Generalizable Task Planning through Representation Pretraining

Chen Wang[1], **Danfei Xu**[2], **Li Fei-Fei**[1]
[1] Stanford Vision and Learning Lab [2] NVIDIA *

Abstract: The ability to plan for multi-step manipulation tasks in unseen situations is crucial for future home robots. But collecting sufficient experience data for end-to-end learning is often infeasible in the real world, as deploying robots in many environments can be prohibitively expensive. On the other hand, large-scale scene understanding datasets contain diverse and rich semantic and geometric information. But how to leverage such information for manipulation remains an open problem. In this paper, we propose a learning-to-plan method that can generalize to new object instances by leveraging object-level representations extracted from a synthetic scene understanding dataset. We evaluate our method with a suite of challenging multi-step manipulation tasks inspired by household activities [1] and show that our model achieves measurably better success rate than state-of-the-art end-to-end approaches. Additional information can be found at: https://sites.google.com/view/gentp

Keywords: Integrated Planning and Learning, Representation Learning

1 Introduction

Planning for everyday manipulation tasks in home environments is challenging for multiple reasons. The problem not only requires searching in high-dimensional, non-convex spaces over long time horizons but also poses major *representation challenges*. For example, consider the task of cleaning a dining table with a towel by soaking the towel first. The planner needs to represent both geometric (e.g., location of the sink) and semantic (e.g., a towel is soaked) states as well as how actions might affect these states. While research fields such as Task and Motion Planning (TAMP) have made significant progress in solving these tasks efficiently [2, 3, 4, 5, 6], most of the approaches rely on carefully-chosen abstract representations and analytically-defined transition models, both of which are often domain- or even task-specific. Thus for a home robot to be effective, it is vital to equip its planner with planning representations that can generalize to a wide range of tasks and environments.

Fortunately, recent progress in deep learning has shown that it is possible to extract generalizable representations from raw perception data such as images. A particular relevant thread is in visual reasoning, where works [7, 8, 9] have shown that implicit object-centric representations learned through large-scale scene understanding tasks such as predicting spatial relationships can be transferred to new tasks and generalize to unseen objects. However, there are two open challenges when applying similar recipes to learning planning representations. First of all, planning with manipulation skills requires a certain level of *representation granularity*. For example, modeling the effects of a *placing* skill requires reasoning about the spatial locations and the shapes of an object and a target surface. Representations derived from just learning abstract concepts such as `on-top` may not be sufficient for planning. What pretraining strategies can allow us to extract suitable planning representations? Second, planning systems such as TAMP assume explicit states (e.g., object poses) and world models (e.g., kinematic transition models). How can we develop a method that can plan with learned implicit representations?

*This work has been accepted at the Robotics and Automation Letters, June 2022.

6th Conference on Robot Learning (CoRL 2022), Auckland, New Zealand.

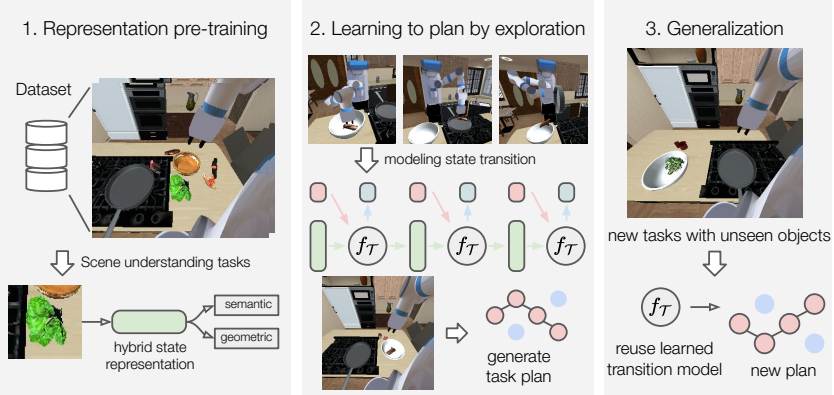


Figure 1: **Overview**: We propose a two-stage approach to learn a generalizable task planner. (1) The first stage learns object-centric representations from a large-scale scene understanding dataset through hybrid semantic-symbolic objectives. (2) Based on the pre-trained representation, the second stage trains an environment dynamics model $f_{\mathcal{T}}$ for search-based task planning. (3) We show that the task planner inherits the generalization capabilities of the learned representation and can successfully plan in new settings with unseen objects.

