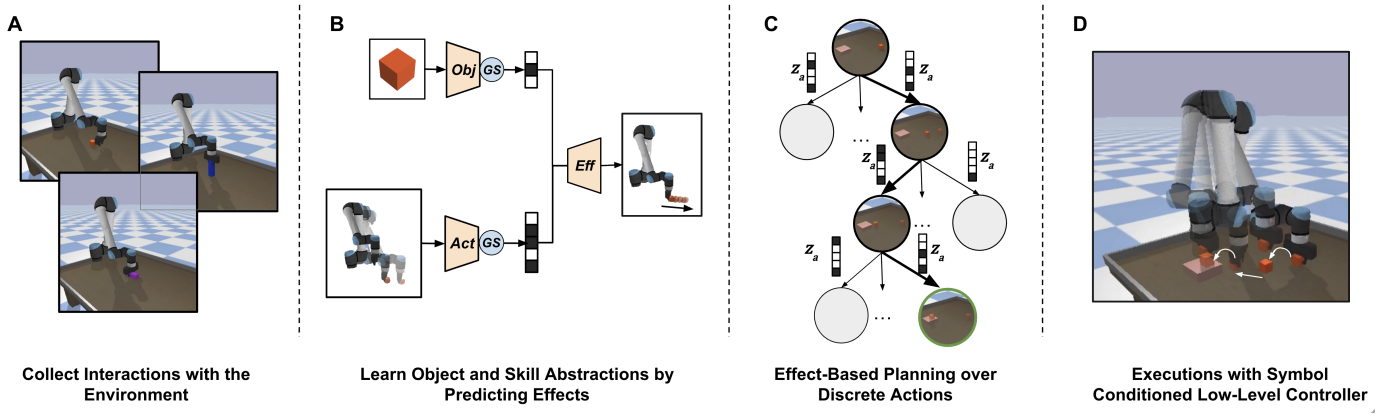

Fig. 1. **Overview of the proposed framework. (A)** The agent explores the environment by interacting with objects, collecting depth images, joint angle trajectories, and multi-modal effect trajectories containing the position of the objects, force vectors, and contact feedback. **(B)** An encoder-decoder network, including object and action encoders with a Gumbel-Sigmoid (GS) activation function, which maps objects and actions to discrete symbols by learning to predict effect trajectories. **(C)** A discrete effect library is built from the discovered symbols, and an A* search over this effect library produces multi-step plans with partial action executions. **(D)** The planned symbolic actions are converted into executable joint-angle trajectories using a symbol-conditioned low-level controller.

- The contact feedback $c_t \in \mathbb{R}$ obtained from inside the gripper of the robot.

The effect trajectory is defined as the change in these quantities relative to the initial timestep:

$$e = \{(\mathbf{p}_t - \mathbf{p}_0, \mathbf{f}_t - \mathbf{f}_0, c_t - c_0)\}_{t=1,\dots,T} \tag{2}$$

4) Symbol Discovery: The aim is to learn mappings $\phi_o : \mathbb{R}^{W\times H} \rightarrow \{0,1\}^{s_o}$, and $\phi_a : \mathbb{R}^{T\times D} \rightarrow \{0,1\}^{s_a}$ where s_o and s_a are the dimension of the object and action symbol, respectively; such that the learned discrete object and action symbols are instrumental about the effects, allowing discrete planning.

5) Planning Problem: The aim is to find a symbolic action sequence π that, when applied to a given object at position p_0, navigates it to the goal position p_{goal}. For the tabletop task, we measure planning performance by using Euclidean distance between the final position of the object and the goal $\|\mathbf{p}_{\text{final}} - \mathbf{p}_{\text{goal}}\|$. For the stacking task, we use a binary success criterion for whether the object is placed on top of a target object.

B. Method Overview

An overview of the proposed framework is illustrated in Fig. 1. The interaction data is collected through the robot's random exploration, including joint angle trajectories, depth images of objects, and multi-modal effect trajectories consisting of the position of the object, force, and contact feedback (Fig. 1-A). An encoder-decoder network is then trained to predict effect trajectories, yielding discrete action and object symbols in its latent space (Fig. 1-B). Using the discovered symbols, a discrete effect library is constructed over which A* search is performed to produce multi-step plans (Fig. 1-C). Then, the symbolic actions in the plan are converted into joint-angle trajectories using symbol-conditioned low-level controllers (Fig. 1-D). Once trained, the learned model and the effect library can additionally be leveraged for few-shot generalization to novel objects by matching their observed effects to the learned object symbols from only a few demonstrations.

C. Discretization of Objects and Actions by Staged Learning of Effects

For discrete planning, we need discrete object and action representations that are predictive of effects. Reconstruction-based methods or unsupervised clustering can produce discrete categories. Still, they group by similarity in the input space, leaving the planner without a reliable link between symbols and outcomes. Therefore, object and action discretization must be learned jointly with effect prediction, since effects emerge only from the interaction of the two. We therefore propose a deep neural network that simultaneously predicts effect trajectories and produces symbolic abstractions through a binary bottleneck layer, which is shown in Figure 2.

