Cross-Domain Demo-to-Code via Neurosymbolic Counterfactual Reasoning

Jooyoung Kim, Wonje Choi, Younguk Song, Honguk Woo*
Department of Computer Science and Engineering, Sungkyunkwan University
{onsaemiro, wjchoi1995, syw2045, hwoo}@skku.edu

Abstract

Recent advances in Vision-Language Models (VLMs) have enabled video-instructed robotic programming, allowing agents to interpret video demonstrations and generate executable control code. We formulate video-instructed robotic programming as a cross-domain adaptation problem, where perceptual and physical differences between demonstration and deployment induce procedural mismatches. However, current VLMs lack the procedural understanding needed to reformulate causal dependencies and achieve task-compatible behavior under such domain shifts. We introduce NeSyCR*, a neurosymbolic counterfactual reasoning framework that enables verifiable adaptation of task procedures, providing a reliable synthesis of code policies.* NeSyCR *abstracts video demonstrations into symbolic trajectories that capture the underlying task procedure. Given deployment observations, it derives counterfactual states that reveal cross-domain incompatibilities. By exploring the symbolic state space with verifiable checks,* NeSyCR *proposes procedural revisions that restore compatibility with the demonstrated procedure.* NeSyCR *achieves a 31.14% improvement in task success over the strongest baseline Statler, showing robust cross-domain adaptation across both simulated and real-world manipulation tasks.*

1. Introduction

Advances in foundation models have accelerated progress toward general-purpose embodied intelligence, enabling robots to interpret human instructions and execute complex tasks as autonomous control policies [7, 15, 25, 51]. In particular, Large Language Models (LLMs) with code-writing capabilities have inspired the Code-as-Policies paradigm, in which executable control code is synthesized from language instructions using predefined APIs [24, 26, 36, 54]. Furthermore, Vision-Language Models (VLMs) have extended this paradigm toward a general form of video-instructed robotic programming, where robotic programs are generated from instructional video demonstrations by translating observed task sequences into structured task specifications that can be compiled into control code [56, 58, 61, 65]. By capturing richer perceptual context and task intent from demonstrations, these methods enable more grounded robotic programming than text instructions alone.

In video-instructed robotic programming, domain gaps between the demonstration and deployment are inevitable due to inherent differences in environmental layouts, object properties, and morphological constraints [11, 16, 45, 46]. While sensory observations can reveal physical discrepancies between the demonstration and the deployment environment, these observations alone do not explain how structural differences disrupt the underlying task procedure or how actions will causally unfold under altered conditions. To address this limitation, we present NeSyCR, a neurosymbolic framework that formulates cross-domain adaptation from demonstrations as counterfactual reasoning. At its core, NeSyCR infers how task outcomes would change under altered domain factors and proposes alternative actions that revise the demonstrated behavior to restore task-oriented behavioral compatibility. Given a demonstration, NeSyCR constructs a symbolic world model that encodes the task procedure as a symbolic abstraction of the trajectory. Leveraging this model, NeSyCR performs neurosymbolic counterfactual adaptation to revise the procedure for the deployment domain, enabling verifiable adaptation of the demonstrated procedure across domains.

As shown in Figure 1, the demonstration domain involves a human performing a drawer-organizing task with objects such as a magnetic hook and a box of screws placed on the table. In the deployment domain, however, the human hand is replaced by a robotic gripper, the magnetic hook is already arranged inside the drawer, and the screws are scattered across the surface. These differences create a procedural incompatibility: direct grasping becomes infeasible, requiring the magnetic hook to be repurposed as an auxiliary tool for gathering the screws. Furthermore, this modification introduces a cascading incompatibility: objects to be organized later in the procedure can obstruct the magnet, rendering it inaccessible. Resolving these coupled

*Honguk Woo is the corresponding author.

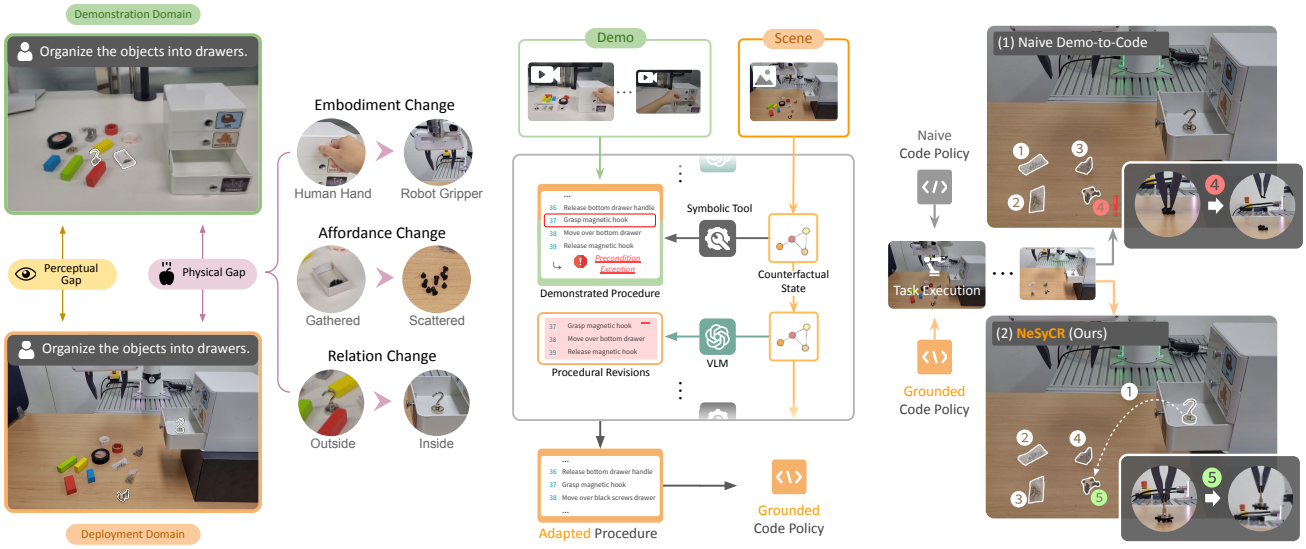


Figure 1. Overview of NESYCR in a drawer-organizing task scenario. **(Left)** Illustration of the domain gap between the demonstration and deployment. **(Middle)** Overview of NESYCR framework, which generates an adapted procedure via neurosymbolic counterfactual reasoning. **(Right)** Outcome of the adapted procedure, showing that NESYCR successfully executes the task via a grounded code policy.

incompatibilities requires reordering the procedure so that the magnet is retrieved and used to aggregate the screws before the target objects are placed. However, a direct demo-to-code approach using VLM-based prompting would still attempt to grasp the scattered screws with the gripper, which fails due to their small size and dispersion. In contrast, NESYCR identifies these incompatibilities through VLM-based symbolic translation and symbolic forward simulation. By combining the VLM's commonsense reasoning with symbolic verification, it produces an adapted and coherent procedure that completes the task. Further details are illustrated in Figure 5.

To construct the symbolic world model, the VLM abstracts the demonstration into a symbolic trajectory that captures objects and their spatial and temporal relations, while the symbolic tool verifies these representations for logical consistency. In a deployment domain, NESYCR identifies structural variations through this symbolic world model, detecting where the demonstrated procedure fails to reach the goal state under counterfactual states derived from target-domain observations. The VLM then proposes alternative action operators for the incompatible steps, and the symbolic tool verifies their causal validity, yielding revised procedural steps that restore compatibility. This reasoning process produces an adapted task specification that remains logically valid while preserving the task intent of the demonstration. The adapted specification is then compiled into a reliable, deployment-grounded code policy.