In this paper, we aim to develop a learning-to-plan method that can generalize to new objects and task goals through addressing these two problems. On a high level, we propose a two-stage approach. The first stage extracts object-level embeddings from a dataset of raw RGB observations using a suite of auxiliary pretraining objectives. The goal is to find invariant features that generalize to new objects without additional finetuning. The objectives include learning high-level semantic attributes and tasks that require more fine-grained geometric understanding such as pixel-level segmentation. We empirically show that such hybrid learning objectives are crucial for the subsequent stage of learning to plan with parameterized manipulation skills. Moreover, the learned representations do not pertain to specific planning problems and thus can be reused across tasks.

The second stage is a novel planning framework that builds a dynamics model of the environment based on the learned object representations. We show that our search-based planner can plan for different tasks with a single learned transition model. More importantly, the learned object representations allow the planner to generalize to new objects with no retraining or finetuning. At the same time, learning to plan with unconstrained, continuously-parameterized skills still poses a major technical challenge. To this front, we propose a novel dynamics model that *grounds* the skill parameters in the learned planning space, allowing the model to directly associate the effects of skill with the implicit representation of a target object. We show that the model greatly enhances the robot's ability to solve tasks that require more precise skill parameter selection.

We evaluate our method with three manipulation tasks of varying difficulties. The *Rearrangment* task is to rearrange two (possibly unseen) objects that are placed on two tables. In the *Cleaning* task, the robot needs to first soak a towel in a sink and use the soaked towel to clean a microwave oven, testing the planner's ability to represent both geometric and semantic states. Finally, in the most challenging *Cooking* task, the robot is asked to plan for cooking unseen food objects with a pan and serve the cooked food in a bowl. We empirically demonstrate our method's ability to plan for complex multi-step manipulation tasks and generalize to new objects and task goals.

2 Related works

2.1 Task and Motion Planning

Task and Motion Planning (TAMP) methods are effective at solving multi-step robot manipulation tasks [2, 3, 4, 5, 6]. However, most established TAMP methods rely on analytically-defined components such as preimage functions and environment kinematics models. More recent works sought

to replace these components with learned modules. Notable examples include learning to predict plan feasibilities [10, 11] and learning skill effect models [12, 13]. While these works have relaxed many assumptions of TAMP methods, the learned modules are often built on top of hand-defined representation spaces. For example, Wang *et al.* [14] explicitly models weight changes of a container as a result of executing a pouring skill, and Liang *et al.* [13] constructs skill effect model based on relative object poses. As a result, these learned modules are often limited to specific tasks or domains. Instead, we propose to learn an implicit planning space through a suite of task-agnostic representation learning objectives. We show that our learning-to-plan framework based on such representations can generalize to new tasks and object categories.

2.2 Learning to plan

Our method is closely related to model-based reinforcement learning [15, 16]. Most recent works have focused on using observation space as the state representation to build full environment models [17, 18]. However, learning to make accurate predictions in, e.g., raw image space, is still challenging [19, 15], especially for long-horizon manipulation tasks. Instead, our approach learns a partial model of the world by first extracting features of image observation through supervised pretraining and building dynamics models based on such compact representations.

For works that plan with partial models, many focus on predicting either reward or task-specific quantities [20, 16] through end-to-end learning. As a result, it is difficult for a learned model to generalize to new tasks and settings. To address the challenge, some recent works turn to task-agnostic partial models [21, 11, 22]. For example, Xu *et al.* [21] proposes to learn environment models based on object affordances that do not pertain to specific task goals. However, since the representations are learned together with the dynamics model in an end-to-end manner from interaction data, the learned representations are still tied to the specific settings where data is collected. Furthermore, collecting interaction data in broad and diverse settings can be prohibitively expensive in the real world. In contrast, our representation learning learns from static scene understanding dataset, which can be either generated synthetically or crowd-sources to cover diverse objects and scenarios.

