PO-PDDL: Learning Symbolic POMDPs from Visual Demonstrations for Robot Planning Under Uncertainty

Wenjing Tang[1,2§], **Xuanjin Jin**[1,2], **Yuan Liu**[1,2], **Renming Huang**[1,2], **Cewu Lu**[1,2], **Panpan Cai**[1,2†]
[1]Shanghai Jiao Tong University [2]Shanghai Innovation Institute

***Abstract*—Real-world robot task planning must handle stochastic execution and partial observability, but building POMDP models for visual robot domains remains difficult. We introduce PO-PDDL, a symbolic POMDP formulation that preserves the relational structure and LLM-friendly syntax of PDDL while adding stochastic effects, observation models, and beliefs. We further present a demonstration-driven pipeline that learns reusable PO-PDDL domains from real-robot execution videos by reconstructing latent symbolic trajectories, identifying partial observability from state-observation mismatches, and estimating transition and observation models. The learned domains support online belief-space planning under perception and execution uncertainty. Experiments on real-world long-horizon manipulation tasks show that our method consistently outperforms existing PDDL and POMDP model-learning approaches while achieving substantially lower planning cost.**[1]

I. Introduction

Real-world robot task planning must operate under uncertainty from both perception and action execution. Robots face occlusions, limited viewpoints, and perception errors, while low-level execution modules such as motion planners and vision-language-action (VLA) policies [1, 8] can produce stochastic outcomes. These factors naturally call for POMDP planning, but specifying POMDP transition, observation, reward, and belief models for real robot domains requires substantial expertise. Learning such models directly from visual demonstrations is even more challenging, because the demonstrations provide raw videos rather than explicit symbolic states.

Existing approaches do not fully address this setting. PDDL domain-learning methods such as UniDomain [20] support structured symbolic reasoning, but they mainly assume deterministic and fully observable environments. POMDPX [5], RDDL [16], and programmatic POMDP models can express uncertainty, but they are difficult to construct from visual robot demonstrations and are poorly aligned with LLM generation. Recent LLM-based POMDP methods [10] reduce manual modeling effort, but are typically evaluated in simplified structured domains and struggle with visual state abstraction, stochastic robot skills, and hidden task-relevant variables.

We bridge this gap with **PO-PDDL**, a PDDL-style symbolic formulation for POMDPs. PO-PDDL keeps the relational syntax that makes PDDL interpretable and LLM-friendly, while adding explicit stochastic transitions, observation rules, and initial beliefs. Building on this formulation, we develop a learning pipeline that constructs a reusable PO-PDDL domain from visual demonstrations of robot policy executions. The pipeline induces predicate and operator schemas, reconstructs hidden symbolic trajectories, detects partial observability by comparing reconstructed states with visual observations, and learns symbolic transition and observation models.

We evaluate the learned domains on real-world robot tasks with long horizons, action constraints, stochastic skill execution, and hidden states. Results show that our method achieves 100% task success and the highest cumulative reward across all task levels, substantially outperforming direct VLM planning and existing model-learning baselines.

In summary, we make the following contributions: (i) we propose PO-PDDL, a symbolic formulation that unifies PDDL-style planning with POMDP uncertainty modeling; (ii) we develop a demonstration-driven pipeline for learning PO-PDDL domains from real-robot videos, including stochastic transitions and observation models; and (iii) we demonstrate that the learned domains enable efficient belief-space planning for long-horizon robot tasks under both perception and execution uncertainty.

II. Related Work

POMDP planning requires explicit transition, observation, reward, and belief models. POMDPX [5] and RDDL [16] provide expressive representations for stochastic planning, but they are not well matched to LLM-based model synthesis from demonstrations. In contrast, PDDL-style representations are compact, symbolic, and interpretable, which makes them suitable for LLM generation and revision. Existing PDDL learning and repair methods [20, 11, 19, 2, 7, 12, 22] reduce manual domain engineering, but mainly focus on deterministic or fully observable models.