The object encoder ϕ_o, a convolutional neural network with a binary activation function, maps the given object depth image to a binary symbol z_o. The action encoder ϕ_a, with a binary activation function, takes the joint angle trajectory over T timesteps and encodes it to a binary representation z_a. Since action trajectories carry temporal structure in the form of sequential joint angles, we employ the Transformer architecture [28] as the action encoder. Both encoders use Gumbel-Sigmoid (GS) [29], [30] as the binary activation function with temperature $Tp = 1$:

$$GS(x) = \sigma(\frac{x+g}{Tp}), g = log(log(u_2) - log(u_1) + \epsilon) \tag{3}$$

Here, the output is rounded to obtain binary symbols, and the reparameterization trick [31] is used to pass gradients through both encoders during training. The binary symbols for object and action are concatenated as $r = (z_o||z_a)$, and the decoder ψ predicts the effect trajectory $\hat{e}$ given this vector r.

We adopt a two-stage training procedure motivated by the finding that coarse action categories can be distinguished mainly from force and contact modalities, while directional action categories require spatial information. In the first stage,

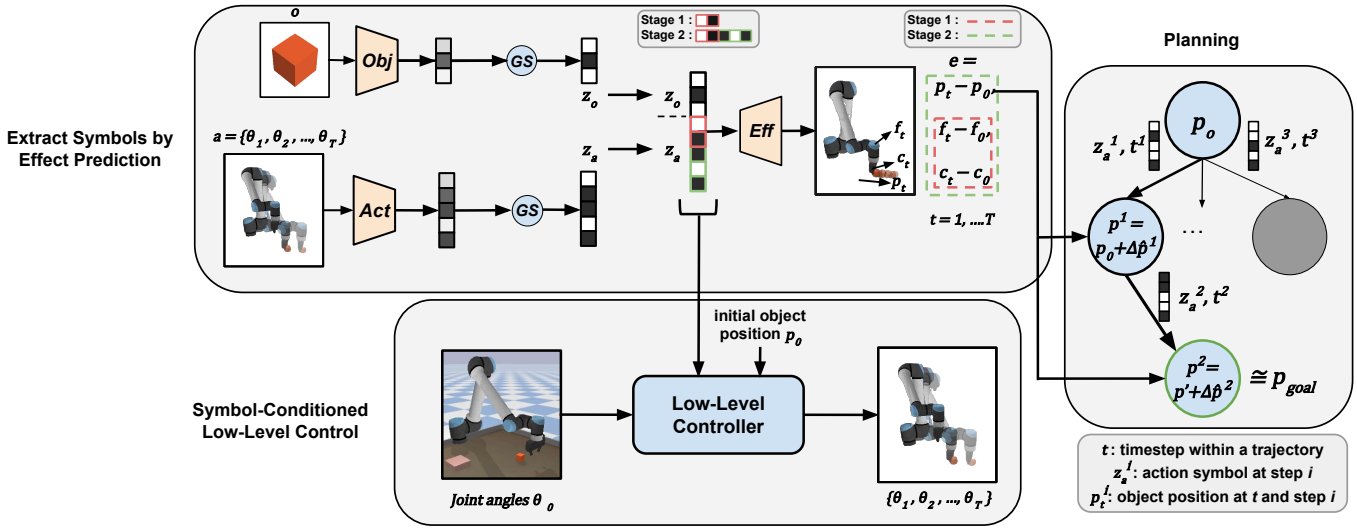

Fig. 2. **Learning, planning, and execution pipeline of the proposed method.** Firstly, the effect prediction network takes joint angle trajectories and object depth maps from the interaction data, and produces discrete vectors by applying Gumbel-Sigmoid (GS) activation over the encoded inputs. In stage 1, the network predicts effect trajectories of force and contact feedback from a limited set of action symbol bits. In stage 2, the model uses all bits to predict all effects, including positional changes. After training is complete, object z_o and action symbols z_a are extracted. The low-level controller (CNMP) learns joint angle trajectories, conditioned on these object and action symbols. Lastly, a tree search planning is employed to reach a goal object position p_{goal} given an initial position p_0. The planning is conducted on a discrete effect library that includes the predicted effects mappings from each tuple of object symbol, action symbol, and timestep (z_o, z_a, t).

the model predicts only the force and contact feedback effects, using the full object symbol bits and fewer action symbol bits. This stage learns to differentiate high-level action primitives (e.g., distinguishing push from pick-and-place). In the second stage, the model uses all symbol bits to predict the entire effect trajectory, including position, force, and contact feedback changes. This stage refines the action symbols to capture directional variations within each primitive (e.g., left push, right push, back pick-and-place).This procedure is also backed up by the developmental findings that infants first acquire coarse sensorimotor skills through tactile feedback, and later refine them into directional variants as visuomotor abilities improve [7], [32]. We find that this two-stage procedure yields empirically more stable training compared to learning all action categories simultaneously. The model is trained with Mean Squared Error (MSE):

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T} ||\hat{e}_t - e_t||^2 \tag{4}$$

After training, the model produces mappings for both objects $\phi_o(o) \rightarrow z_o$ and action trajectories $\phi_a(a) \rightarrow z_a$. To execute the learned symbolic actions on the robot, we need to invert the action mapping to generate continuous low-level trajectories from action symbols.

D. Symbol-Conditioned Low-Level Controller

High-level action symbols enable efficient planning but cannot be directly executed on the robot. A low-level controller is required to generate executable joint angle trajectories conditioned on the discovered symbols. We use Conditional Neural Movement Primitives (CNMP) [33] for this purpose, for two reasons. First, CNMP is a Learning from Demonstrations framework, so it can leverage the same interaction data used for training the effect prediction network without requiring additional trajectory collection. Second, CNMP can be conditioned on task parameters, allowing the learned symbols to guide trajectory generation. A task or a demonstration is defined by the action to apply (z_a), the object it is applied to (z_o), and the object's initial position (o_x, o_y, o_z). Therefore, the task parameter is constructed as:

$$\gamma = [z_a, z_o, (o_x, o_y, o_z)] \tag{5}$$

For each demonstration, the task parameter γ is set using the symbols extracted from the trained effect-prediction model, along with the initial object location.

