BusyBot: Learning to Interact, Reason, and Plan in a BusyBoard Environment

Anonymous Author(s)
Affiliation
Address
email

Abstract: We introduce BusyBoard, a toy-inspired robot learning environment that leverages a diverse set of articulated objects and inter-object functional relations to provide rich visual feedback for robot interactions. Based on this environment, we introduce a learning framework, BusyBot, which allows an agent to jointly acquire three fundamental capabilities (interaction, reasoning, and planning) in an integrated and self-supervised manner. With the rich sensory feedback provided by BusyBoard, BusyBot first learns a policy to efficiently interact with the environment; then with data collected using the policy, BusyBot reasons the inter-object functional relations through a causal discovery network; and finally by combining the learned interaction policy and relation reasoning skill, the agent is able to perform goal-conditioned manipulation tasks. We evaluate BusyBot in both simulated and real-world environments, and validate its generalizability to unseen objects and relations. Code and simulation will be publicly available.

Keywords: Manipulation, Learning Environment, Reasoning

1 Introduction

Learning through physical interactions plays a critical role in human cognitive development [1, 2, 3]. For instance, a well-designed toy like the "busyboard" (Fig. 1a) can provide an effective learning environment for children to develop fundamental manipulation and reasoning skills: the rich and amplified sensory feedback encourages children to actively explore and interact; and the observed inter-object functional relations (e.g., a switch turns on a light) facilitate the development of reasoning and task solving skills.

In this paper, we aim to provide a similar learning environment for embodied artificial agents, the **BusyBoard** environment, where agents learn to discover the underlying relations of objects through informative interactions and plan for goal-conditioned tasks. While simple at the first glance, this relational environment provides an *integrated* tool for learning and evaluating three critical capabilities of an embodied intelligent system:

- **Interact:** The ability to infer action affordances from visual observations – knowing where and how to manipulate an object to effectively change its state. Learning this skill through visual feedback is particularly hard for small-displacement objects (e.g., switches), whose appearance changes can be subtle even under effective actions.
- **Reason:** The ability to reason about inter-object functional relations (e.g., pressing a button turns on a light). In particular, the agent should learn to infer the relations by observing and predicting future states of the environment, without using the ground-truth relations as supervision.
- **Plan:** The ability to use the learned manipulation and reasoning skills in goal-conditioned planning tasks, in other words, generating a sequence of actions to transform the environment from a random initial state to a given goal state.

To learn these skills from the environment, we propose the **BusyBot** framework that acquires the above three capabilities through **self-supervised** interactions. To acquire the manipulation skill, the

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

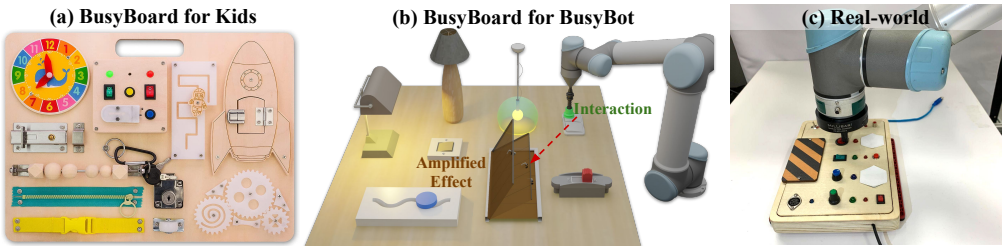


Figure 1: **BusyBoard Environments** inspired by toys for children, an integrated tool for learning and evaluating a robot's capabilities in interaction, reasoning, and planning.

algorithm learns a visual affordance model that infers effective action candidates through visual feedback. To reason about inter-object functional relations, the algorithm infers a functional scene graph and predicts the future states through a causal discovery network. Finally, to accomplish goal-conditioned manipulation tasks, the algorithm combines the learned action affordances, inter-object relationship, and dynamics to plan its actions with a model predictive control (MPC) framework. In summary, our contributions are two-fold:

- We introduce a new learning environment for embodied agents, **BusyBoard**, which features a diverse set of articulated objects with typical- and small- displacement joints, and rich inter-object functional relations.
- We propose **BusyBot**, an integrated learning framework which allows an embodied agent to acquire fundamental interaction, reasoning, and planning skills through self-supervised interactions and visual feedback.

Our experiments demonstrate that by taking advantage of the amplified effects (provided by BusyBoard), the agent is able to acquire an effective manipulation policy for small-displacement articulated objects (e.g., switches). Furthermore, by explicitly inferring the underlying inter-object functional relationship using a causal discovery network, the agent is able to predict the future states of the environment under different actions, and generalizes well to unseen environments, including real-world environment with physical robot interactions.