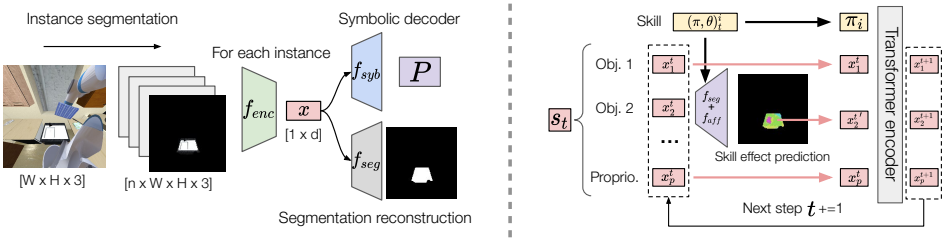
2.3 Representation learning for generalizable manipulation

Recent research suggests that learning from diverse and large-scale dataset is a promising path towards generalizable manipulation [23, 24, 25]. Examples of such dataset include demonstrations across domains [23, 24, 26, 27], mixture of exploration and expert data [28], and instruction-conditional data [25].

One way to leverage such datasets for robotic manipulation is learning generalizable state representations [27, 29, 9]. For example, SORNet [9] learns object embeddings that could be used to estimates the spatial relationships of unseen objects. Similarly, Cliport [25] transfers representations pretrained on a vision-language dataset [8] to manipulation skills and show generalizations to new tasks. However, these works almost exclusively focus on single-step state estimation or learning short-horizon skills. Our work instead takes a step towards building generalization planners for multi-step manipulation problems by leveraging representations learned from large-scale datasets.

3 Method

We propose a two-stage framework to achieve generalizable task planning with parameterized skills. In the first stage, an observation encoder is trained to extract object-level features from a large-scale scene understanding dataset. The goal is to discover a suitable state representation space that could generalize to a wide variety of objects. In the second stage, we develop a novel task planner that (1) learns environment dynamics in the same representation space obtained in the first stage and (2) plans for long-horizon tasks using a search-based strategy. Here we first lay down the task planning problem setups, then describe the representation learning scheme, and finally present the learning-to-plan algorithm.



(a). Learning object-centric representations

(b). Spatially-grounded transition model

Figure 2: (**a**) The representation learning process consists of a high-level symbol grounding and a pixel-level segmentation reconstruction task. (**b**) The spatially-grounded transition model $f_{\mathcal{T}}$ takes a set of object-centric embeddings as input and generates a pixel-level prediction of the skill effect. Each pixel feature on the prediction map refers to the next symbolic state of the target object after the skill execution. To handle the situation where one action affects multiple objects, a transformer encoder network takes all objects' state embeddings as inputs and outputs a final prediction of the skill effect over all objects.

3.1 Problem Setup

State representation: Consider a multi-step manipulation planning problem in a partially-observable setting. The goal is to take observation $o \in O$ as inputs and plan for a sequence of parameterized skills (described later) to reach a task goal $g \in G$. The goal space G in this work is a set of known symbolic predicates $g = \{p_1, p_2, \dots\}$ that summarize the object-level state information of the environment. For example, the goal of a cooking task can be expressed as $p_1 =$ `Cooked(food)`, $p_2 =$ `OnTop(food, plate)`. To bridge the gap between the symbolic representation and raw observations, we adopt an object-centric observation space and state representation. We follow prior works [30, 31, 32] and assume the observation is given in the form of segmentation-masked images for each object (Fig. 5 (a)). In the representation learning stage (Sec. 3.2), we train an observation encoder to map the object-centric observations into an implicit representation space $f_{obs} : O \to \mathcal{S}$, where each state $s = \{x_0, x_1, \dots, x_n\}$ is a set of object-level embeddings. The representation space serves as a foundation for the subsequent learning-to-plan stage (Sec. 3.3).

Parameterized skills: Following prior works [33, 21, 13], we define a parameterized skill as a policy $\pi \in \Pi$. Each skill π is modulated by a parameter θ that specifies skill-specific execution details such as the 3D grasping poses for the grasping skill. In our task-planning formulation, there are two additional elements for a given parameterized skill: a pre-condition identifier that specifies whether the skill is *executable* with the given parameters θ, and a state transition model $s_{t+1} = f_{\mathcal{T}}(s_t, \pi, \theta)$. Different from prior task planning works which either assume the pre-conditions of the skills are given [13, 4] or the transition model is pre-defined [4, 34], our method treats both as unknown and should be learned from experience data (Sec. 3.3).