During training, CNMP predicts target joint angles θ_{target} from given observations $(t_{obs}, \theta_{obs}, \gamma)$, where both target t_{target} and observation points t_{obs} are uniformly sampled within a demonstration. In the forward pass, the CNMP encoder takes observations as input. A latent representation L is obtained by applying average pooling over the encoder outputs. Then, a target timestep t_{target} is concatenated with L and queried to the decoder to predict the joint angles θ_{target} at that timestep. At inference time, a trajectory is generated by feeding initial joint angles with the task parameter to the encoder and querying the decoder with each target timestep. In this way, high-level action z_a and object symbols z_o are transformed into low-level actions that are executable by the manipulator robot. Please refer to [33] for more details.

E. Planning with Partial Action Executions

We utilize a search-based algorithm over discrete effects for planning. To this end, we first create an effect library

that includes the positional changes exerted by the learned actions for each given object. We pass the object's depth image through the trained object encoder ϕ_o to find its symbol z_o. Then, for each discrete action symbol $z_a \in \{0,1\}^{s_a}$, we concatenate z_a and z_o and pass the resulting representation through the trained decoder ψ to predict the effect trajectories $\hat{e}$. Since position changes are sufficient to perform spatial planning, we discard the contact and force feedback in the predicted effect, retaining only the position component of the effect $\hat{e}_{pos}$. Predicting the entire effect trajectory rather than only the final effect allows us to plan with partial action executions, where each intermediate timestep provides a distinct reachable state.

Therefore, our constructed effect library consists of mappings from each combination of object symbol, action symbol, and timestep z_o, z_a, t to a position effect $\hat{e}^t_{pos}$. Using this library, we can perform A* search over the space of discrete effects. Each node in the search tree corresponds to a 3D position of the object, with the root node corresponding to the initial position of the object. The edges represent transitions defined by the discrete effect predictions in the library. The A* search algorithm uses the Euclidean distance as the heuristic, and is bounded by maximum depth d_{max}. The search algorithm returns a plan when a state within an error margin $epsilon$ of p_{goal} is reached. The resulting plan is a sequence of action symbols and timesteps: $\pi = \{(z^1_a, t^1), \ldots, (z^d_a, t^d)\}$ where d is the length of a found plan, smaller than d_{max}.

To execute the generated discrete plans, we sequentially query the trained low-level controller for each step k, conditioned on z_o and the k-th action symbol z^k_a. The resulting joint angle trajectory is truncated at the planned timestep t^k to enable the partial execution.

To ensure the integrity of the execution, we implement a drift validation between steps. Before executing the step k, we compare the object's current position with the position predicted from the effect library. If the Euclidean distance between these positions exceeds a drift threshold δ, we trigger a replanning with the same algorithm from the current position with remaining depth budget $d_{max} - k$.

F. Few-Shot Object Generalization

To generalize novel objects, we categorize them based on behavioral similarity rather than visual appearance. We collect a small number of random demonstrations of a novel object, and use the trained action encoder ϕ_a to extract the action symbol z_a for each demonstration. We then predict effects for each available object symbol z_o combined with extracted action symbol, and compute the error between the predicted and observed effects. We finally assign the object symbol corresponding to the lowest mean prediction error across the demonstrations to the novel object, which enables planning on novel objects utilizing the existing effect library without retraining or large-scale data collection.

IV. Experiments

In this section, we first explain our experimental setup, including the data collection procedure and training details.

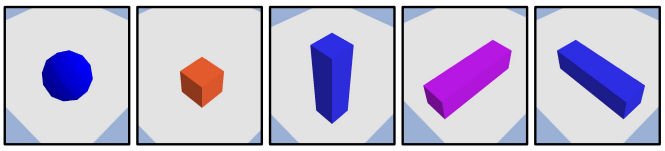

Fig. 3. Objects used during the data collection. From left to right: Ball, Cube, Block Z, Block X, and Block Y.

We then show the experiments we conducted to analyze the following:

- Are the learned actions sufficiently diverse and precise for use in multi-step planning? (Sec. IV-B)
- Does our transformer-based network with two-stage learning achieve high effect prediction accuracy? (Sec. IV-C)
- Does our framework outperform diffusion-based baseline across two different object manipulation tasks? (Sec. IV-D)
- Is an effect-based approach stronger in few-shot object generalization than spatial approaches? (Sec. IV-E)

A. Experimental Setup

1) Data Collection: The experiments are conducted in PyBullet [34] tabletop simulation environment with a 6-DOF UR10e manipulator robot. Five different types of objects containing ball, cube, and long blocks with three different orientations, as shown in Fig. 3, are used for data collection. We name these long blocks Block X, Block Y, and Block Z by their alignment axis (e.g., Block Z is vertical). The data are collected through a random exploration, illustrated at Fig. 4. An interaction starts with the robot approaching the object