We evaluate NESYCR on a diverse set of video-instructed robotic programming scenarios, deploying robotic agents in both simulated and real-world environments. The cross-domain setting between demonstration and deployment is characterized by domain factors that capture variations in environmental and embodiment configurations. The scenarios consist of long-horizon manipulations involving multiple subtask types and requiring up to 116 visual-motor API calls, yielding compositional and procedurally complex tasks. NESYCR outperforms the strongest baseline, Statler [66], achieving an average improvement of 31.14% in task success rate, as reported in Tables 1 and 2.

Our contributions are summarized as follows: (1) We present the NESYCR framework that casts cross-domain adaptation from demonstrations as counterfactual reasoning for video-instructed robotic programming. (2) We implement a VLM–symbolic tool pipeline that proposes and verifies alternative action steps, ensuring procedural compatibility across domains. (3) We evaluate NESYCR across simulated and real-world robotic manipulation tasks using an experimental design that provides high granularity over domain factors and task complexity, enabling precise analysis of its cross-domain procedural adaptation.

2. Problem formulation

We formulate the embodied domain as a tuple $\mathcal{M} = (\mathcal{S},\mathcal{A},\mathcal{T},g)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, $\mathcal{T}:\mathcal{S}\times\mathcal{A}\to\mathcal{S}$ the transition function, and $g\in\mathcal{G}$ the goal state. Under partial observability [52], the agent receives observations $o_t\in\mathcal{O}$ at each timestep t via an observation function $\Omega:\mathcal{S}\times\mathcal{A}\to\mathcal{O}$, which maps the underlying state to perceptual observations (e.g., RGB images). In cross-domain settings, a demonstration $\mathcal{D}=(\{o_t\}_{t=1}^{N},\ell)$ specifies a domain $\mathcal{M}_{\mathrm{S}} = (\mathcal{S}_{\mathrm{S}},\mathcal{A}_{\mathrm{S}},\mathcal{T}_{\mathrm{S}},g)$ performed under specific environmental and embodiment configurations, optionally

with a language description ℓ. The agent must achieve the same goal in a deployment domain $\mathcal{M}_{\mathrm{T}} = (\mathcal{S}_{\mathrm{T}}, \mathcal{A}_{\mathrm{T}}, \mathcal{T}_{\mathrm{T}}, g)$, where $\mathcal{S}_{\mathrm{S}} \neq \mathcal{S}_{\mathrm{T}}$, $\mathcal{A}_{\mathrm{S}} \neq \mathcal{A}_{\mathrm{T}}$, or $\mathcal{T}_{\mathrm{S}} \neq \mathcal{T}_{\mathrm{T}}$. Our objective is to optimize a policy π_θ from a single demonstration to maximize task success in the deployment domain:

$$\pi_\theta^* = \underset{\pi_\theta}{\operatorname{argmax}}\, \mathbb{E}_{\tau \sim p(\cdot \mid \pi_\theta, \mathcal{M}_{\mathrm{T}})} \left[\mathrm{SR}(\tau, g) - \lambda \mathrm{D}(\tau, \mathcal{D}) \right] \quad (1)$$

where $\tau = \{o_1, a_1, \ldots, o_N, a_N\}$ denotes a trajectory sampled from $p(\tau \mid \pi_\theta, \mathcal{M}_{\mathrm{T}})$, induced by executing π_θ in $\mathcal{M}_{\mathrm{T}}$. Here, $\mathrm{SR}(\tau, g)$ measures task success, $\mathrm{D}(\tau, \mathcal{D})$ measures the deviation between τ and $\mathcal{D}$, and λ is a weighting factor. Thus, the policy π_θ^* aims to maximize task success in the deployment domain while maintaining alignment with the demonstration. Each timestep t corresponds to a semantically coherent interval, where the observation o_t reflects the causal effect of action a_t [22, 27]. We represent π_θ as executable control code such that τ corresponds to its execution trace during deployment [36, 58].

3. Neurosymbolic Counterfactual Reasoning

We formulate video-instructed robotic programming as a cross-domain adaptation problem, in which the agent must transform an instructional video demonstration recorded into a logically verified and deployable task specification for execution in deployment domains with diverse environmental or embodiment conditions. Such domain gaps alter how the agent interacts with its environment, thereby inducing procedural discrepancies between the demonstrated and deployable behaviors. Rather than directly imitating the demonstrated behavior, the agent must reason about whether and how the demonstrated procedure should be revised under structural variations, thereby adjusting its actions to preserve task-level consistency.

We address this challenge through neurosymbolic counterfactual reasoning (NeSyCR), which bridges the domain gaps between demonstrations and target observations by enabling procedure adaptation. NeSyCR translates a demonstration into a symbolic representation of preconditions, actions, and effects, thereby constructing a symbolic world model. Given a target observation, the framework identifies its corresponding counterfactual situation by verifying which preconditions in the procedure are satisfied or violated. Based on this, NeSyCR adapts the demonstrated procedure by adding or removing actions so that the resulting procedure aligns the counterfactual state with the desired preconditions and effects. Specifically, NeSyCR integrates VLM-based procedural alternative generation with symbolic verification and operates in two phases: (i) *symbolic world model construction* and (ii) *neurosymbolic counterfactual adaptation*. In phase (i), the demonstration is abstracted into a compact symbolic state sequence, from which a symbolic world model is constructed. In phase (ii),

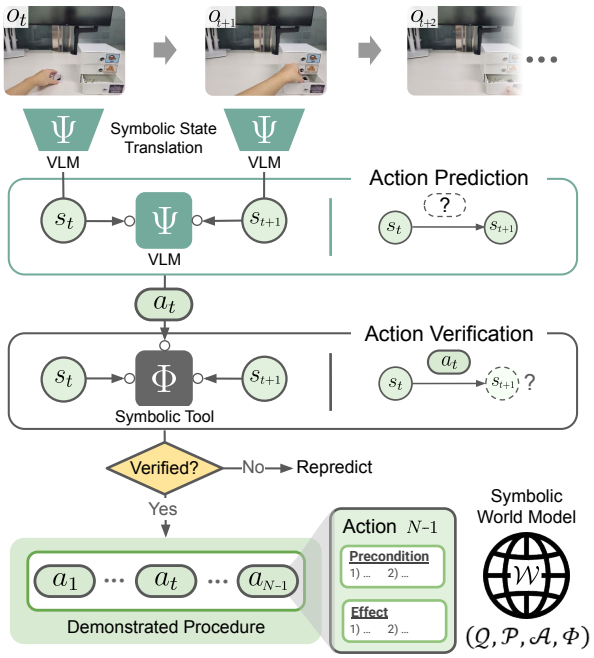


Figure 2. Symbolic world model construction

the framework contrasts this symbolic world model with target-domain observations to derive counterfactual states. The symbolic tool identifies where the demonstrated procedure fails, the VLM proposes alternatives to repair these steps, and the symbolic tool verifies their causal validity. Through this neurosymbolic loop, NeSyCR produces a causally coherent, deployment-grounded task specification that is compiled into an executable code policy.