LLM and VLM planners have been used to reason under uncertainty through prompting, hypothesis generation, reinforcement learning, or tree search [17, 15, 13, 9, 18, 24]. However, their beliefs and dynamics are often implicit, which leads to

[§]Wenjing Tang is the visiting students at Shanghai Innovation Institute.
[†]Corresponding author: cai_panpan@sjtu.edu.cn
[1]Project page: https://po-pddl.github.io.

high online inference cost and limited solver-level structure. Other work handles partial observability through deterministic replanning [6], coarse symbolic beliefs [23], stochastic action learning [4], or LLM-generated POMDP programs [3]. Our method differs by learning reusable probabilistic transition and observation models directly from real robot execution videos and using them for closed-loop belief-space planning.

III. PO-PDDL Overview

PDDL and POMDPs provide complementary strengths. PDDL represents planning domains with typed objects, predicates, and operators, making it interpretable and easy to manipulate with LLMs. However, classical PDDL assumes that the symbolic state is directly observed and that operator effects are deterministic. POMDPs model hidden states, stochastic transitions, observations, rewards, and beliefs, but are usually specified in low-level mathematical or programmatic forms that are difficult to learn from demonstrations.

PO-PDDL combines these strengths in a symbolic POMDP interface. A PO-PDDL domain specifies predicate schemas for latent symbolic state, observable schemas for perceptual outcomes, stochastic operators for action dynamics, and symbolic observation rules for perception. A problem file specifies task objects, an initial belief over complete symbolic states, and a goal condition. After grounding, predicates induce the POMDP state space, operators induce the action space, transition model, and reward model, and observation rules induce the observation model. Thus, PO-PDDL preserves the domain/problem separation and relational structure of PDDL while supporting belief-space planning under stochastic execution and partial observability.

This design is important for learning from robot videos. Predicates provide a compact symbolic abstraction of task-relevant properties such as object locations, drawer states, and container contents. Observable schemas separate what the robot can visually perceive from what is truly latent in the world. Stochastic operators capture the fact that the same high-level skill can succeed, fail, or produce different side effects depending on the hidden state and low-level execution. By representing all of these components in a PDDL-like syntax, PO-PDDL enables LLMs and VLMs to generate, verify, and revise POMDP models in a structured form.

IV. Learning and Planning Pipeline

A PO-PDDL domain is tied to a specific scene and low-level execution stack, but is reusable across tasks in that environment. Each demonstration contains synchronized head- and wrist-camera videos, a task-level instruction, step-level subtask instructions, VLM-generated scene descriptions, and execution outcomes. The demonstrations are segmented into clips aligned with high-level subtasks. From this data, our goal is to learn a PO-PDDL domain containing predicate schemas, observable schemas, stochastic operators, and symbolic observation rules.

A. Fully Observable Domain Learning

The first stage learns the predicate vocabulary and stochastic operator model. An LLM induces state concepts from task instructions, subtask descriptions, and VLM scene descriptions, producing typed predicates that represent changing object properties and spatial relations. The same model abstracts subtask instructions into operator interfaces with names and typed parameters. Operators are classified as manipulation operators, which may change the environment, or active-perception operators, which are used to reveal hidden information.

Given these interfaces, the pipeline reconstructs hidden symbolic trajectories. For each demonstration, an LLM grounds the task goal and maps each subtask to a grounded operator. It then infers symbolic effects from the visual observations before and after execution. Because visual observations may be partially observable, we use forward-backward inference: the model proposes action effects from local evidence, infers an initial symbolic state that makes the full trajectory and final goal consistent, and revises the trajectory when inconsistencies remain. This produces latent state sequences that explain both the observed execution and the task outcome.

The reconstructed trajectories are used to estimate stochastic operators. Preconditions are extracted from predicate assignments that consistently hold before an operator is applied. Effects are clustered into distinct outcome modes, and each mode is assigned an empirical probability. Operator rewards are defined from execution cost and task-level success or failure conditions. A VLM-based review loop then checks representative clips for each operator effect mode and revises the symbolic interpretation when visual evidence contradicts the learned model. After convergence, the pipeline obtains a refined fully observable symbolic backbone that induces the POMDP state space, action space, transition model, and reward model.

B. Partially Observable Domain Learning

The second stage identifies which predicates are not reliably observed and learns the corresponding observation model. The key signal is a mismatch between reconstructed latent states and visual states parsed from scene descriptions. For each step, we prompt an LLM to assign each grounded predicate a visual value: true, false, or unknown. If a visually definite value contradicts the reconstructed latent value, the predicate is treated as a candidate partially observable predicate. A VLM rechecks the raw video to filter out mistakes caused by incorrect visual grounding or trajectory reconstruction. Predicates with verified contradictions are assigned observable schemas, while predicates without such schemas are treated as fully observable.