Search-based task planning: The task planning problem is to find a parameterized skill plan $\{(\pi, \theta)_t\}_{t=1}^{T}$ such that the goal condition g is satisfied at the end of the last skill. Recall that goals are conjunctions of symbolic predicate conditions. A task is successful if and only if all conditions are satisfied. We adopt a tree-search algorithm to plan for both skill skeletons and parameters (Sec. 3.4).

3.2 Representation Learning for Planning

A challenge for planning with raw sensory observation is to extract suitable representation. TAMP methods rely on carefully-designed abstractions such as object poses or semantic states [4, 13]. These representations are often domain-specific and require extensive hand-engineering. Other works learn state representation together with the world models in an end-to-end fashion[16, 21]. However, these methods require interactive experience data and the learned representation often overfits to the specific task settings. Recent works in visual reasoning [7, 9] showcase that it is possible to learn implicit representation from scene understanding tasks that could transfer to unseen objects. However, as we will show empirically, representations containing only abstract concepts are

not sufficient for manipulation planning. Modeling the effects of parameterized skills also requires more detailed information such as spatial location and shape. In this work, we propose a suite of supervised tasks to encode information at different abstraction levels.

Specifically, we consider a high-level symbol grounding task and a pixel-level segmentation reconstruction task as shown in (Fig. 5 (a)). We use an encoder-decoder structure to jointly optimize the two objectives. First, an observation model f_{enc} maps the object-centric observations into object embedding space S. The symbol grounding task is to predict a set of high-level symbolic states P for each embedding. We consider two types of predicates: object-level states (unary) and relationships between objects (binary). Each predicate has an individual decoder network f_{syb} that is shared across objects. The network takes a concatenation of two object embeddings as input for binary predicates. For the segmentation reconstruction task, we use a CNN-based decoder f_{seg} to reconstruct the object's segmentation mask from its feature vectors. Intuitively, this task is to instill a compact encoding of the shape and the location information in the learned object representation. To satisfy both supervised learning objectives, the f_{enc} network has to extract both high-level symbolic features and low-level geometrical locations of the object from the observation. Note that, our framework is not limited to these two types of information encoding and could potentially incorporate additional supervisions to enrich the representation for different manipulation goals.

3.3 Learning Transition Model

Another challenge of task planning is to predict the effects of skills so that the robot could determine the best skill plan for a task goal. To enable such capability, traditional methods rely on hardcoded analytical transition models [5, 4]. But this may not always be feasible, especially for complex dynamics such as nonprehensile contacts. For learning-to-plan methods, although the transition model can be learned from experience data [16, 17, 21, 35], it is often tied to the training environments and requires additional finetuning to transfer to new objects and task settings.

Here we present a learning-to-plan method that builds a generalizable transition model by leveraging the learned object representations. As described above, the pretraining scheme allows the object embeddings to focus their representation powers on task-relevant features. For example, learning to predict the relationship `on-top` encourages the encoder to discard appearance features and focus on the spatial locations of the objects. Accordingly, a transition model built within such a representation space can potentially generalize its knowledge about the skill effects to new objects. For example, the effect of a placing skill depends only on the shape of an object and the location of the supporting surface instead of their textures.

To realize this intuition, we need to address two technical challenges: (1) How to constrain the learned transition to be within the pre-trained representation space. (2) How to ground parameterized skills on the implicit representations.

Learning generalizable transition model. To allow the transition model to inherit the generalizability of the learned representation, we must constrain its prediction to the same representation space S. We employ two regulatory objectives: state regression and representation alignment. Given an encoded trajectory $\{(s_t, \pi_t, \theta_t)\}_{t=1}^{T}$, state regression is to train the transition model $f_{\mathcal{T}}(\cdot)$ to predict future states in the learned representation space through an L2 loss.

$$L_{reg} = \sum_{t=1}^{T-1} ||f_{\mathcal{T}}(s_t, \pi_t, \theta_i) - f_{enc}(s_{t+1})|| \tag{1}$$

However, optimizing for this objective alone cannot guarantee that the predicted state is still in S, because a small error in the representation space may be amplified during decoding. Hence we introduce a second objective to *align* the predicted representation using the same learning objectives described in Sec. 3.2. More specifically, we freeze the decoder networks f_{syb} and f_{seg} and use the gradient from their prediction errors to update the object representations. Intuitively, these objectives encourage the transition model to predict future states that are (1) close to the reference states and (2) can be decoded into meaningful symbolic predicates and object segmentations.