3.1. Symbolic world model construction

NeSyCR translates the demonstration into a symbolic world model that encodes the causal structure of the demonstrated behavior, ensuring its reproducibility within a symbolic state space. As shown in Figure 2, rather than treating the demonstration as a raw sequence of observations, the VLM abstracts it into symbolic transitions. Consecutive observations are parsed into scene graphs representing symbolic states, from which the VLM predicts a symbolic operator that specifies the preconditions and effects of the executed action. The symbolic tool then verifies the consistency of these transitions and integrates them into a unified world model that is logically coherent and reconstructable with respect to the demonstrated behavior. This symbolic world model $\mathcal{W}$ serves as a plan verification model and is expressed in a STRIPS-style formalism [17], supporting forward execution and logical validation.

$$\mathcal{W} = (\mathcal{Q}, \mathcal{P}, \mathcal{A}, \Phi), \quad \Phi : (s, a) \mapsto s' \quad (2)$$

Here, $\mathcal{Q}$ denotes the set of objects identified in the scene, $\mathcal{P}$ the set of predicates representing object affordances and

spatial relations, and $\mathcal{A}$ the set of symbolic actions, each defined by preconditions and effects over $\mathcal{P}$. Φ denotes the symbolic tool (e.g., VAL [18, 23]) responsible for forward simulation and consistency verification. Given a current symbolic state $s \in 2^{\mathcal{P}}$ and an action $a \in \mathcal{A}$ whose preconditions are satisfied in s, Φ applies the effects of a to produce the next symbolic state s'.

Symbolic state translation. To obtain a symbolic state sequence from the demonstration, NESYCR prompts the VLM Ψ to generate grounded scene graphs for key frames [34, 60]. At each timestep t, Ψ extracts object entities and spatial relations from the image observation o_t and language description ℓ, forming a symbolic state s_t:

$$\Psi(\{o_1, \ldots, o_N\}; \ell) = \{s_1, \ldots, s_N\},$$
$$\mathcal{Q} = \bigcup_{t=1}^{N} \text{Object}(s_t), \quad \mathcal{P} = \bigcup_{t=1}^{N} \text{Predicate}(s_t) \tag{3}$$

where $\text{Object}(s_t)$ and $\text{Predicate}(s_t)$ denote the objects and predicates grounded in s_t, respectively. The resulting sequence $\{s_1,\ldots,s_N\}$ represents a dynamic scene graph as a sequence of symbolic states derived from the demonstration, grounded in the object set $\mathcal{Q}$ and predicate set $\mathcal{P}$.

Symbolic dynamics reconstruction. Given the symbolic state sequence $\{s_1,\ldots,s_N\}$, NESYCR abduces a set of symbolic action operators $\mathcal{A}$ that capture the causal transitions between consecutive states. For each state pair (s_t, s_{t+1}), the VLM Ψ predicts an action operator $a_t = \Psi(s_t, s_{t+1})$ whose effects correspond to the state difference $s_{t+1} \backslash s_t$, and whose preconditions hold in s_t. Each a_t is represented as a tuple $(name, pre, eff)$ and appended to $\mathcal{A}$. To verify that $\mathcal{A}$ is consistent with the symbolic state sequence, the symbolic tool Φ performs forward simulation by applying each a_t to s_t and ensuring that the resulting symbolic state satisfies s_{t+1} at every step.

$$\big(\forall t \in [1, N-1],\ \Phi(s_t, a_t) \models s_{t+1}\big) \Rightarrow \text{Verified}(\mathcal{W}) \tag{4}$$

If this condition holds over the entire trajectory, $\mathcal{W}$ is constructed and verified, and the predicted action sequence is adopted as demonstrated procedure $\pi = \{a_1,\ldots,a_{N-1}\}$.

3.2. Neurosymbolic counterfactual adaptation

Based on the symbolic world model, NESYCR performs counterfactual adaptation in a neurosymbolic manner, identifying causal inconsistencies induced by target-domain constraints and revising the demonstrated procedure to restore causal consistency. As shown in Figure 3, given an observation from the deployment domain, the VLM generates a target symbolic state that serves as a counterfactual configuration by intervening on variables reflecting the target-domain conditions. Using this counterfactual state, the symbolic tool performs forward simulation along the demonstrated procedure to identify actions whose preconditions

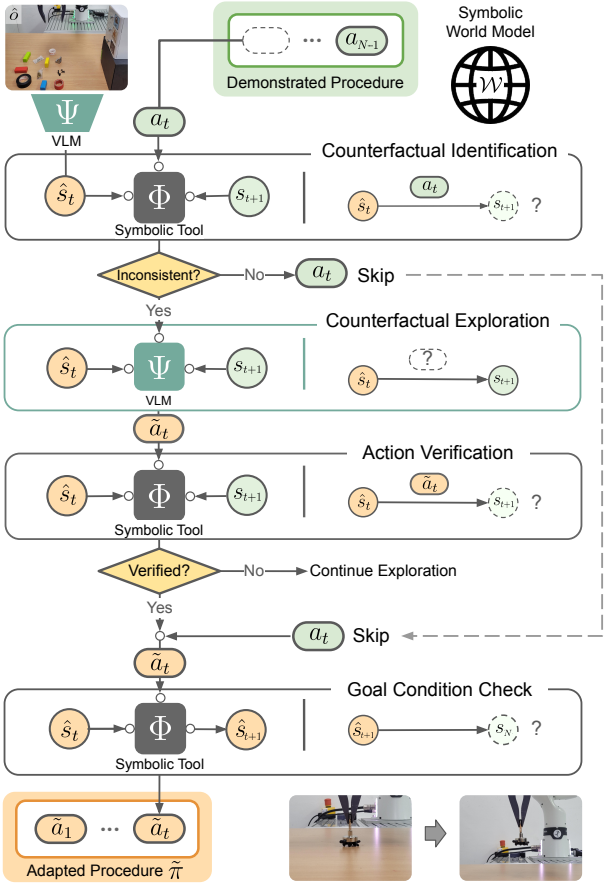


Figure 3. Neurosymbolic counterfactual adaptation

are inconsistent with the counterfactual setting, thereby revealing cross-domain inconsistencies that hinder execution. To resolve these conflicts, the VLM abduces alternative procedure steps through the insertion or removal of actions, while the symbolic tool iteratively verifies their logical consistency. By iterating this exploration, the adapted procedure is refined into a deployment-grounded task specification, from which an executable code policy is synthesized.

Counterfactuals identification. As the first step of neurosymbolic counterfactual adaptation, NESYCR identifies causal inconsistencies in the demonstrated procedure through forward simulation under counterfactual conditions. The VLM Ψ generates a counterfactual state $\hat{s}_1$ that reflects the target observation, while the symbolic tool Φ simulates each action a_t in the demonstrated procedure to estimate its outcome in the deployment domain.

$$\hat{s}_{t+1} = \Phi(\hat{s}_t, a_t), \quad t = 1, \ldots, N-1 \tag{5}$$

An action is regarded as inconsistent when its preconditions are not satisfied in the current counterfactual state or when its effects fail to reproduce the expected predicates in the

corresponding next state s_{t+1}.

$$\text{Inconsistent}(a_t) \Leftrightarrow \big(\mathit{pre}(a_t) \not\subseteq \hat{s}_t\big) \vee \big(\mathit{eff}(a_t) \not\subseteq s_{t+1}\big) \quad (6)$$

Such inconsistencies in the procedure must be resolved to restore entire causal consistency in the deployment domain.