For each partially observable predicate, we estimate symbolic observation rules from paired latent-state and observation samples. Passive observations are collected from scene descriptions after manipulation actions. Active observations are collected from active-perception clips, where a VLM identifies the targeted predicate and its revealed value. The resulting samples estimate the likelihood of positive or negative observations conditioned on the hidden predicate value and the action type.

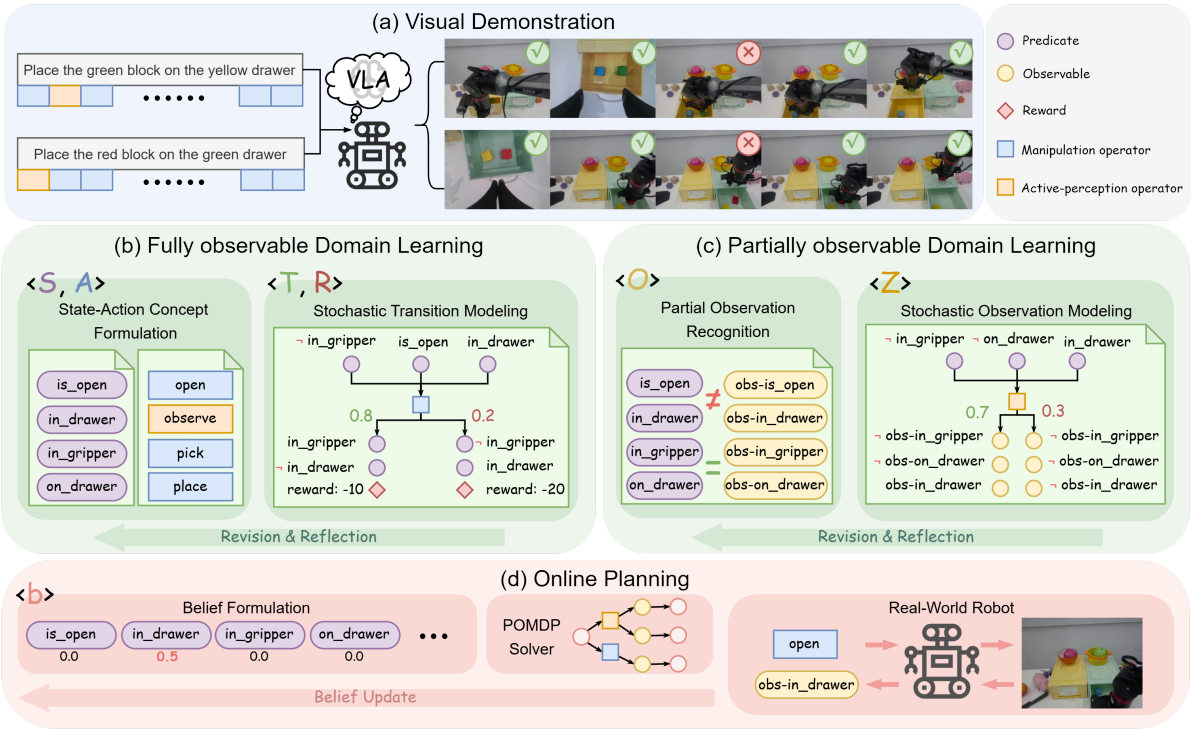


Fig. 1: Overview of our method. The pipeline learns a PO-PDDL domain from visual demonstrations and uses the learned model for online belief-space planning.

These rules form the PO-PDDL observation model and induce the POMDP observation function after grounding.

C. Problem Generation and Belief-Space Planning

At test time, the learned domain is combined with an initial scene observation and a task instruction. Fully observable predicates are grounded directly from the initial observation. For partially observable predicates, the system constructs a feasible set of complete symbolic states using structural constraints such as mutual exclusiveness among object locations. It then combines empirical priors from demonstrations with a uniform prior over feasible assignments to form the initial belief. The goal condition is generated from the task instruction by selecting relevant predicate schemas and enumerating semantically matching goal states.