2 Related Work

Simulation environments for robot learning. Simulation environments are crucial for advances in robot learning. However, most of the existing simulated environments are developed for specific tasks or capabilities, such as navigation [4, 5, 6, 7], manipulation [8, 9, 10, 11, 12], causal reasoning in 2D [13, 14], or high-level task planning [15]. Inspired by human toys, our BusyBoard is an integrated environment that is compact and relevant to real-world applications, where an embodied agent can jointly learn three critical capabilities: interaction, reasoning, and planning.

Learning interaction policy. The ability to interact with a diverse set of objects is critical for many robotics tasks. Different methods have been proposed to learn interaction polices through human demonstrations [16, 17, 18] or self-guided explorations [19, 20, 21, 22]. However, most prior works have been ignoring a set of common but challenging objects: small-displacement objects. When interacting with these objects (e.g., switches), the effectiveness of an action often cannot be observed from the object's own visual appearance, which introduces significant challenges for learning. In this work, we address this challenge by taking advantage of BusyBoard, which amplifies action effects through responder objects and enables learning by enriching the supervision signal.

Inferring inter-object functional relations. Perceiving and understanding objects individually is often not sufficient for a lot of real-world applications that involve environments with multiple objects. Objects are usually related and understanding inter-object physical [23, 24, 25] or functional relations is a crucial skill for efficient planning. In this work, we will focus on uncovering the inter-object functional relationship, as defined by Li et al. [26]. One common approach to solve this problem is to induce changes through interventions and iteratively construct a functional scene graph [27]. More recently, Graph Neural Networks (GNNs) have been demonstrated to be promising for extracting the underlying structural causal model (SCM) and predicting future dynamics from motions [25]. In

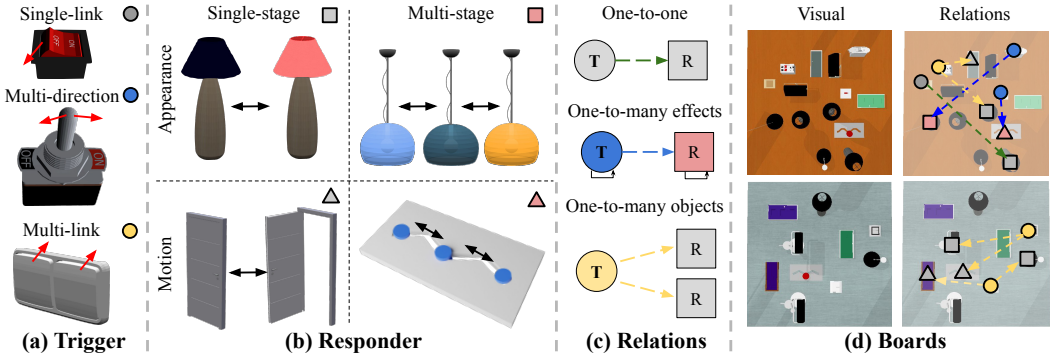


Figure 2: **BusyBoard Environment** is procedurally generated using articulated objects (a,b) with randomly sampled inter-object functional relation pairs between trigger and responder objects (c). (d) shows example boards and the underlying functional relations.

our work, we further demonstrate that GNNs are able to infer inter-object functional relations from changes in visual appearances. We also show that the inferred inter-object functional relationship and scene dynamics can assist action planning for downstream goal-conditioned manipulation tasks.

3 The BusyBoard Environment

To learn from a diverse set of environments, we designed a data generation pipeline to generate a large collection of simulated BusyBoard (illustrate on Fig. 2), which provides the following advantages:

- **Amplified effects.** In addition to providing visual feedback on the interacted object (e.g., appearance change on a button after being pressed), BusyBoard also amplifies the effect of an action with responder objects (e.g., a light turns on after the button is pressed). This is especially useful for learning manipulation policies for objects with small displacements upon interaction, for which state changes are often hard to observe.
- **Compact.** Unlike learning from room-scale environments [26], learning from BusyBoard does not require the agent to navigate in a large space to observe all the state changes. Thus BusyBoard is able to provide denser rewards for learning manipulation and reasoning before developing skills for navigation, which is analogous to the human cognition development process.
- **Relevant.** In contrast to game-like environments (e.g., Atari), our environment contains everyday objects and functional relations that are analogous to those in the real world, making the learned manipulation and reasoning skills relevant and potentially transferable to the real world.