Counterfactual exploration. To resolve these inconsistencies, NESYCR performs counterfactual exploration within the symbolic state space, grounding the task procedure through additive and subtractive modifications. For each inconsistent action identified, the VLM Ψ proposes alternative actions whose effects restore the violated preconditions of the subsequent valid action a_{t+1}. If no such alternative is applicable or the action becomes redundant to the task objective, it is removed from the procedure. Let the action revised through counterfactual exploration be

$$\tilde{a}_t = \begin{cases} \Psi(\hat{s}_t, a_t; s_{t+1}), & \text{if Inconsistent}(a_t), \\ a_t, & \text{otherwise.} \end{cases} \quad (7)$$

The adapted procedure $\tilde{\pi}$, derived from Eq. (7), restores procedural compatibility in deployment domain by satisfying

$$\forall t \in [1, N-1], \quad \Phi(\hat{s}_t, \tilde{a}_t) = \hat{s}_{t+1}, \quad \text{until } \hat{s}_{t+1} \models s_N. \quad (8)$$

Once $\tilde{\pi}$ reaches the goal condition specified by s_N, Ψ translates $\tilde{\pi}$ into a code policy $\pi_\theta = \Psi(\tilde{\pi})$, ensuring that it remains grounded in the deployment domain. Algorithm 1 summarizes the overall process of NESYCR.

Algorithm 1 NESYCR Framework

Demonstration $\mathcal{D} = (\{o_t\}_{t=1}^N, \ell)$, Target Observation $\hat{o}$
VLM Ψ, Symbolic Tool Φ, Symbolic Action Set $\mathcal{A} = \{\}$

/ Symbolic world model construction */*
Get $\mathcal{Q}, \mathcal{P}$ and $\{s_t\}_{t=1}^N$ via Eq. (3)
for $t = 1, \ldots, N-1$ **do**
  Get grounded action $a_t = \Psi(s_t, s_{t+1})$
  Verify a_t using Φ; otherwise repredict a_t *cf.* Eq. (4)
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{a_t\}$
end for
Get verified symbolic world model $\mathcal{W} = (\mathcal{Q}, \mathcal{P}, \mathcal{A}, \Phi)$
/ Neurosymbolic counterfactual adaptation */*
Initialize adapted procedure $\tilde{\pi} = [\,]$, $t \leftarrow 0$
repeat
  $t \leftarrow t+1$; Get counterfactual state $\hat{s}_t = \Psi(s_t; \hat{o})$
  Do forward simulation using Eq. (5)
  if Inconsistent(a_t) **then**
    Get interpolated action $\tilde{a}_t = \Psi(\hat{s}_t, a_t; s_{t+1})$
  else
    $\tilde{a}_t = a_t$
  end if *cf.* Eq. (6) & Eq. (7)
  Verify $\tilde{a}_t$ using Φ and get $\hat{s}_{t+1}$ via Eq. (8)
  Append $\tilde{a}_t$ to $\tilde{\pi}$
until $\hat{s}_{t+1} \models s_N$
Synthesize code policy $\pi_\theta = \Psi(\tilde{\pi})$

4. Evaluation

We evaluate NESYCR through experiments designed to address the following questions: **(Q1)** How does NESYCR perform compared to existing baselines in cross-domain demo-to-code settings? **(Q2)** How robust is NESYCR to increasing domain gaps between the demonstration and deployment? **(Q3)** How does NESYCR respond as the complexity gap between the demonstration and deployment tasks grows? **(Q4)** What is the contribution of each component of NESYCR?

4.1. Experiment setting

Cross-domain settings. Our cross-domain settings are defined by domain factors that introduce perceptual and physical variations between the demonstration and deployment domains. We consider five such factors that induce procedural differences in task execution: (1) *Obstruction* introduces interfering objects that require additional resolving steps. (2) *Object affordance* alters object states and inter-object relations, yielding new affordances and relational dependencies. (3) *Kinematic configuration* changes the robot's joint structure, affecting its reachable workspace and motion constraints. (4) *Gripper type* modifies the end-effector design, altering feasible grasp actions and contact affordances. (5) *Combination* jointly applies multiple domain factors, ranging from environmental factors (i.e., (1)-(2)) to embodiment factors (i.e., (3)-(4)), to create diverse and complex cross-domain scenarios. For demonstrations, we use data collected directly from both simulated and real environments, including robot executions and human-performed instructional videos. For deployment, simulated experiments are conducted in Genesis [70], a physics-based general-purpose robotics platform, while real-world evaluations are performed on a 7-DoF Franka Emika Research 3.

Benchmark tasks. The deployment setting comprises long-horizon manipulation scenarios (up to 116 API calls per scenario) focused on table organization and object rearrangement [31, 59]. Each scenario is composed of primitive subtasks (e.g., *pick&place*, *sweep*, *rotate*, *slide*), forming procedurally compositional manipulation sequences. This compositional structure allows systematic control over task complexity. For evaluation, scenarios are categorized into 3 complexity levels—*Low*, *Medium*, and *High*—according to the number, diversity, and dependency depth of subtasks. Higher levels correspond to longer action horizons and stronger inter-subtask dependencies. A total of 440 scenarios are categorized into three complexity levels: 160 for

Method	*Low-Complexity*			*Medium-Complexity*			*High-Complexity*		
	SR	GC	PD	SR	GC	PD	SR	GC	PD
Cross-domain factor: *Obstruction* and *Object affordance*									
Demo2Code	26.67±5.76	55.00±4.24	5.00±3.42	25.00±5.64	51.11±4.37	10.67±5.11	22.50±6.69	61.25±4.38	6.94±2.20
GPT4V-Robotics	71.67±5.87	82.50±3.91	29.15±5.34	41.67±6.42	68.89±4.11	43.18±5.09	20.00±6.41	57.50±4.49	35.94±10.68
Critic-V	45.00±6.48	65.83±4.52	10.37±4.58	35.00±6.21	64.44±4.18	24.97±4.95	15.00±5.72	58.75±3.96	45.83±9.50
MoReVQA	53.33±6.49	71.67±4.35	32.29±5.17	43.33±6.45	75.00±3.23	25.17±6.07	27.50±7.15	73.75±3.22	40.91±8.10
Statler	61.67±6.33	75.83±4.37	38.92±4.86	41.67±6.42	71.67±3.62	49.07±5.95	5.00±3.49	60.62±3.09	84.38±15.62
LLM-DM	51.67±6.51	67.50±4.87	66.02±4.19	20.00±5.21	47.78±4.37	77.13±3.21	12.50±5.30	53.12±4.49	91.25±5.45
NESYCR	**86.67**±4.43	**92.50**±2.61	1.92±1.35	**75.00**±5.64	**90.56**±2.25	10.57±2.22	**60.00**±7.84	**83.12**±3.94	12.50±2.91
Cross-domain factor: *Kinematic configuration* and *Gripper type*									
Demo2Code	33.33±6.14	36.67±6.04	0.00±0.00	25.00±5.64	28.33±5.70	0.00±0.00	20.00±6.41	20.00±6.41	7.81±2.29
GPT4V-Robotics	60.00±6.38	68.33±5.44	17.22±4.91	56.67±6.45	81.67±3.01	38.86±4.38	30.00±7.34	76.88±3.40	41.15±9.50
Critic-V	36.67±6.27	54.17±5.22	0.00±0.00	25.00±5.64	59.44±3.81	8.00±5.45	22.50±6.69	66.25±3.96	8.33±2.08
MoReVQA	48.33±6.51	63.33±5.16	13.10±5.70	31.67±6.06	70.56±3.08	46.43±6.43	25.00±6.93	69.38±3.52	20.00±7.95
Statler	68.33±6.06	74.17±5.25	29.43±5.14	50.00±6.51	70.56±4.35	53.35±4.31	42.50±7.92	74.38±4.24	50.00±5.60
LLM-DM	53.33±6.49	69.17±4.77	58.33±5.32	35.00±6.21	62.22±4.15	81.69±3.70	27.50±7.15	67.50±4.12	76.14±9.79
NESYCR	**93.33**±3.25	**96.67**±1.62	0.00±0.00	**78.33**±5.36	**85.00**±4.15	5.11±2.47	**52.50**±8.00	**78.12**±4.22	10.12±3.18
Cross-domain factor: *Combination* (*Obstruction*, *Object affordance*, *Kinematic configuration*, and *Gripper type*)									
Demo2Code	30.00±7.34	46.25±6.55	0.00±0.00	20.00±6.41	48.33±5.96	5.56±3.64	7.50±4.22	28.75±5.77	12.50±6.25
GPT4V-Robotics	55.00±7.97	70.00±5.88	15.76±5.80	35.00±7.64	69.17±4.53	32.38±7.25	15.00±5.72	63.12±3.80	36.46±13.35
Critic-V	40.00±7.84	60.00±5.99	5.00±3.42	37.50±7.75	69.17±4.68	5.93±3.41	27.50±7.15	66.25±4.44	16.48±4.15
MoReVQA	55.00±7.97	72.50±5.36	14.85±4.88	45.00±7.97	68.33±5.46	48.49±8.61	25.00±6.93	62.50±4.56	25.62±6.81
Statler	67.50±7.50	81.25±4.63	30.12±5.19	52.50±8.00	82.50±3.15	64.42±4.22	32.50±7.50	73.12±4.04	61.06±5.26
LLM-DM	32.50±7.50	56.25±5.71	82.05±3.80	22.50±6.69	60.00±4.65	80.56±2.65	10.00±4.80	46.88±4.92	81.25±3.61
NESYCR	**80.00**±6.41	**88.75**±3.79	2.81±1.97	**65.00**±7.64	**83.33**±4.30	6.60±2.53	**47.50**±8.00	**75.00**±4.48	17.11±3.16