The resulting problem induces a symbolic POMDP, which we solve using DESPOT [21]. During execution, the planner repeatedly selects an action from the current belief, sends it to the low-level policy, receives the execution outcome and visual observation, updates the belief using the learned transition and observation models, and replans. Most steps only require symbolic belief updates. VLM calls are concentrated on visual grounding and active-perception observations, which reduces online inference cost compared with direct VLM planning.

V. Experiments

We evaluate whether PO-PDDL can learn reusable symbolic POMDP models from robot videos and use them for effective planning under action constraints, stochastic execution, and uncertain observations. Results show that PO-PDDL consistently achieves the strongest planning performance across all tasks, substantially outperforming existing model-learning approaches and direct VLM planning.

A. Experimental Setup

We conduct experiments in a real-world tabletop scene, shown in Figure 2. The environment contains drawers, cups, blocks, and a movable black box. We collect 5 short-horizon demonstration task types with an average horizon of 3.4 steps and evaluate on 5 longer-horizon task types requiring compositional reasoning and online planning under uncertainty. Manipulation is executed by local VLA skills chained with motion planning, and perception is provided by a VLM operating on RGB observations. The VLA policy is based on the public release of $\pi_{0.5}$ [8], post-trained on in-domain robot data. The robot uses 41 manipulation skills for opening drawers, moving cups and blocks, and pushing the black box, together with 2 active-perception skills for looking into a cup and looking into a drawer. The five evaluation tasks increase in difficulty: *push_box* requires pushing the black box to the

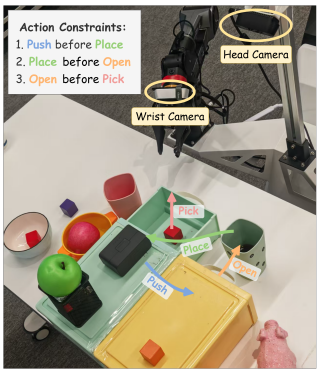
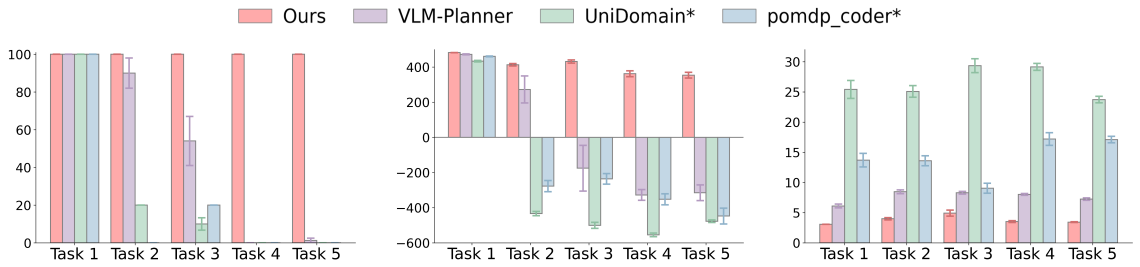


Fig. 2: Experimental setup.



(a) Task Progress (%) ↑ (b) Cumulative Reward ↑ (c) Planning Time per Step (s) ↓

Fig. 3: Performance comparison of our method and baselines.

top of a drawer; *block_in_drawer* requires placing a block on a drawer and closing drawers under hidden drawer interiors and stochastic grasping; *cup_with_water* requires identifying cups with dark liquid and resolving a box that blocks the target placement area; *cups_then_blocks* requires placing all blocks in the same drawer while handling long-horizon drawer-opening constraints; and *box_cups_blocks* adds cup placement and box-removal constraints on top of *cups_then_blocks*. For each evaluation task, we assess 10 trials. We report task progress, cumulative reward, and planning time. The reward function assigns positive reward for task completion, penalties for false completion and irreversible failures, and time cost for planning and execution. All evaluated methods use GPT-5.4 [14] as the LLM/VLM backend.