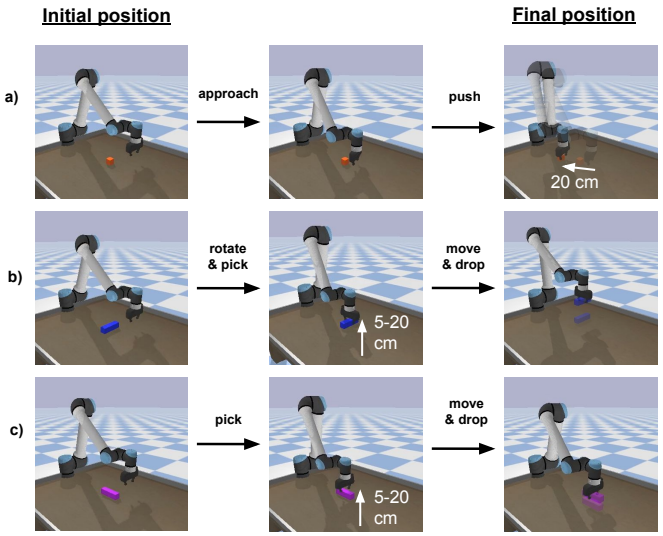

Fig. 4. Data collection via random exploration. **(a) Push Action**: approach from a random angle and push the object. **(b) Pick & Place (Y-aligned)**: approach while rotating the gripper, then grasp and lift the object. At last, move the object and drop it. **(c) Pick & Place (X-aligned)**: same as (b), but no gripper rotation is applied. Lift heights are uniformly sampled from 5-20 cm. Each move or push action moves the object 20 cm in a randomized direction.

from a uniformly sampled direction. The robot then randomly executes either a push or a pick-and-place primitive on the object, with x-aligned or y-aligned grasp orientation. In the pushing primitive, the object is displaced by $20cm$ along a uniformly sampled direction on the table. In the pick-and-place primitive, the robot holds the object, lifts it to a uniformly sampled height $h \sim U[5, 20]cm$, and places it at a randomly sampled target location 20 cm away from the initial position. The two grasp orientations during the pick-and-place primitive enable the grasping of objects with different geometries. For example, a block wide along the X axis needs the X-aligned grasp to be lifted, but a cube works with both. The different grasps are also reflected in the force sensor readings recorded during interactions, allowing the effect-prediction model to differentiate objects based on their grasp affordances. We collected 1000 exploration data samples per object (5000 in total).

2) Training: The effect trajectory prediction model is trained on 5000 interaction samples consisting of joint angle trajectories, object depth maps, and effect trajectories. The model is trained for 600 epochs with a learning rate of 1×10^{-4} using Adam optimizer [35]. The object symbol dimension is set to $s_o = 3$, and the action symbol dimension is set to $s_a = 5$ in the first training stage, allowing $2^5 = 32$ distinct actions to be learned.

After training the effect trajectory prediction model, we extract binary object and action symbols from the bottleneck layer using hard-rounded Gumbel-Sigmoid for both object and action encoders across all samples in the dataset. We then form the training data for the low-level controller using learned symbols along with the object initial positions and joint angle trajectories. We train the low-level controller CNMP, conditioned on object and action symbols and the object's initial position, to predict joint angle trajectories.

3) Evaluated Planning Methods: We compare the following methods in the planning experiments:

- **Effect-Driven Planning Framework (Ours)**: We use the effect-trajectory prediction network to extract action and object symbols and construct a discrete effect library. We employ A* search over the effect library, with partial action executions, to find plans, and then execute the symbolic actions of the plan using the trained CNMP as a low-level controller.
- **Diffusion Policy**: We train a Diffusion Policy conditioned on the goal position and the object's depth map. Since multi-step planning in our framework uses at most 4 action steps, we execute the Diffusion Policy at most 4 times sequentially for each test, and we condition each execution on the current position of the object. We consider an adaptive execution count for each test that is proportional to the distance of the given goal position.

B. Generated Symbols

Fig. 5 shows an example of the learned effect trajectories for the cube object, illustrating that the discovered action symbols capture a diverse set of manipulation primitives. The top-left projection illustrates the directional variety among the discovered primitives, while the bottom-row plots show the separation between push and pick-and-place primitives, where the pick-and-place primitives have a nonzero Z component indicating they involve lifting. The 3D plot on the top-right shows that the primitives corresponding to discovered action symbols cover the effect space with distinct directions and heights. This diverse and distributed set of action primitives is essential for planning, as they enable the planner to achieve precision in reaching arbitrary goal positions through multi-step planning.

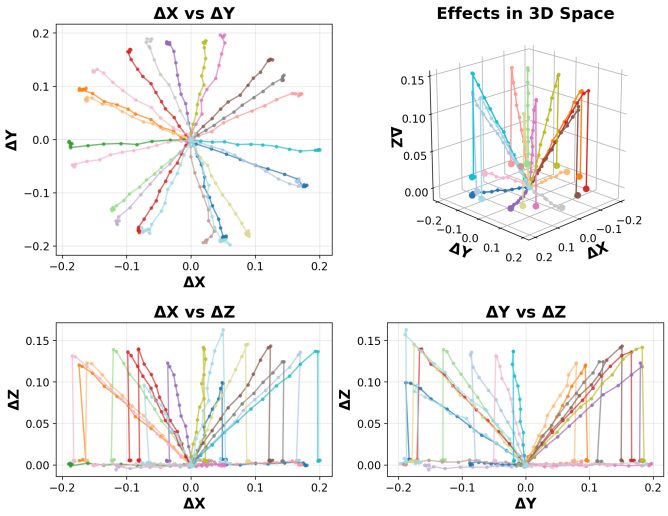

Fig. 5. Visualization of the learned effect trajectories for the cube object. Each colored trajectory represents the predicted positional effect of a distinct learned action symbol. The 2D projections include all discovered actions, while some actions corresponding to less frequently encountered symbols are discarded from the 3D plot (on the top-right) for clarity.