Table 1. Performance on cross-domain demo-to-code tasks using simulation-based demonstration and deployment

Low, 160 for *Medium*, and 120 for *High*.

Baselines. We implement six state-of-the-art baselines for video-instructed robotic programming, grouped into three: (1) *VLM-based code policy synthesis*, which generates task specifications from demonstrations and synthesizes code policies from the generated specifications, represented by Demo2Code [58], (2) *VLM-based reasoning*, which produces target-domain task specifications through the reasoning capabilities of VLMs, including GPT4V-Robotics [55], Critic-V [69], and MoReVQA [41], and (3) *World-model-based approaches*, which construct LLM-based or neurosymbolic world models that support the generation of target task specifications, including Statler [66] and LLM-DM [21]. The baselines in (2) and (3) are not designed for code policy synthesis; in our evaluation, we apply their adaptation mechanisms in task specification generation, from which code policies are synthesized.

Evaluation metrics. To evaluate the objectives in Eq. (1), we use several metrics. **Success Rate (SR)** is the percentage of tasks completed in full, with a task counted as successful only when all subtasks are achieved. **Goal Condition (GC)** measures the proportion of success conditions achieved, reflecting the degree of subtask completion [49]. **Procedure Deviation (PD)** quantifies the alignment between the adapted and demonstrated procedures using a success-weighted, length-normalized edit distance over their subtask-achievement sequences [19, 28].

4.2. Main result

To address **(Q1)**, we evaluate NESYCR in a simulated cross-domain environment where domain gaps are systematically induced through controlled variations of our domain factors. This setting allows fine-grained and reproducible analysis of code policy performance under different environmental and embodiment changes. As shown in Table 1, we compare NESYCR with six state-of-the-art baselines across environmental, embodiment, and combined cross-domain scenarios. NESYCR consistently outperforms all baselines in both SR and GC across every domain factor. On average, it achieves a 27.73% higher SR and 15.16% higher GC than GPT4V-Robotics, the strongest *VLM-based reasoning* baseline. Similarly, NESYCR improves SR by 24.77% and GC by 13.20% over Statler, the strongest *world-model-based* baseline. These results demonstrate that NESYCR provides substantially more robust cross-domain adaptation than existing methods.

The baselines in the *VLM-based reasoning* category, including GPT4V-Robotics, Critic-V, and MoReVQA, struggle to maintain consistent task specifications under domain shifts. In particular, erroneous feedback generation in Critic-V and error propagation across multi-stage reasoning in MoReVQA lead to substantial performance degradation, causing 40.91% and 32.95% drops in SR, respectively. Al-

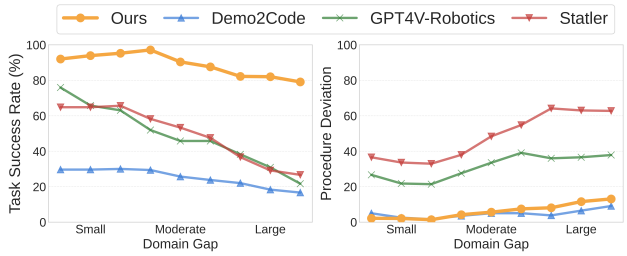
though Statler leverages symbolic state representations and supports VLM-guided simulative planning, its lack of symbolic tool integration and dependence on replanning from scratch result in substantial performance degradation, especially in high-complexity scenarios. LLM-DM, while employing symbolic tools, depends on constructing complete domain knowledge from a single demonstration, which frequently produces invalid or illogical plans, resulting in a 41.82% drop in SR. Demo2Code, which lacks a dedicated adaptation mechanism, exhibits a 49.09% drop in SR compared to NESYCR. In terms of PD, however, NESYCR achieves a comparable score, with only a 1.73 average increase across domain factors. This is because Demo2Code tends to imitate the demonstration without accounting for the deployment domain, yielding procedurally similar but frequently unreliable code policies.

Method	*Medium-Complexity*		
	SR	GC	PD
Cross-domain factor: *Combination*			
Demo2Code	0.00±0.00	25.00±3.57	–
GPT4V-Robotics	50.00±18.90	75.00±9.64	0.00±0.00
Statler	50.00±18.90	67.86±15.45	42.86±24.74
NESYCR	**87.50**±12.50	**98.21**±1.79	24.49±11.54

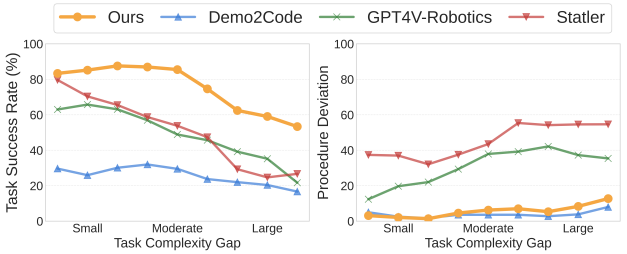
Table 2. Performance on cross-domain demo-to-code tasks using real-world demonstrations and deployment

Furthermore, we evaluate NESYCR in real-world deployment using a physical robot, with results presented in Table 2. Using demonstrations captured from human video recordings, we evaluate performance on the same task executed under an obstructive domain gap using a Franka Research 3 arm. In the demonstration, a human organizes objects into two parallel drawers. In the deployment domain, these drawers sit at a perpendicular angle, creating a mutual-interference constraint: one drawer cannot open unless the other is closed. Because the objects are arranged in stacks requiring ordered handling, the agent must alternate drawer operations, correctly inferring and maintaining the interference constraint throughout the procedure. NESYCR effectively adapts to domain changes, achieving an 87.50% higher SR than Demo2Code and a 37.50% higher SR than both Statler and GPT4V-Robotics. GPT4V-Robotics often overlooks the constraints imposed by the perpendicular orientation and the stacked-object setting, which require alternating drawer operations; instead, it attempts to open both drawers at once. Statler maintains a world state but, lacking any explicit mechanism for enforcing this constraint, intermittently attempts to open both drawers at once throughout the procedure. In contrast, NESYCR explicitly encodes the interference as a symbolic precondition and uses symbolic tools to verify its satisfaction throughout the entire procedure. This enables an algorithmic process for detecting and resolving incompatible procedural steps regardless of horizon length, supporting reliable real-world control.