Trigger and responder objects. The BusyBoard is procedurally generated using object URDF models. For trigger objects, we select switch instances from the Partnet-Mobility dataset [12], including small-displacement objects (with small displacements upon interaction), multi-direction objects (contain one movable link that can be pushed to multiple directions), and multi-link objects (contain multiple movable links). We use other object categories (e.g., lamp, door, tracktoy) as responder objects, which can be either single-stage or multi-stage. A single-stage responder has only two possible states defined by either appearance (e.g., lamp on or off) or joint state (e.g., door open or closed). A multi-stage responder has multiple possible states (e.g., multiple light colors of a lamp responder or multiple joint positions of a tracktoy responder).

Relations. One instance of BusyBoard may contain 2-3 triggers and 5-7 responders with randomly generated relations. Effects are triggered by changes in the joint positions of the trigger objects. We use Pybullet to simulate object movement under robot interactions. We introduce three types of inter-object functional relations (shown in Fig. 2):

- **One-to-one:** one trigger controls one single-stage responder.
- **One-to-many effects:** one trigger controls multiple effects on one responder. The trigger is a multi-direction object and the responder is a multi-stage object.
- **One-to-many objects:** one trigger controls multiple responders. The trigger is a multi-link object (a switch with multiple buttons), and each link controls one single-stage responder.

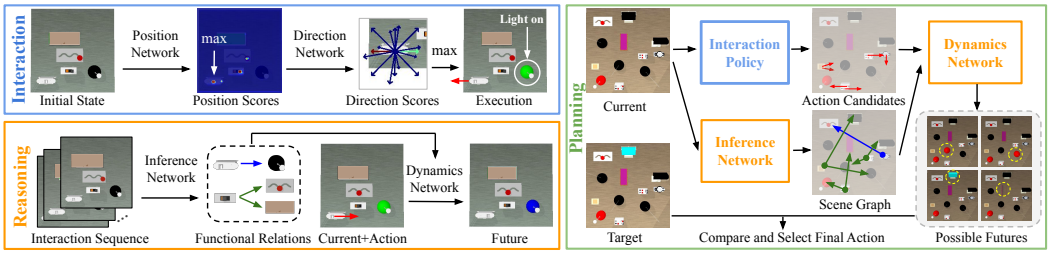


Figure 3: **BusyBot Overview.** [Interaction] infers a sequence of actions to efficiently interact with a given scene from visual input. [Reasoning] infers a functional scene graph (i.e., inference network) and predicts future states (i.e., dynamic network). [Planning] uses the manipulation policy network (learned from multiple boards), inference and dynamics network (extracted from the specific board) to plan actions for the target state.

In goal-conditioned tasks, for both "one-to-many effects" and "one-to-many objects" relations, the algorithm needs to not only know the trigger object to interact with, but also the direction and position of the action to execute. In this paper, we use "one-to-many" to refer to the super-set of both categories. We exclude many-to-one relations to eliminate possible ambiguities.

4 The BusyBot Framework

The goal of BusyBot is to learn how to meaningfully interact with the BusyBoard environment (§4.1), infer the inter-object functional relations and dynamics through these interactions (§4.2), and eventually perform goal-conditioned manipulation using the learned interaction policy, relations, and dynamics. (§4.3). We will discuss each module in detail below.

4.1 Interact: Learning to Interact with Amplified Effects

The goal of the interaction policy π is to generate a sequence of actions to interact with articulated objects (i.e. triggers). The interaction policy should be able to: (a) choose the right interact position, (b) select a proper action direction based on the object's articulation structure (i.e., move up-and-down or left-and-right). (c) interact with different objects to explore novel states of the board.

Action Inference. The task is formulated as: given a top-down depth image $o_t \in \mathbb{R}^{W\times H}$, the agent with policy π generates an action a_t at each step: $\pi(o_t) \rightarrow a_t$. The action is represented in SE(3) space, parameterized by an end-effector (i.e., a suction-based gripper) position and a moving direction $a_t = (a_t^{\text{pos}}, a_t^{\text{dir}})$, where $a_t^{\text{pos}} \in \mathbb{R}^3$ is the 3D coordinate and $a_t^{\text{dir}} \in \mathbb{R}^3, (||a_t^{\text{dir}}|| = 1)$ is a unit vector in 3D indicating the moving direction of the end-effector. The moving distance is incrementally assigned until reaching a pre-defined limit.