C. Effect Prediction Error

In this experiment, we evaluate the effect prediction performance of the proposed transformer-based action encoder with a two-stage learning procedure, compared to alternative designs such as single-stage learning and a fully connected action encoder. All models are trained to predict an effect trajectory $\hat{e}_t$ representing changes in object position, contact feedback, and force vectors at each timestep. We compute the mean absolute error between the predicted and ground truth positions of the object in effect trajectories on a test set of size $N = 1000$:

$$Error = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}||\hat{\mathbf{e}}_t^{pos} - \mathbf{e}_t^{pos}||_1 \tag{6}$$

The results are averaged over 5 runs with different random seeds. As shown in Table I, our proposed architecture, which combines two-stage learning with a transformer-based action encoder, achieves significantly lower prediction error than both the single-stage training variant (paired t-test, $p < 0.01$) and the fully-connected action encoder baseline (paired t-test, $p < 0.01$), validating our design choices.

TABLE I
MEAN POSITION EFFECT PREDICTION ERRORS AVERAGED OVER 5 RUNS.

Model	Error (cm)
Effect Traj. Prediction Network (Ours)	**1.70 ± 0.036**
Effect Traj. Prediction Network (Single Stage)	1.77 ± 0.041
Effect Traj. Prediction Network (Fully Connected)	1.82 ± 0.058

D. Planning Performance

In this experiment, we assess the planning performance of our method and the Diffusion Policy baseline on two tasks. Task 1 is a tabletop task with no obstacles, in which the robot must move the object to a goal position within a larger area of the table. Task 2 is a stacking task where a height-varying block object is placed on the table, and the robot must stack the manipulated object on top of it. In general, the goal is not reachable with a single primitive execution, requiring multi-step planning with partial executions. Example planning executions of our approach can be seen in Fig. 6. To evaluate the performance in Task 1, we use the mean Euclidean distance error between the final object position and the goal with paired t-tests to assess statistical significance. For Task 2, we report the success rates and obtain significance values from McNemar's test [36]. We run each experiment 3 times and use 100 planning tests per task and object.

1) Action Symbol Dimension Ablation: We first performed an ablation study on the dimension of the action symbol to investigate its impact on planning performance. We train our entire pipeline with action symbol dimensions of 4, 5, and 6 bits and compute the mean planning errors for each configuration. As shown in Fig. 7, the 4-bit configuration demonstrates poor performance compared to others due to limited planning resolution with only 16 distinct action possibilities. The 6-bit configuration achieves slightly lower planning errors than the 5-bit model. However, the planning time for the 6-bit configuration is higher since the branching factor in the search tree doubles with each additional bit. Therefore, we chose the 5-bit configuration for all subsequent experiments, as it provides a favorable trade-off between planning performance and computational cost.

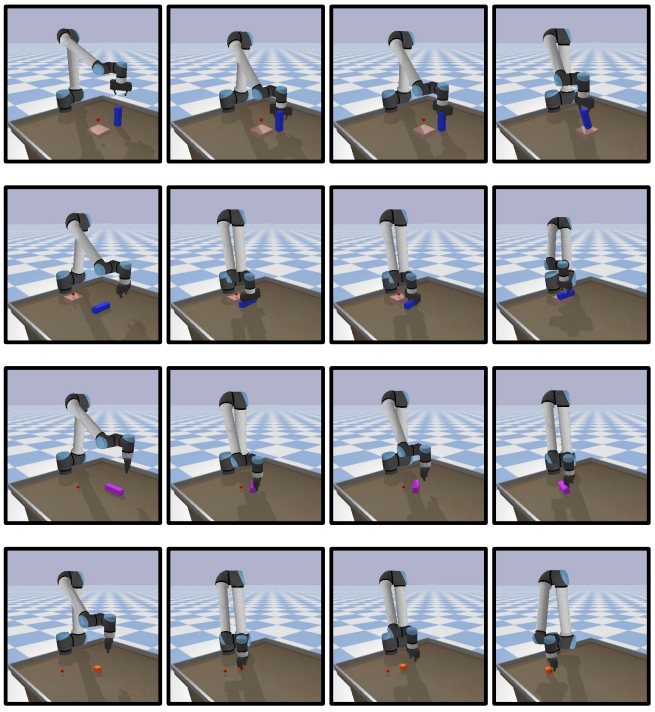

Fig. 6. Planning examples of our approach for Block Z, Block Y, Block X, and Cube objects (from top to bottom). The red dots show the goal position in each case. The top two rows belong to Task 2, and the bottom two rows show Task 1.

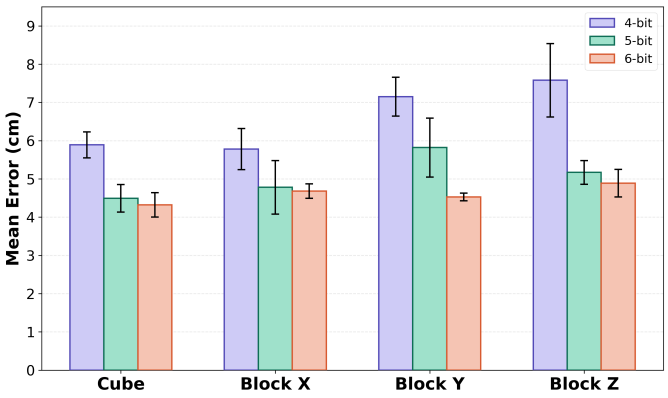

Fig. 7. Mean planning errors (in cm) for Task 1 across different action symbol dimensions (4-bit, 5-bit, and 6-bit) for each object that is in the training set. The results are averaged over 3 runs, with 100 planning tests per object. Error bars refer to standard deviations across runs.