4.3. Analysis and ablation



(a) Comparison of SR, PD across different levels of domain gap



(b) Comparison of SR, PD across different levels of task complexity gap

Figure 4. Analysis on (a) domain gap and (b) task complexity gap

Analysis on domain gap. To address **(Q2)**, we examine the robustness of NESYCR as the domain gap between demonstration and deployment increases in Figure 4a. We amplify variations in environmental factors while keeping task complexity fixed, creating increasingly challenging cross-domain evaluation settings. As the gap widens, the number of obstructing objects grows, requiring additional alternative actions to maintain procedural compatibility. For all gap levels, NESYCR explores and revises procedural steps more accurately than the baselines, achieving higher SR and lower PD. However, VLM-based baselines struggle to infer verified alternative action operators required to restore procedural compatibility, leading to a sharp performance drop from the Moderate to the High gap level.

Analysis on task complexity gap. To address **(Q3)**, we evaluate the performance of NESYCR under increasing task complexity gaps between the demonstration and deployment domains, as shown in Figure 4b. Task complexity is controlled by increasing the number of subtasks and the depth of their dependencies while fixing a domain factor such as *Obstruction*. When the complexity gap is moderate (e.g., up to 3 additional subtasks), NESYCR effectively adapts the procedure. Yet, as the gap widens, the deployment-domain task scenario diverges from the demonstrated one, causing a pronounced performance drop as the demonstration no longer provides sufficient guidance and a fundamentally new demonstration becomes necessary.

Method	*Medium-Complexity*		
	SR	GC	PD
Cross-domain factor: *Combination*			
NESYCR	68.42±7.64	84.21±4.49	6.60±2.53
NESYCR w/o Eq. (8)	50.00±8.22	78.07±4.03	14.65±3.20
NESYCR w/o Eq. (6)	47.37±8.21	77.19±4.00	8.55±2.61
NESYCR w/o Eq. (6) & Eq. (8)	39.47±8.04	74.56±4.25	9.63±3.90
NESYCR w/o Eq. (4)	34.21±7.80	67.54±4.79	6.84±3.89

Table 3. Ablation study of NESYCR

Ablation study. To address **(Q4)**, we conduct an ablation study to assess the contribution of each component within NESYCR. As shown in Table 3, removing either the neurosymbolic counterfactual adaptation or the symbolic world model leads to a significant drop in SR and an increase in PD, underscoring the importance of these components for effective cross-domain demo-to-code performance. Specifically, removing the verification of alternative actions (w/o Eq. (8)) results in an 18.42% drop in SR, as the lack of verification allows the VLM to generate semantically plausible but logically inconsistent actions. Removing the counterfactual identification (Eq. (6)) prevents the framework from obtaining feedback from the symbolic tool on which procedural steps require revision, leading to a 21.05% drop in SR. When both components are removed (w/o Eq. (8) & Eq. (6)), SR decreases by 28.95%. Removing the symbolic world model (w/o Eq. (4)) causes the most severe degradation, as the framework can no longer perform symbolic tool-based verification, resulting in the lowest SR of 34.21%.

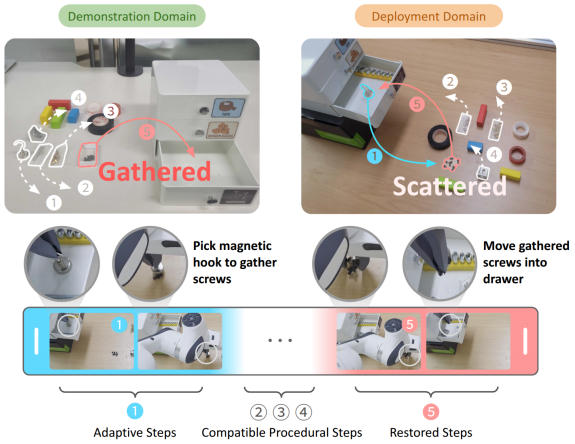


Figure 5. Visualization of a cross-domain demo-to-code task

Visualization of adaptation. To elaborate on the example in Figure 1, we visualize the demonstration and the adapted procedure in the real-world deployment scenario. As shown in Figure 5, the demonstration domain features a human performing a drawer-organizing task with a magnetic hook and a box of screws on the table. In the deployment domain, however, the human hand is replaced by a robotic gripper, the magnetic hook is already inside the drawer, and the screws are scattered across the surface. First, NESYCR detects that the magnetic hook already occupies the target position in the deployment domain and removes the redundant steps related to placing it. The VLM then repurposes the hook as an intermediate tool for aggregating the scattered screws, introducing actions for retrieving the hook and placing it over them. These alternative steps, however, can be obstructed by objects that must also be placed in the drawer. NESYCR resolves this potential failure by reordering the steps, performing the retrieval and aggregation steps before the organizing steps and producing a final procedure that restores compatibility with the deployment domain.

5. Related work

Foundation models provide commonsense knowledge and reasoning capabilities that enable generalized embodied control, leveraging pretrained language and vision knowledge to generate executable control code from instructions or demonstrations [36, 54, 58, 61]. Learning from demonstrations is common for embodied agents, but policies trained via behavioral cloning or inverse reinforcement learning struggle to generalize under perceptual and physical changes [5, 30, 40, 47]. Prior work has explored feature alignment, state-transition matching, and video-based imitation [10, 44, 67, 68]. Neurosymbolic approaches combine neural adaptability with symbolic reasoning to improve correctness and interpretability in embodied planning, with recent methods using LLMs or VLMs to derive symbolic representations that are verified by symbolic solvers [13, 37, 50, 53], whereas our work integrates such neurosymbolic reasoning with counterfactual inference to revise demonstrated procedures for cross-domain adaptation. Further discussion of related work is in Appendix A.

6. Conclusion

In this work, we presented NESYCR, a neurosymbolic framework for cross-domain code synthesis in video-instructed robotic programming grounded in counterfactual reasoning. By constructing a verifiable symbolic world model and performing neurosymbolic counterfactual adaptation, NESYCR converts video demonstrations into executable code policies that remain valid under perceptual and physical variations. Extensive experiments in simulation and real-world settings show that NESYCR consistently outperforms existing baselines, maintaining higher task success and procedural consistency even as domain gaps and task complexity increase. As shown in Figure 4, when the task complexity gap between the demonstration and deployment domains becomes large, the target task can no longer be solved via demonstration-based counterfactual reasoning alone. Future work will extend this boundary through causal re-grounding, enabling NESYCR to infer new causal relations and remain robust in novel contexts.