The position network takes the depth image as input and outputs per-pixel position affordance $P \in [0,1]^{W\times H}$, which indicates the likelihood of an effective interact position. The direction inference network takes in the depth observation and the selected action position (represented as a 2-D Gaussian distribution centered around the corresponding pixel location of the 3-D action position) and outputs a score for each direction candidate $r(a_t^{\text{dir}}) \in [0,1]$. We uniformly sample 18 directions in SO(3) as the direction candidates. Since it is often hard to identify the state of small-displacement objects from visual observations, the agent executes both the selected direction and its opposite direction.

Supervision. For each executed action the reward is computed by image difference:

$$r_{\text{img}}(a_t^{\text{dir}}) = \begin{cases} 1 & \text{if } \sum_{i=1}^{H}\sum_{j=1}^{W} I(o_{ij}, o'_{ij}) > \delta \\ 0 & \text{if } \sum_{i=1}^{H}\sum_{j=1}^{W} I(o_{ij}, o'_{ij}) \leq \delta \end{cases} \quad I(o_{ij}, o'_{ij}) = \begin{cases} 1 & \text{if } o_{ij} = o'_{ij} \\ 0 & \text{if } o_{ij} \neq o'_{ij} \end{cases} \tag{1}$$

where o and o' denote RGB image observations before and after the action execution. δ is a threshold specifies the minimum number of different pixels. We use binary cross-entropy (BCE) loss between the inferred action score $r(a_t^{\text{dir}}) \in [0,1]$ and the ground-truth reward computed from image observations $r_{\text{img}}(a_t^{\text{dir}}) \in [0,1]$.

Exploration. At the early stage of training the position and direction inference network, we use the epsilon-greedy method to encourage random exploration. Additionally, in order to prevent the model from only selecting the position that has the highest affordance score, we apply the Upper Confidence

Bound (UCB) Bandit algorithm on the inferred position affordance to encourage exploration of all object positions in the environment. Given the position affordance P, the updated position affordance is $P'(i,j) = P(i,j) + c\sqrt{\frac{ln(t)}{N}}$, where $c = 0.5$, t is the number of steps, and N is the times when the pixel (i,j) falls in the $M \times M (M = 10)$ window centered around each previously selected pixels.

4.2 Reason: Learning to Discover Inter-object Relations by Predicting the Future

To meaningfully interact within a multi-object environment, the robot not only needs to manipulate individual objects, but also needs to learn the inter-object functional relations. The reasoning module takes in RGB image sequences of the agent's interactions (§4.1), infers the inter-object relations, and predicts future dynamics, which would guide goal-conditioned planning in the next section (§4.3). To accomplish this goal, we adopt and modify the V-CDN model [25].

The interaction dataset is collected using learned interaction policy: at each step, positions with affordance score above a threshold are grouped into clusters using the K-means clustering algorithm ($k = 7$, the maximum possible number of movable links on all busyboards), and positions with the highest score in each cluster are selected as position candidates. Conditioned on each position candidate, directions with the highest affordance score will be selected to form the final action candidate, from which an action will be randomly chosen to execute. For each board environment, RGB image observations of 30 interaction steps are generated and split into two subsequences: the first T steps for inferring the functional scene graph, and the rest for predicting future dynamics. In addition, to prevent the model from overfitting on board appearances, we ensure that every 20 board environments share the same initial visual appearance but with different functional relations.

The inference network is implemented with three Graph Neural Networks (GNNs) to extract functional relations as a scene graph. Each object O_i corresponds to a node i in the graph, with a node input $n_i^{1:T}$ that combines the object's 256-dimensional visual features (extracted at the centers of the objects' bounding boxes using the 10th layer of ResNet-50) and the object's 3D position. The first GNN learns spatial node and edge embeddings at each step, which are concatenated with 256-dimensional embeddings of the executed actions $a_i^{1:T}$ learned from a MLP layer. The combined embeddings are then aggregated over temporal dimension using a 1-D convolution network and input to the second GNN which predicts a probabilistic distribution over edge types $e^d = \{e_{ij}^d | e_{ij}^d \in \mathbb{R}^2\}_{i,j=1}^N$ (index 0 indicates no relation and index 1 indicates has relation). Conditioned on the edge types, the third GNN predicts 32-dimensional edge embeddings $e^h = \{e_{ij}^h | e_{ij}^h \in \mathbb{R}^{32}\}_{i,j=1}^N$ which stores history dynamics associated with each edge.