2) Planning Performance Comparison with Baseline: We evaluate both methods on cube, block X, block Y, and block Z objects for both Task 1 and Task 2. As reported in Table II, our proposed method achieves significantly lower planning errors across all objects in Task 1 (paired t-test, $p < 0.001$), and higher success rates in Task 2 (McNemar's test, $p < 0.001$). The performance of our method stems from the precision of search-based planning compared to goal-conditioned reactive policy execution. Furthermore, since our model learns the effects of actions at intermediate timesteps, it can generate new midway points that were not explicitly demonstrated in the training data. This capability provides superior in-distribution generalization compared to Diffusion Policy. The

TABLE II
PLANNING RESULTS ACROSS SEEN OBJECTS FOR BOTH TASKS 1 & 2

Object	Diffusion Policy	Ours
Task 1 (Error in cm)		
Cube	5.09 ± 0.11	**4.49 ± 0.36**
Block X	6.50 ± 0.69	**4.78 ± 0.70**
Block Y	7.99 ± 0.45	**5.82 ± 0.77**
Block Z	7.05 ± 1.56	**5.17 ± 0.31**
Task 2 (Success Rate %)		
Cube	37.67 ± 3.3	**55.33 ± 4.03**
Block X	17.33 ± 3.3	**39.67 ± 5.44**
Block Y	19.67 ± 7.72	**45.0 ± 2.16**
Block Z	18.33 ± 2.87	**69.67 ± 5.56**

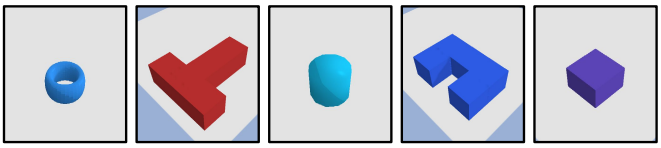

Fig. 8. Novel objects for few-shot generalization. From left to right: Torus, T-Block, Cylinder, U-Block, and Wide Block.

performance gap is specifically notable in Task 2 as stacking strictly requires correct grasping of different objects, aligning with their grasp-affordances. For example, the direction of grasping differentiates Block X and Block Y as they require grasping from the narrower dimensions, and the grasping point differentiates Block Z, as it requires grasping from a higher point. The Diffusion Policy baseline failed to consistently match the grasp affordances of such objects, colliding with Block X and Block Y and overthrowing Block Z due to incorrect grasping. As a result, it performs poorly on this task, which requires precise execution of actions. On the other hand, our framework is preferable since it constructs object and action categories based on their effect trajectory, incentivizing representations that are influenced by both push and grasp actions in various directions.

E. Few-Shot Object Generalization

In our framework, we categorize objects based on similar effect-driven affordances rather than their visual appearances for planning. In this experiment, we introduce 5 new objects, including Torus, T-Block, Cylinder, U-Block, and Wide Block, to evaluate the impact of effect-driven categorization on generalization to novel objects. As shown in Fig. 8, the novel objects have shapes or sizes that differ from those of the existing objects in the dataset.

Using our data collection procedure described in Section IV-A1, we collect only three random demonstrations per novel object, and we then assign each an object symbol based on the lowest mean effect prediction error across the demonstrations, as explained in Section III-F. We create 100 different planning tests for each novel object and task, and perform planning using the assigned object symbol in our proposed method. We compare our framework against the Diffusion Policy baseline, which is conditioned on the depth maps of the novel objects. As shown in Table III, our method significantly outperforms the Diffusion Policy baseline across all objects in both Task 1 (paired t-test, $p < 0.001$) and Task 2 (McNemar's test, $p < 0.001$). Figure 9 illustrates a stacking action with the novel T-Block object, where the Diffusion Policy fails to grasp the object properly, while our method successfully picks and places it on top of the target.

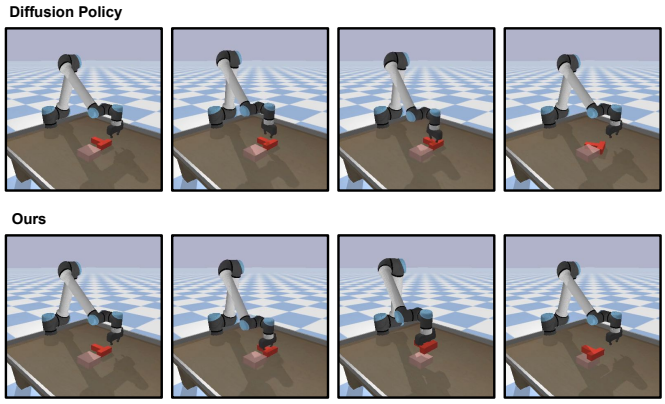

Fig. 9. Example executions for stacking planning task including novel T-Block object, where Diffusion Policy (top) fails, while our proposed method (bottom) successfully performs the action.