Acknowledgement

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT), RS-2022-II220043, Adaptive Personality for Intelligent Agents, RS-2022-II221045, Self-directed multi-modal Intelligence for solving unknown, open domain problems, RS-2025-02218768, Accelerated Insight Reasoning via Continual Learning, and RS-2025-25442569, AI Star Fellowship Support Program (Sungkyunkwan Univ.) RS-2019-II190421, Artificial Intelligence Graduate School Program (Sungkyunkwan University)), the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2026-25474409), IITP-ITRC (Information Technology Research Center) grant funded by the Korea government (MSIT) (IITP-2025-RS-2024-00437633, 10%), IITP-ICT Creative Consilience Program grant funded by the Korea government (MSIT) (IITP-2026-RS-2020-II201821), and by Samsung Electronics Co., Ltd.

References

[1] Constructions Aeronautiques, Adele Howe, Craig Knoblock, ISI Drew McDermott, Ashwin Ram, Manuela Veloso, Daniel Weld, David Wilkins Sri, Anthony Barrett, Dave Christianson, et al. Pddl— the planning domain definition language. *Technical Report, Tech. Rep.*, 1998. 13

[2] Sudhir Agarwal, Anu Sreepathy, David H Alonso, and Prarit Lamba. Llm+ reasoning+ planning for supporting incomplete user queries in presence of apis. *arXiv preprint arXiv:2405.12433*, 2024. 13

[3] Sanghyun Ahn, Wonje Choi, Junyong Lee, Jinwoo Park, and Honguk Woo. Towards reliable code-as-policies: A neuro-symbolic framework for embodied task planning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 13

[4] Montserrat Gonzalez Arenas, Ted Xiao, Sumeet Singh, Vidhi Jain, Allen Ren, Quan Vuong, Jake Varley, Alexander Herzog, Isabel Leal, Sean Kirmani, et al. How to prompt your robot: A promptbook for manipulation skills with code as policies. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024. 13

[5] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 2009. 8, 13

[6] Ashay Athalye, Nishanth Kumar, Tom Silver, Yichao Liang, Jiuguang Wang, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. From pixels to predicates: Learning symbolic world models via pretrained vision-language models. *arXiv preprint arXiv:2501.00296*, 2024. 13

[7] Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Proceedings of the 6th Conference on Robot Learning*, 2023. 1, 13

[8] Kaylee Burns, Ajinkya Jain, Keegan Go, Fei Xia, Michael Stark, Stefan Schaal, and Karol Hausman. Genchip: generating robot policy code for high-precision and contact-rich manipulation tasks. In *Proceedings of the 37th IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024. 13

[9] Yongchao Chen, Yilun Hao, Yang Zhang, and Chuchu Fan. Code-as-symbolic-planner: Foundation model-based robot planning via symbolic code generation. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025. 13

[10] Sungho Choi, Seungyul Han, Woojun Kim, Jongseong Chae, Whiyoung Jung, and Youngchul Sung. Domain adaptive imitation learning with visual observation. *Advances in Neural Information Processing Systems*, 2023. 8, 13

[11] Wonje Choi, Woo Kyung Kim, SeungHyun Kim, and Honguk Woo. Efficient policy adaptation with contrastive prompt ensemble for embodied agents. *arXiv preprint arXiv:2412.11484*, 2024. 1, 13

[12] Wonje Choi, Jooyoung Kim, and Honguk Woo. Nesypr: Neurosymbolic proceduralization for efficient embodied reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 13

[13] Wonje Choi, Jinwoo Park, Sanghyun Ahn, Daehee Lee, and Honguk Woo. Nesyc: A neuro-symbolic continual learner for complex embodied tasks in open domains. *arXiv preprint arXiv:2503.00870*, 2025. 8

[14] Cristina Cornelio and Mohammed Diab. Recover: A neuro-symbolic framework for failure detection and recovery. *arXiv preprint arXiv:2404.00756*, 2024. 13

[15] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model. In *arXiv preprint arXiv:2303.03378*, 2023. 1, 13

[16] Arnaud Fickinger, Samuel Cohen, Stuart Russell, and Brandon Amos. Cross-domain imitation learning via optimal transport. *arXiv preprint arXiv:2110.03684*, 2021. 1, 13

[17] Richard E Fikes and Nils J Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 1971. 3

[18] Maria Fox and Derek Long. Pddl2. 1: An extension to pddl for expressing temporal planning domains. *Journal of artificial intelligence research*, 2003. 4

[19] Maria Fox, Alfonso Gerevini, Derek Long, Ivan Serina, et al. Plan stability: Replanning versus plan repair. In *ICAPS*, 2006. 6, 18

[20] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. Multi-shot asp solving with clingo. *Theory and Practice of Logic Programming*, 2019. 13

[21] Lin Guan, Karthik Valmeekam, Sarath Sreedharan, and Subbarao Kambhampati. Leveraging pre-trained large language models to construct and utilize world models for model-based task planning. *Advances in Neural Information Processing Systems*, 2023. 6, 24

[22] Haoran He, Chenjia Bai, Ling Pan, Weinan Zhang, Bin Zhao, and Xuelong Li. Learning an actionable discrete diffusion policy via large-scale actionless video pre-training. *Advances in Neural Information Processing Systems*, 2024. 3

[23] Richard Howey, Derek Long, and Maria Fox. Val: Automatic plan validation, continuous effects and mixed initiative planning using pddl. In *16th IEEE International Conference on Tools with Artificial Intelligence*, 2004. 4

[24] Siyuan Huang, Zhengkai Jiang, Hao Dong, Yu Qiao, Peng Gao, and Hongsheng Li. Instruct2act: Mapping multi-modality instructions to robotic actions with large language model. *arXiv preprint arXiv:2305.11176*, 2023. 1, 13

[25] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022. 1, 13

[26] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023. 1, 13

[27] William Huey, Huaxiaoyue Wang, Anne Wu, Yoav Artzi, and Sanjiban Choudhury. Imitation learning from a single temporally misaligned video. *arXiv preprint arXiv:2502.05397*, 2025. 3

[28] Mert Inan, Aishwarya Padmakumar, Spandana Gella, Patrick Lange, and Dilek Hakkani-Tur. Multimodal contextualized plan prediction for embodied task completion. *arXiv preprint arXiv:2305.06485*, 2023. 6, 18

[29] Adam Ishay, Zhun Yang, and Joohyung Lee. Leveraging large language models to generate answer set programs. *arXiv preprint arXiv:2307.07699*, 2023. 13

[30] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, 2022. 8, 13

[31] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. In *Fortieth International Conference on Machine Learning*, 2023. 5, 15

[32] Byeonghwi Kim, Jinyeon Kim, Yuyeong Kim, Cheolhong Min, and Jonghyun Choi. Context-aware planning and environment-aware memory for instruction following embodied agents. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 13

[33] Bowen Li, Tom Silver, Sebastian Scherer, and Alexander Gray. Bilevel learning for bilevel planning. *arXiv preprint arXiv:2502.08697*, 2025. 13

[34] Rongjie Li, Songyang Zhang, Dahua Lin, Kai Chen, and Xuming He. From pixels to graphs: Open-vocabulary scene graph generation with vision-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024. 4

[35] Yin Li, Liangwei Wang, Shiyuan Piao, Boo-Ho Yang, Ziyue Li, Wei Zeng, and Fugee Tsung. Mccoder: streamlining motion control with llm-assisted code generation and rigorous verification. *arXiv preprint arXiv:2410.15154*, 2024. 13