The dynamics network is a Graph Recurrent Network (GRN) that predicts the next state n^{t+1} given the current observation n^t, the executed action a^t, and the edges $E = \{e^d, e^h\}$ from the inferred functional scene graph. The inference and dynamics network are jointly trained on the objective to minimize the mean squared error (MSE) between predicted and ground-truth object features.

$$L = \min_{\phi,\psi} \sum_t MSE(n^{t+1}, f_\psi^D(n^t, a^t, f_\phi^I(n^{1:T}, a^{1:T}))) \tag{2}$$

where f_ϕ^I is the inference model parameterized by ϕ, f_ψ^D is the dynamics model parameterized by ψ.

4.3 Plan: Goal-conditioned Manipulation with Relation Predictive Agent

Finally, we apply BusyBot on goal-conditioned manipulation tasks. Given an initial and target state image of a board, the task is to infer 1) which object(s) to manipulate; 2) what action(s) to execute in order to successfully reach the target state.

Using the data collection method as discussed in the reasoning module, the agent infers an action candidate set and generate an interaction sequence of 30 images, which are input to the inference network to obtain the functional scene graph. Then we consider three options to plan for goal-conditioned tasks: 1) **Relation agent**, at each step, identify a responder that needs to be changed, and find the corresponding trigger based on the functional scene graph. This method is similar to the idea of Li et al. [26]. However, the agent might have trouble handling one-to-many relations. To solve this issue, we propose 2) **Predictive agent** that uses the dynamics network from the reasoning module and choose the action that minimizes the L2 distance of the predicted next state and the target state.

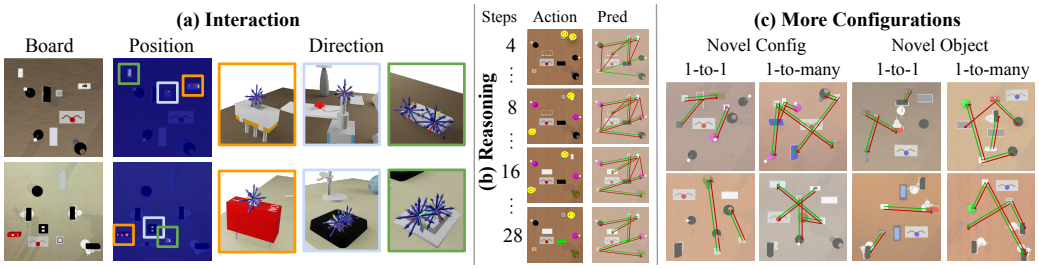


Figure 4: **Qualitative Results**, (a) action affordances (b) interaction steps and corresponding reasoning results (c) More reasoning results →: inferred inter-object functional relations. →: ground truth.

However, the predictive agent may have difficulty generalizing to novel object instances, due to the difficulty of predicting unseen dynamics. 3) Our final method **BusyBot** combines the relation and predictive agent, where action candidates are first filtered based on the functional scene graph and then selected based on future dynamic predictions. More discussions are provided in Sec. 5.

5 Evaluation

We evaluate BusyBot with both simulated (Fig. 4) and real-world busyboards (Fig. 5). In simulation experiments, we set up the following environments: **a) Training Board**: for training interaction and reasoning module. **b) Novel Config**: testing board constructed with training object instances, but in new configurations, which includes inter-object functional relations, position and orientation of objects, board color and texture; **c) Novel Object**: both object instances and board configurations are novel. In total, we generate 10,000 training boards, 2000 boards with novel configurations, and 2000 boards with novel object instances. Each board has 30 interaction images, where 23 images are reserved for relation inference and the rest for future predictions. As for objects on the board, we use 41 switches, 10 doors, 5 lamps, and 2 tracktoy objects, split into training / testing with ratio: 32/9, 5/5, 3/2, 1/1. The setup for real-world evaluation is described in Sec 5.4

5.1 Interaction Module Evaluation

To evaluate the interaction policy network, we compute the average precision and recall of the inferred actions for the boards, where precision = # successful actions / # total proposed actions, and recall = # successfully interacted objects / # total interactable objects. We compare with the following methods:

- **Oracle (joint state supervision):** Interaction policy supervised on joint states. This is considered as the oracle because changes in joint states are directly obtained from simulation.
- **w/o responder:** Interaction policy supervised on visual feedback but no responder effects.
- **w/o exploration:** An ablated version of BusyBot without using UCB for exploration.