B. Baselines and Comparison Results

We compare against explicit model-learning methods and direct VLM-based planning:

- *VLM-Planner*: a VLM directly selects the next action from the current visual observation, execution history, and in-context demonstrations.
- *UniDomain** [20]: a deterministic PDDL domain-learning method. We adapt it to online planning by regenerating the PDDL problem at each step from the current visual observation and solving it with Monte Carlo tree search.
- *pomdp_coder** [3]: a method that learns a POMDP model represented as Python code from language-based demonstrations. We adapt it to our visual setting by applying our visual grounding pipeline to construct symbolic demonstrations and using the same belief initialization and belief update interface for online execution.

During domain learning, PO-PDDL learns a complete symbolic POMDP model with the lowest learning cost among explicit model-learning approaches. It takes 291 seconds to learn from 29 demonstrations containing 103 execution steps, while *UniDomain** and *pomdp_coder** require 5111 and 328 seconds, respectively.

As shown in Figure 3, PO-PDDL consistently outperforms all baselines across all task settings. *VLM-Planner* performs reasonably well on *push_box* and *block_in_drawer*, where constraint reasoning is limited, but it struggles once hidden states and action constraints become important and fails on the long-horizon tasks. *UniDomain** performs well on *push_box*, where uncertainty is minimal, but degrades on tasks with partial observability and stochastic execution because deterministic symbolic domains cannot consistently represent the same action under varying hidden pre-states. *pomdp_coder** fails on all tasks except the single-step *push_box*; although its learned programs cover many demonstrated cases, combining them into a globally consistent and solvable POMDP model remains difficult for complex visual robot domains.

In contrast, PO-PDDL successfully solves all tasks, achieving 100% task success. The strong performance comes from explicitly modeling action constraints with symbolic operator preconditions, representing partial observability and stochastic execution with a symbolic POMDP model, and applying closed-loop verification and revision during domain learning. PO-PDDL also achieves the shortest planning time, with an average improvement of 50.4% over *VLM-Planner*, 85.7% over *UniDomain**, and 73.2% over *pomdp_coder**. This efficiency comes from the fact that most steps only require symbolic belief updates using the learned transition model, while VLM calls are reserved for observation grounding and active-perception steps. Prompts, implementation details, and qualitative results are available on the project website: https://po-pddl.github.io.

VI. Conclusion and Limitations

We presented PO-PDDL, a symbolic POMDP formulation extending PDDL with stochastic transitions, partial observability, and belief representations while remaining LLM-friendly. Based on this formulation, we propose a demonstration-driven pipeline that learns reusable PO-PDDL domains from robot videos and enables belief-space planning.

Experiments on real-world manipulation tasks show that our method consistently outperforms existing PDDL and POMDP model-learning approaches, achieving stronger planning performance with significantly lower learning and planning cost.

Despite these results, limitations remain. First, we assume a closed-world setting without post-demonstration domain shift; extending to open-world environments requires online model revision from new observations. Second, learning cost may grow with large-scale demonstrations; a promising direction is building reusable modular PO-PDDL knowledge bases across tasks, similar to UniDomain. Finally, performance is bounded by the underlying VLM; future systems may explicitly detect such bottlenecks and request human assistance when needed.

REFERENCES

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[2] Walker Byrnes, Miroslav Bogdanovic, Avi Balakirsky, Stephen Balakirsky, and Animesh Garg. Climb: Language-guided continual learning for task planning with iterative model building. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8. IEEE, 2025.

[3] Aidan Curtis, Hao Tang, Thiago Veloso, Kevin Ellis, Joshua B Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Llm-guided probabilistic program induction for pomdp model estimation. In *Conference on Robot Learning*, pages 3137–3184. PMLR, 2025.

[4] Ziwei Deng, Mian Deng, Chenjing Liang, Zeming Gao, Chennan Ma, Chenxing Lin, Haipeng Zhang, Songzhu Mei, Siqi Shen, and Cheng Wang. Planu: Large language model reasoning through planning under uncertainty. *arXiv preprint arXiv:2510.18442*, 2025.

[5] Yanzhu Du, David Hsu, Hanna Kurniawati, Wee Sun Lee, Sylvie Ong, and Shao Wei Png. A pomdp approach to robot motion planning under uncertainty. In *International Conference on Automated Planning & Scheduling, Workshop on Solving Real-World POMDP Problems*, 2010.

[6] Liancheng Gong, Wang Zhu, Jesse Thomason, and Li Zhang. Zero-shot iterative formalization and planning in partially observable environments. *arXiv preprint arXiv:2505.13126*, 2025.