To further analyze the benefit of effect-based object categorization, we also modify our method such that the depth image of the novel object is directly passed through the trained object encoder, and the resulting symbol is used for planning without any effect-matching categorization. The encoder categorizes Torus as Ball, and T-Block as Block Y, while our effect-based categorization assigns Torus to Cube, and T-Block to Block X. These differences represent the distinctions in behavior of the objects that visual similarity fails to capture, such as Torus, which appears like a Ball but does not roll freely when push actions are applied, and T-Block is visually similar to both Block X and Block Y, but its grasp affordance aligns with Block X. As shown in Figure 10, our effect-based

TABLE III
PLANNING RESULTS ACROSS UNSEEN OBJECTS FOR BOTH TASKS 1 & 2

Object	Diffusion Policy	Ours
Task 1 (Error in cm)		
Wide Block	5.28 ± 0.39	**4.83 ± 0.11**
Torus	5.57 ± 0.13	**4.45 ± 0.26**
U-Block	6.32 ± 0.84	**4.76 ± 0.42**
T-Block	6.84 ± 0.30	**5.87 ± 0.33**
Cylinder	6.96 ± 0.86	**5.02 ± 0.42**
Task 2 (Success Rate %)		
Wide Block	38.67 ± 3.77	**44.33 ± 5.25**
Torus	53.67 ± 3.30	**56.33 ± 1.25**
U-Block	27.0 ± 9.09	**45.00 ± 4.24**
T-Block	16.0 ± 0.82	**31.33 ± 5.25**
Cylinder	41.67 ± 6.13	**43.00 ± 2.94**

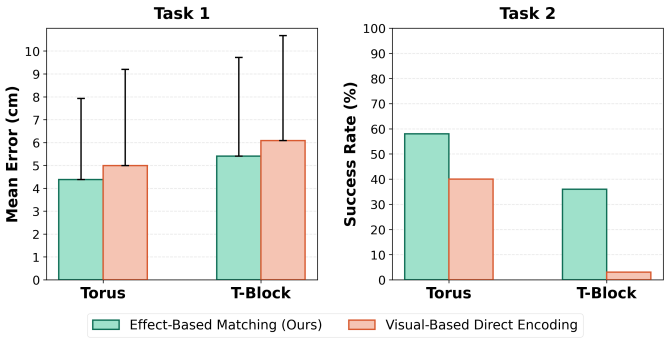

Fig. 10. Comparison of effect-based matching (our proposed method) and visual-based encoding for novel object categorization on Torus and T-Block objects in both tasks, with 100 planning tests for each. For a visual-based encoding baseline, the symbol for the novel object is obtained directly from the object encoder without effect matching.

categorization yields lower planning errors for Task 1 (paired t-test, $p = 0.05$), and significantly higher success rates for Task 2 (McNemar's test, $p < 0.001$) compared to visual-based alternative method. These results support the idea that object categorization based on behavioral effects, rather than image or geometric shapes, leads to more robust object generalization.

V. Conclusion

In this study, we propose a neuro-symbolic framework that discovers object and action symbols jointly from interaction data through multi-modal effect prediction and utilizes these learned symbols for multi-step planning on manipulation tasks. We utilize an encoder-decoder architecture with a binary bottleneck layer and a two-stage learning procedure, where the initial stage learns high-level action primitives from force and contact feedback, and the latter stage uses spatial effects to differentiate directional variations of primitives. In the experiments, we first validate our choices for the action encoder architecture and the two-stage learning approach. We then evaluate the planning performance of our method, which performs A* search over a discrete effect library with partial action executions and uses a symbol-conditioned low-level controller, outperforming the Diffusion Policy baseline significantly across all object types and manipulation tasks. Furthermore, we show that our effect-based object categorization enables robust object generalization with only a few demonstrations, outperforming a baseline that relies on visual appearance-based conditioning. Overall, we provide an approach to robotic manipulation planning that leverages symbolic abstractions derived from the multimodal effects of actions, enabling precise multi-step planning and efficient transfer for object generalization.

Our framework has several limitations that contain potential future research opportunities. Firstly, we focused on learning various unary single-object affordances and conducted multi-step planning for one object at a time. In the future, the discovery of inter-object relational symbols can be integrated into our system for both learning in complex environments and performing multi-object planning. In addition, the drift-based replanning mechanism we employ can trigger replanning only after an action is executed. This mechanism can be enhanced by incorporating a verification module that checks the feasibility of actions before execution. Lastly, we currently use the A* algorithm for planning, which lacks the speed and memory efficiency compared to the Planning Domain Definition Language (PDDL) [37] solvers. In the future, our system can be developed to produce symbolic operators compatible with PDDL, enabling faster and more scalable symbolic planning.

References

[1] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida, "Cognitive developmental robotics: A survey," *IEEE transactions on autonomous mental development*, vol. 1, no. 1, pp. 12–34, 2009.

[2] G. Konidaris, "On the necessity of abstraction," *Current opinion in behavioral sciences*, vol. 29, pp. 1–7, 2019.

[3] G. Konidaris, L. P. Kaelbling, and T. Lozano-Perez, "From skills to symbols: Learning symbolic representations for abstract high-level planning," *Journal of Artificial Intelligence Research*, vol. 61, pp. 215–289, 2018.

[4] E. Ugur, A. Ahmetoglu, Y. Nagai, T. Taniguchi, M. Saveriano, and E. Oztop, "Neuro-symbolic robotics," 2025, http://dx.doi.org/10.13140/RG.2.2.25854.09283.

[5] M. Asai and A. Fukunaga, "Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary," in *AAAI*, vol. 32, no. 1, 2018.