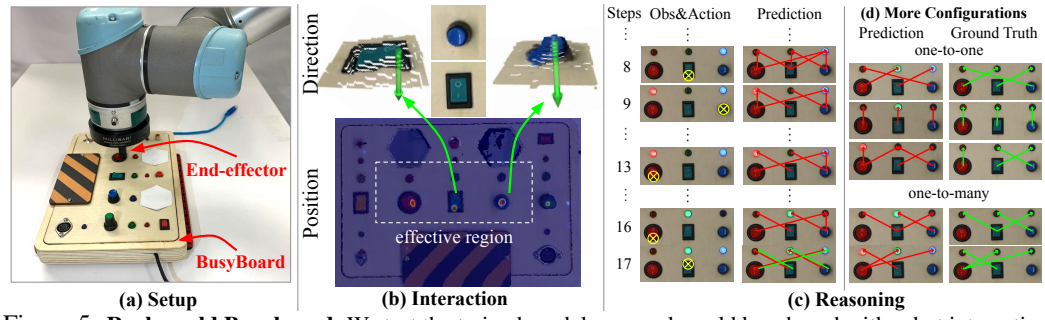
Results and Analysis. Tab. 1 summarizes the result tested on boards with novel configurations and objects. We can see that [w/o responder] achieves poor performence since the visual feedback of small-displacement objects alone is insufficient for learning. In contrast, by taking into account the responder effects, [BusyBot] is able to get informative reward and achieves comparable performance with the oracle. This validates our hypothesis that triggered responder effects can be used to amplify the visual feedback and assist the model in learning good interaction policies. We also did an ablation study on exploration and observe that without encouraging exploration of new objects, the recall of [w/o exploration] dropped by more than 45% than that of [BusyBot].

	Novel Config		Novel Object	
	Prec	Recall	Prec	Recall
Oracle	91.8	82.4	79.5	90.2
w/o responder	0.71	0.65	0.24	0.49
w/o exploration	**94.2**	33.7	81.9	38.6
BusyBot	90.1	**80.1**	**82.6**	**84.8**

Table 1: **Performance of Interaction Policy**

5.2 Relation Reasoning Module Evaluation

The reasoning module is evaluated by the following metrics: 1) Relation inference accuracy, measured by the precision (**Edge-P**) and recall (**Edge-R**) of the inferred functional relation pairs. 2) Future state prediction accuracy (**Pred-A**), measured by the percentage of correct future state predictions. We compare the following alternative methods:



(a) Setup (b) Interaction (c) Reasoning

Figure 5: **Real-world Busyboard.** We test the trained model on a real-world busyboard with robot interactions (a). We also manually modify the underlying inter-object functional relations and show that the algorithm discovers the relations (c) through interactions (b). More results are included in supp.

- **w/o inference:** An ablated version of the model without the inference network. The dynamics network takes in all history interaction data and directly predicts the next state.
- **Bad interact:** An ablated version of our method, where the input of the reasoning network are data collected under an inferior interaction policy (w/o exploration).

Results and Analysis. Comparing to [w/o inference], we see that without inferring the inter-object relations, the model overfits on the training data and generalizes poorly to boards with novel configurations and objects. We also observe that with [Bad interact], the reasoning model is not able to uncover the relations accurately and make correct future state predictions. In comparison, our model generalizes well to novel board configurations and achieves performance comparable to that of the training board. This demonstrates that a good interaction policy helps the agent uncover the correct inter-object functional relations, which then helps the agent to understand scene dynamics. For boards with novel object instances, even though the future state prediction accuracy drops by around 40% than the seen instances, which is expected since the object features are never seen by the dynamics model, the relation inference accuracy is still comparable to boards with seen object instances. The performance on novel boards verifies that the model's ability to infer inter-object functional relationship can transfer to new scenes and new objects.

	Training Board			Novel Config			Novel Object		
	Edge-P	Edge-R	Pred-A	Edge-P	Edge-R	Pred-A	Edge-P	Edge-R	Pred-A
w/o inference	-	-	79.2	-	-	36.2	-	-	7.04
w/o exploration	55.6	51.0	**89.6**	74.6	3.10	14.3	75.1	0.96	11.6
BusyBot	**95.8**	**100**	88.1	**95.5**	**99.7**	**73.8**	**85.0**	**99.5**	**31.0**

Table 2: **Performance of Reasoning Module.** For BusyBot, while the future state prediction accuracy (Pred-A) decreases for unseen board appearances (novel config, novel object), the reasoning module is still able to reliably infer the inter-object functional relations (Edge-P, Edge-R) in novel scenarios.