[7] Jinbang Huang, Yixin Xiao, Zhanguang Zhang, Mark Coates, Jianye Hao, and Yingxue Zhang. One demo is all it takes: Planning domain derivation with llms from a single demonstration. *arXiv preprint arXiv:2505.18382*, 2025.

[8] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[9] Connie Jiang, Yiqing Xu, and David Hsu. Llms for robotic object disambiguation. *arXiv preprint arXiv:2401.03388*, 2024.

[10] Yoonwoo Kim, Raghav Arora, Roberto Martín-Martín, Peter Stone, Ben Abbatematteo, and Yoonchang Sung. Large-language-model-guided state estimation for partially observable task and motion planning. *arXiv preprint arXiv:2603.03704*, 2026.

[11] Emanuele La Malfa, Ping Zhu, Samuele Marro, Sara Bernardini, and Michael Wooldridge. An end-to-end planning framework with agentic llms and pddl. *arXiv preprint arXiv:2512.09629*, 2025.

[12] Sadegh Mahdavi, Raquel Aoki, Keyi Tang, and Yanshuai Cao. Leveraging environment interaction for automated pddl translation and planning with large language models. *Advances in Neural Information Processing Systems*, 37: 38960–39008, 2024.

[13] James F Mullen Jr and Dinesh Manocha. Towards robots that know when they need help: Affordance-based uncertainty for large language model planners. *arXiv preprint arXiv:2403.13198*, 3, 2024.

[14] OpenAI. Gpt-5.4 thinking deployment safety report. Technical report, OpenAI, April 2026. URL https://deploymentsafety.openai.com/gpt-5-4-thinking/gpt-5-4-thinking.pdf. Accessed: May 1, 2026.

[15] Allen Z Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, et al. Robots that ask for help: Uncertainty alignment for large language model planners. *arXiv preprint arXiv:2307.01928*, 2023.

[16] Scott Sanner et al. Relational dynamic influence diagram language (rddl): Language description. *Unpublished ms. Australian National University*, 32(27):12, 2010.

[17] Lingfeng Sun, Devesh K Jha, Chiori Hori, Siddarth Jain, Radu Corcodel, Xinghao Zhu, Masayoshi Tomizuka, and Diego Romeres. Interactive planning using large language models for partially observable robotic tasks. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14054–14061. IEEE, 2024.

[18] Kangrui Wang, Pingyue Zhang, Zihan Wang, Yaning Gao, Linjie Li, Qineng Wang, Hanyang Chen, Chi Wan, Yiping Lu, Zhengyuan Yang, et al. Vagen: Reinforcing world model reasoning for multi-turn vlm agents. *arXiv preprint arXiv:2510.16907*, 2025.

[19] Qianwei Wang, Bowen Li, Zhanpeng Luo, Yifan Xu, Alexander Gray, Tom Silver, Sebastian Scherer, Katia Sycara, and Yaqi Xie. Unifying deep predicate invention with pre-trained foundation models. *arXiv preprint arXiv:2512.17992*, 2025.

[20] Haoming Ye, Yunxiao Xiao, Cewu Lu, and Panpan Cai. Unidomain: Pretraining a unified pddl domain from real-world demonstrations for generalizable robot task planning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[21] Nan Ye, Adhiraj Somani, David Hsu, and Wee Sun Lee. DESPOT: Online POMDP planning with regularization. *Journal of Artificial Intelligence Research*, 58:231–266, 2017.

[22] Zhouliang Yu, Yuhuan Yuan, Tim Z Xiao, Fuxiang Frank Xia, Jie Fu, Ge Zhang, Ge Lin, and Weiyang Liu. Generating symbolic world models via test-time scaling of large language models. *arXiv preprint arXiv:2502.04728*, 2025.

[23] Linfeng Zhao, Willie McClinton, Aidan Curtis, Nishanth Kumar, Tom Silver, Leslie Pack Kaelbling, and Lawson LS Wong. Seeing is believing: Belief-space planning with foundation models as uncertainty estimators. *arXiv preprint arXiv:2504.03245*, 2025.

[24] Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for large-scale task

planning. *Advances in neural information processing systems*, 36:31967–31987, 2023.