[36] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. *arXiv preprint arXiv:2209.07753*, 2022. 1, 3, 8, 13

[37] Yichao Liang, Nishanth Kumar, Hao Tang, Adrian Weller, Joshua B Tenenbaum, Tom Silver, João F Henriques, and Kevin Ellis. Visualpredicator: Learning abstract world models with neuro-symbolic predicates for robot planning. *arXiv preprint arXiv:2410.23156*, 2024. 8, 13

[38] Xinrui Lin, Yangfan Wu, Huanyu Yang, Yu Zhang, Yanyong Zhang, and Jianmin Ji. Clmasp: Coupling large language models with answer set programming for robotic task planning. *arXiv preprint arXiv:2406.03367*, 2024. 13

[39] Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023. 13

[40] YuXuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *2018 IEEE international conference on robotics and automation (ICRA)*, 2018. 8, 13

[41] Juhong Min, Shyamal Buch, Arsha Nagrani, Minsu Cho, and Cordelia Schmid. Morevqa: Exploring modular reasoning models for video question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 6, 23

[42] Theo X Olausson, Alex Gu, Benjamin Lipkin, Cedegao E Zhang, Armando Solar-Lezama, Joshua B Tenenbaum, and Roger Levy. Linc: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. *arXiv preprint arXiv:2310.15164*, 2023. 13

[43] Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. *arXiv preprint arXiv:2305.12295*, 2023. 13

[44] Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *European Conference on Computer Vision*, 2022. 8, 13

[45] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017. 1, 13

[46] Dripta S Raychaudhuri, Sujoy Paul, Jeroen Vanbaar, and Amit K Roy-Chowdhury. Cross-domain imitation from observations. In *International conference on machine learning*, 2021. 1, 13

[47] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011. 8, 13

[48] Sangwoo Shin, Daehee Lee, Minjong Yoo, Woo Kyung Kim, and Honguk Woo. One-shot imitation in a non-stationary environment via multi-modal skill. In *International Conference on Machine Learning*, 2023. 13

[49] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020. 6, 17

[50] Tom Silver, Soham Dan, Kavitha Srinivas, Joshua B Tenenbaum, Leslie Kaelbling, and Michael Katz. Generalized planning in pddl domains with pretrained large language models. In *Proceedings of the AAAI conference on artificial intelligence*, 2024. 8, 13

[51] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the 19th IEEE/CVF International Conference on Computer Vision*, 2023. 1, 13

[52] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 2

[53] Hao Tang, Darren Key, and Kevin Ellis. Worldcoder, a model-based llm agent: Building world models by writing code and interacting with the environment. In *Advances in Neural Information Processing Systems*, 2024. 8, 13

[54] Sai Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. Chatgpt for robotics: Design principles and model abilities. 2023. *Published by Microsoft*, 2023. 1, 8, 13

[55] Naoki Wake, Atsushi Kanehira, Kazuhiro Sasabuchi, Jun Takamatsu, and Katsushi Ikeuchi. Gpt-4v (ision) for robotics: Multimodal task planning from human demonstration. *IEEE Robotics and Automation Letters*, 2024. 6, 23

[56] Beichen Wang, Juexiao Zhang, Shuwen Dong, Irving Fang, and Chen Feng. Vlm see, robot do: Human demo video to robot action plan via vision language model. *arXiv preprint arXiv:2410.08792*, 2024. 1, 13

[57] Lirui Wang, Yiyang Ling, Zhecheng Yuan, Mohit Shridhar, Chen Bao, Yuzhe Qin, Bailin Wang, Huazhe Xu, and Xiaolong Wang. Gensim: Generating robotic simulation tasks via large language models. *arXiv preprint arXiv:2310.01361*, 2023. 13

[58] Yuki Wang, Gonzalo Gonzalez-Pumariega, Yash Sharma, and Sanjiban Choudhury. Demo2code: From summarizing demonstrations to synthesizing code via extended chain-of-thought. *Advances in Neural Information Processing Systems*, 2023. 1, 3, 6, 8, 13, 23

[59] Yufei Wang, Zhou Xian, Feng Chen, Tsun-Hsuan Wang, Yian Wang, Katerina Fragkiadaki, Zackory Erickson, David Held, and Chuang Gan. Robogen: Towards unleashing infinite data for automated robot learning via generative simulation. *arXiv preprint arXiv:2311.01455*, 2023. 5, 15

[60] Zuoxu Wang, Zhijie Yan, Shufei Li, and Jihong Liu. Indvissgg: Vlm-based scene graph generation for industrial spatial intelligence. *Advanced Engineering Informatics*, 2025. 4

[61] Senwei Xie, Hongyu Wang, Zhanqi Xiao, Ruiping Wang, and Xilin Chen. Robotic programmer: Video instructed policy code generation for robotic manipulation. *arXiv preprint arXiv:2501.04268*, 2025. 1, 8, 13

[62] Dongil Yang, Minjin Kim, Sunghwan Mac Kim, Beongwoo Kwak, Minjun Park, Jinseok Hong, Woontack Woo, and Jinyoung Yeo. Llm meets scene graph: Can large language models understand and generate scene graphs? a benchmark and empirical study. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025. 28

[63] Yijun Yang, Tianyi Zhou, Kanxue Li, Dapeng Tao, Lusong Li, Li Shen, Xiaodong He, Jing Jiang, and Yuhui Shi. Embodied multi-modal agent trained by an llm from a parallel textworld. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024. 13

[64] Zhun Yang, Adam Ishay, and Joohyung Lee. Coupling large language models with logic programming for robust and general reasoning from text. *arXiv preprint arXiv:2307.07696*, 2023. 13

[65] Weirui Ye, Fangchen Liu, Zheng Ding, Yang Gao, Oleh Rybkin, and Pieter Abbeel. Video2policy: Scaling up manipulation tasks in simulation through internet videos. *arXiv preprint arXiv:2502.09886*, 2025. 1, 13

[66] Takuma Yoneda, Jiading Fang, Peng Li, Huanyu Zhang, Tianchong Jiang, Shengjie Lin, Ben Picker, David Yunis, Hongyuan Mei, and Matthew R Walter. Statler: State-maintaining language models for embodied reasoning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024. 2, 6, 24

[67] Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. Visual imitation made easy. In *Conference on Robot learning*, 2021. 8, 13

[68] Tianhe Yu, Chelsea Finn, Annie Xie, Sudeep Dasari, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. *arXiv preprint arXiv:1802.01557*, 2018. 8, 13

[69] Di Zhang, Jingdi Lei, Junxian Li, Xunzhi Wang, Yujie Liu, Zonglin Yang, Jiatong Li, Weida Wang, Suorong Yang, Jianbo Wu, et al. Critic-v: Vlm critics help catch vlm errors in multimodal reasoning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025. 6, 23

[70] Xian Zhou, Yiling Qiao, Zhenjia Xu, TH Wang, Z Chen, J Zheng, Z Xiong, Y Wang, M Zhang, P Ma, et al. Genesis: A generative and universal physics engine for robotics and beyond. *arXiv preprint arXiv:2401.01454*, 2024. 5, 13

[71] Wang Bill Zhu, Miaosen Chai, Ishika Singh, Robin Jia, and Jesse Thomason. Psalm-v: Automating symbolic planning in interactive visual environments with large language models. *arXiv preprint arXiv:2506.20097*, 2025. 13