5.3 Goal-Conditioned Manipulation Evaluation

We generate 50 one-to-one tasks and 50 one-to-many tasks for each type of board (training, novel config, novel object). One-to-one tasks contain only two-state triggers, and thus only require the algorithm to identify the correct trigger (similar task studied in IFRexplore [26]). One-to-many tasks contain both multi-direction and multi-link triggers that require the agent to not only identify the correct trigger, but also infer the correct action to manipulate the trigger (e.g., the correct button position or pushing direction).

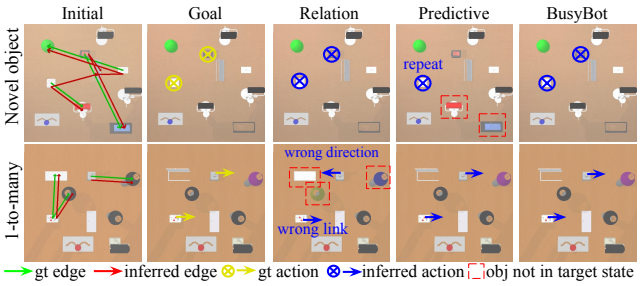


Figure 6: **Goal-conditioned manipulation.** Compared to the predictive agent, the relation agent generalizes better on novel objects, while struggles in handling one-to-many relations. Our method BusyBot combines the advantages of both agent.

Metrics & Baselines. We measure object-level success rate on both one-to-one and one-to-many manipulation tasks for each type of board. The success rate is defined at object level at the end of an

interaction sequence with a maximum of 8 steps. Success rate = # affectable responders in goal state / # total affectable responders. We compare the three agents as discussed in the method section.

Results and Analysis. All agents achieve good performances on one-to-one tasks. This means that both the relation and dynamics learned by the reasoning module can generalize to novel board configurations and objects. The predictive agent achieves better performance on one-to-many tasks with seen object instances by leveraging future predictions to select the correct action to apply on the trigger object. In contrast, the relation agent can only identify the trigger object but not the exact action (e.g., which link to interact with or which direction to push). On the other hand, the relation agent performs slightly better than the predictive agent on all one-to-one tasks and boards with novel objects, when the dynamics model sometimes fails to predict the correct next state. This shows that inter-object functional relationship can generalize to scenarios when future predictions are not reliable enough to assist planning. Our method combines the advantages of both relation and predictive agent and achieves the highest performance on all tasks.

	Training		Novel Config		Novel Object	
	1-to-1	1-to-m	1-to-1	1-to-m	1-to-1	1-to-m
Relation	98.3	61.1	93.7	60.0	92.0	62.8
Predictive	97.7	67.5	91.0	67.0	89.0	58.2
BusyBot	**98.3**	**71.0**	**93.7**	**69.4**	**92.3**	**64.9**

Table 3: **Goal-conditioned Manipulation Result**

5.4 Real-world Experiments

Setup. We test the trained model on a busyboard in real world with robot interactions (Fig. 5). The board consists of 3 intractable trigger objects (switches) and 3 responder objects (LEDs). Objects outside the effective region are ignored. We manually modified the underlying inter-object functional relations of the board by rewiring the objects. We test with 6 different configurations including 4 one-to-one and 2 one-to-many configurations. For each configuration, the robot interacts with the board for 30 steps and the rollout is grouped into 6 overlapping and continual sub-sequences, each of which has a length of 25. In total, we generate a real-world dataset of 36 sequences with 108 inter-object functional relation pairs for evaluating the reasoning module.

Results. Fig. 5 (c) shows example results, where the algorithm is able to refine inter-object functional relations (remove additional edges) through interactions. The precision and recall of inferred relations are **93.9%** and **100%**, respectively. All inter-object functional relations can be discovered by our model, with only a few additional pairs predicted. The result shows that the relation reasoning ability of the model is transferable to real-world scenarios.

5.5 Limitation and Future Work

While BusyBoard environment is inspired by toys, it still lacks some of the diversity and complexity that appear in real-world toys. For example, real-world toys are often designed with multi-sensory feedback including both sound and tactile, while our environment focuses on visual effects only.

The BusyBot's interactions assume the trigger objects are small and can reach all states through single-step actions. However, it does not accommodate large-displacement object that might require a sequence of actions [20]. The relation reasoning module of BusyBot assumes objects are detected. This assumption is easy to satisfy when the environment is simple and objects are well-separated. However, this assumption may not hold when the environment gets more complicated and cluttered.