Spatially-grounded transition model. Here we dive into the details of the transition model $f_{\mathcal{T}}(s, \pi, \theta)$. A major challenge of modeling the effects of a parameterized skill is to associate its parameters with its target object. For example, it is hard to predict whether a 6-D grasping pose in the robot coordinate frame will successfully grasp the handle of a pan represented as image observation. Inspired by prior works in visual skill affordance [36, 21, 37], we propose to *ground* skill parameters *spatially* onto the image observation space.

Having established connections between pixel space and skill parameter space, we are ready to introduce our spatially-grounded transition model. On a high level, the model learns to predict the effect of applying a skill on a target object. And it generates predictions for all parameter choices included in an image plane. As shown in Fig 5 (b), the proposed transition model has two components. First, the model decodes each object embedding x_i to an object embedding map of shape $[H \times W \times d]$, with each pixel representing the effect of a skill parameterized by that pixel location. However, the prediction assumes the skill has effects confined to the target object. But most manipulation skills affect more than one object. As a simple example, a pouring skill changes both the state of the receptacle and the container in hand. Hence the second component in our transition model is a transformer encoder network [38] that models how applying a skill may influence each object in a scene. The transformer network takes the skill-effect prediction on the target object and all the other objects' current state embeddings as inputs, and learns to reason the dependency between objects and outputs the skil-effect prediction for each object. Together, the two components allows us to model the effects of parameterized skills as a single-step transition $s_{t+1} = f_{\mathcal{T}}(s_t, \pi, \theta)$. We apply the model autoregressively to predict the outcome of a multi-step plan.

3.4 Planning with Learned Transition Models

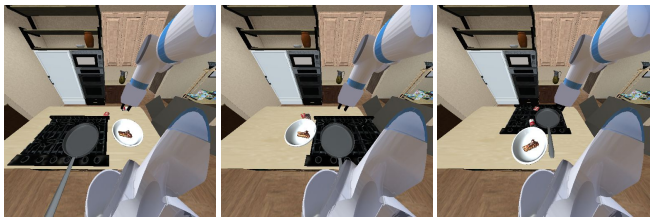
The goal of a task planner is to find a sequence of parametrized skills that are likely to reach a task goal (Sec. 3.3). More importantly, the planner should be able to plan for different task goals without the need of re-training or finetuning. Because our transition model is trained with task-agnostic exploration data, a planner can use it to search for any reachable goals that are represented in the transition model. However, the major challenge of searching for a multi-step manipulation task is to *efficiently* find the plan and skill parameters in large planning space. Since enumerating all skill parameters is infeasible, it is crucial to prune less promising branches to accelerate the search.

We introduce two pruning criteria. First, we score each parameter by its corresponding segmentation confidence given by f_{seg}. Then for the effect embedding of each skill, we decode their symbolic states using the decoder f_{syb} and score the parameter by the average confidence (binary classification score) of the symbolic state predictions. The intuition is that if the confidence is low, then the model is likely not to have been trained on similar transitions and is likely to make poor predictions along such branches. Note that these two pruning criteria are by-products of the pixel-level transition model and could substantially improve the inference speed. Based on the pruning criteria, we leverage a breadth-first tree-search algorithm to conduct the planning. In each search node, the predicted symbolic state P is checked against the task goal g. If multiple skill plans are found to reach the goal, the plan with the highest accumulated confidence score is executed. Appendix. 6.2 includes more implementation details of the pruning and searching process.

4 Experiments

In this section, we aim to validate our hypothesis that: (1) the object representations obtained from the pretraining stage is suitable for multi-step task planning, (2) our spatially-grounded transition model can better model the effects of parameterized skills compared to an ungrounded model [21], and (3) our learning-based task planner built on the pre-trained representations can generalize to unseen objects and improve the sample efficiency of learning to plan in different environments.