[6] R. Sun, "Symbol grounding: a new look at an old idea," *Philosophical Psychology*, vol. 13, no. 2, pp. 149–172, 2000.

[7] E. J. Gibson and A. S. Walker, "Development of knowledge of visual-tactual affordances of substance," *Child Development*, vol. 55, no. 2, pp. 453–460, 1984.

[8] L. B. Smith, "Action alters shape categories," *Cognitive Science*, vol. 29, no. 4, pp. 665–679, 2005.

[9] J. J. Gibson, "The theory of affordances," in *Perceiving, Acting, and Knowing: Toward an Ecological Psychology*, R. E. Shaw and J. Bransford, Eds. Hillsdale, NJ: Lawrence Erlbaum Associates, 1977, pp. 67–82.

[10] L. Jamone, E. Ugur, A. Cangelosi, L. Fadiga, A. Bernardino, J. Piater, and J. Santos-Victor, "Affordances in psychology, neuroscience, and robotics: A survey," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 1, pp. 4–25, 2018.

[11] E. Ugur and J. Piater, "Bottom-up learning of object categories, action effects and logical rules: From continuous manipulative exploration to symbolic planning," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2627–2633.

[12] A. Ahmetoglu, M. Y. Seker, J. Piater, E. Oztop, and E. Ugur, "Deepsym: Deep symbol generation and rule learning for planning from unsupervised robot interaction," *Journal of Artificial Intelligence Research*, vol. 75, pp. 709–745, 2022.

[13] B. Kilic, A. Ahmetoglu, and E. Ugur, "Predictability-based curiosity-guided action symbol discovery," in *2025 IEEE International Conference on Development and Learning (ICDL)*, 2025, pp. 1–6.

[14] T. Tian, X. Kang, and Y.-L. Kuo, "O^3Afford: One-shot 3d object-to-object affordance grounding for generalizable robotic manipulation," in *9th Annual Conference on Robot Learning*, 2025.

[15] C. Ning, R. Wu, H. Lu, K. Mo, and H. Dong, "Where2explore: Few-shot affordance learning for unseen novel categories of articulated objects," *Advances in Neural Information Processing Systems*, vol. 36, pp. 4585–4596, 2023.

[16] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, 2024.

[17] M. Asai and C. Muise, "Learning neural-symbolic descriptive planning models via cube-space priors: the voyage home (to strips)," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, ser. IJCAI'20, 2021.

[18] M. Asai, H. Kajino, A. Fukunaga, and C. Muise, "Classical planning in deep latent space," *Journal of Artificial Intelligence Research*, vol. 74, pp. 1599–1686, 2022.

[19] A. Ahmetoglu, B. Celik, E. Oztop, and E. Ugur, "Discovering predictive relational object symbols with symbolic attentive layers," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1977–1984, 2024.

[20] A. Ahmetoglu, E. Oztop, and E. Ugur, "Symbolic manipulation planning with discovered object and relational predicates," *IEEE Robotics and Automation Letters*, 2025.

[21] A. E. Tekden, A. Erdem, E. Erdem, T. Asfour, and E. Ugur, "Object and relation centric representations for push effect prediction," *Robotics and Autonomous Systems*, vol. 174, p. 104632, 2024.

[22] H. Aktas, U. Bozdogan, and E. Ugur, "Multi-step planning with learned effects of partial action executions," *Advanced Robotics*, vol. 38, no. 8, pp. 562–576, 2024.

[23] J. Liang, M. Sharma, A. LaGrassa, S. Vats, S. Saxena, and O. Kroemer, "Search-based task planning with learned skill effect models for lifelong robotic manipulation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6351–6357.

[24] T. Li, R. Calandra, D. Pathak, Y. Tian, F. Meier, and A. Rai, "Planning in learned latent action spaces for generalizable legged locomotion," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2682–2689, 2021.

[25] T. Girgin and E. Uğur, "Multiobject graph affordance network: Goal-oriented planning through learned compound object affordances," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 17, no. 4, pp. 847–858, 2024.

[26] P. Lorang, H. Lu, J. Huemer, P. Zips, and M. Scheutz, "Few-shot neuro-symbolic imitation learning for long-horizon planning and acting," in *9th Annual Conference on Robot Learning*, 2025.

[27] Y. Wang, R. Wu, K. Mo, J. Ke, Q. Fan, L. J. Guibas, and H. Dong, "Adaafford: Learning to adapt manipulation affordance for 3d articulated objects via few-shot interactions," in *European conference on computer vision*. Springer, 2022, pp. 90–107.
[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
[29] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," *arXiv preprint arXiv:1611.00712*, 2016.
[30] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.
[31] D. P. Kingma, M. Welling *et al.*, "Auto-encoding variational bayes," 2013.
[32] E. Oztop, N. S. Bradley, and M. A. Arbib, "Infant grasp learning: a computational model," *Experimental Brain Research*, vol. 158, no. 4, pp. 480–503, 2004.
[33] M. Y. Seker, M. Imre, J. Piater, and E. Ugur, "Conditional neural movement primitives," in *Proceedings of Robotics: Science and Systems*, FreiburgimBreisgau, Germany, June 2019.
[34] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2021.
[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[36] Q. McNemar, "Note on the sampling error of the difference between correlated proportions or percentages," *Psychometrika*, vol. 12, no. 2, pp. 153–157, 1947.
[37] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins, "PDDL—the planning domain definition language," Yale Center for Computational Vision and Control, Technical Report TR-98-003, 1998.