6 Conclusion

We propose a toy-inspired relational environment, BusyBoard, and a learning framework, BusyBot, for embodied AI agents to acquire interaction, reasoning, and planning abilities. Our experiments demonstrate that the rich sensory feedback and amplified effects in BusyBoard help the agent learn a policy to efficiently interact with the environment; using the data collected under this interaction policy, inter-object functional relations can be inferred and used for predicting future states; and by combining the ability to interact and reason, the agent is able to perform goal-conditioned manipulation tasks. We verify the effectiveness and generalizability of our method in both simulation and real-world setups.

References

[1] R. Held and A. Hein. Movement-produced stimulation in the development of visually guided behavior. *Journal of comparative and physiological psychology*, 56, 1963. doi:10.1037/h0040546.

[2] G. E. Roberson, M. T. Wallace, and J. A. Schirillo. The sensorimotor contingency of multisensory localization correlates with the conscious percept of spatial unity. *Behavioral and Brain Sciences*, 24(5), 2001. doi:10.1017/S0140525X0154011X.

[3] R. Baillargeon. Infants' physical world. *Current Directions in Psychological Science*, 13(3), 2004. doi:10.1111/j.0963-7214.2004.00281.x.

[4] D. Perille, A. Truong, X. Xiao, and P. Stone. Benchmarking metric ground navigation. In *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 116–121. IEEE, 2020.

[5] N. Tsoi, M. Hussein, O. Fugikawa, J. D. Zhao, and M. Vázquez. An approach to deploy interactive robotic simulators on the web for hri experiments: Results in social robot navigation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021.

[6] Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[7] C. Li, F. Xia, R. Martín-Martín, M. Lingelbach, S. Srivastava, B. Shen, K. E. Vainio, C. Gokmen, G. Dharan, T. Jain, A. Kurenkov, K. Liu, H. Gweon, J. Wu, L. Fei-Fei, and S. Savarese. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=2uGN5jNJROR.

[8] T. Yu, D. Quillen, Z. He, R. Julian, A. Narayan, H. Shively, A. Bellathur, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning, 2021.

[9] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

[10] O. Ahmed, F. Träuble, A. Goyal, A. Neitz, Y. Bengio, B. Schölkopf, M. Wüthrich, and S. Bauer. Causalworld: A robotic manipulation benchmark for causal structure and transfer learning, 2020.

[11] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

[12] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas, and H. Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[13] A. Bakhtin, L. van der Maaten, J. Johnson, L. Gustafson, and R. Girshick. Phyre: A new benchmark for physical reasoning, 2019.

[14] A. Jain, A. Szot, and J. Lim. Generalization to new actions in reinforcement learning. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4661–4672. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/jain20b.html.

[15] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi. Ai2-thor: An interactive 3d environment for visual ai, 2019.

[16] S. Brahmbhatt, C. Ham, C. C. Kemp, and J. Hays. Contactdb: Analyzing and predicting grasp contact via thermal imaging, 2019.

[17] T. Nagarajan, C. Feichtenhofer, and K. Grauman. Grounded human-object interaction hotspots from video. In *ICCV*, 2019.

[18] T. Nagarajan, Y. Li, C. Feichtenhofer, and K. Grauman. Ego-topo: Environment affordances from egocentric video, 2020.

[19] K. Mo, L. Guibas, M. Mukadam, A. Gupta, and S. Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *ICCV*, 2021.

[20] Z. Xu, H. Zhanpeng, and S. Song. Umpnet: Universal manipulation policy network for articulated objects. *IEEE Robotics and Automation Letters*, 2022.

[21] B. Eisner, H. Zhang, and D. Held. Flowbot3d: Learning 3d articulation flow to manipulate articulated objects. *RSS*, 2022.

[22] S. Y. Gadre, K. Ehsani, and S. Song. Act the part: Learning interaction strategies for articulated object part discovery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15752–15761, 2021.

[23] P. Battaglia, R. Pascanu, M. Lai, D. Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in neural information processing systems*, 29, 2016.

[24] C. Mitash, A. Boularias, and K. Bekris. Physics-based scene-level reasoning for object pose estimation in clutter. *The International Journal of Robotics Research*, page 0278364919846551, 2019.

[25] Y. Li, A. Torralba, A. Anandkumar, D. Fox, and A. Garg. Causal discovery in physical systems from videos. *Advances in Neural Information Processing Systems*, 33, 2020.

[26] Q. Li, K. Mo, Y. Yang, H. Zhao, and L. Guibas. IFR-Explore: Learning inter-object functional relationships in 3d indoor scenes. In *International Conference on Learning Representations (ICLR)*, 2022.

[27] S. Nair, Y. Zhu, S. Savarese, and L. Fei-Fei. Causal induction from visual observations for goal directed tasks, 2019.