Task setup. We test our models and baselines in three simulated manipulation tasks adapted from the BEHAVIOR [1] and simulated using PyBullet [39]. BEHAVIOR is a collection of long-horizon



(a) Layout 1 (b) Layout 2 (c) Layout 3

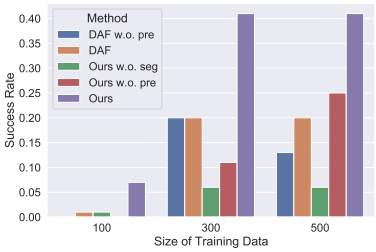


Figure 3: (1). Three different layouts in the Cooking task. (2). Evaluation results with different amount of training samples in the Cooking task with three environment layouts.

Table 1: Results of the **Rearrangement** task

Rearranging	DAF w.o. pre	DAF	Ours w.o. seg	Ours w.o. pre	Ours
Seen	0.65	**0.71**	0.29	0.68	0.65
Unseen 1	0.01	0.62	0.31	0.1	0.47
Unseen 2	0	0.34	0.28	0.02	0.39
Unseen 3	0.01	0.26	0.33	0.01	0.56
Mean unseen	0.01	0.41	0.31	0.04	**0.47**

Table 2: Results of the **Cooking** task

Cooking	DAF w.o. pre	DAF	Ours w.o. seg	Ours w.o. pre	Ours
Seen	0.09	0.30	0.24	0.10	**0.48**
Unseen 1	0.09	0.34	0.22	0.04	0.32
Unseen 2	0.04	0.12	0.13	0.03	0.30
Unseen 3	0.03	0.12	0.16	0.03	0.31
Mean unseen	0.06	0.19	0.17	0.04	**0.31**

Table 3: Results of the **Cleaning** task

Cleaning	DAF w.o. pre	DAF	Ours w.o. seg	Ours w.o. pre	Ours
Seen	0.16	0.08	0.14	**0.44**	0.37

tasks that strives to capture the complexity of real-world household activities. The tasks feature a wide range of kinematic and semantic states. For example, a food item can be `OnTop` of a pan and `Cooked`. These states are both represented symbolically and externalized as visual appearance changes. Appendix. 6.1.1 includes more details of the initial and goal state definitions of each task.

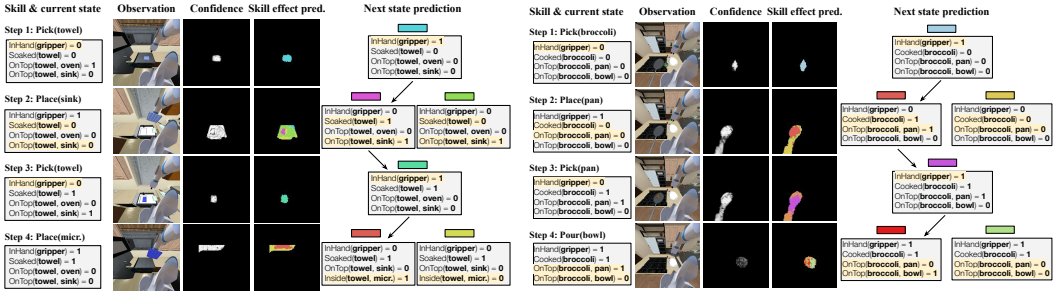
Parameterized skills. The agent is provided with three location-based parameterized motor skills: *pick*, *place* and *pour*. *Pick* executes top-down grasps parameterized by a 3D grasping location. Similarly, *place* releases a grasped object at a certain 3D location relative to the agent. *pour* moves a container object (e.g. pan) to a 3D location and execute a fixed pouring motion. The motion trajectory of all skills are generated using RRT-based [40] motion planners.

Dataset. We make use of two data sources. The first is a scene understanding dataset for representation learning. The content is synthesized using the BEHAVIOR engine by randomizing both the kinematic and semantic states of a wide range of objects. The second is an experience dataset for learning the state transition model $f_{\mathcal{T}}$, which is collected through random robot exploration. Appendix. 6.1.2 includes more details of our dataset components.

4.1 Results

Hybrid semantic-geometric representation is important for manipulation planning. In all three experiments, the model learned without low-level segmentation details (Ours w.o. seg) underperforms Ours in both *Seen* and *Unseen* object settings. Moreover, we observe that when training the model with data collected from different environment layouts (Fig. 3), **Ours w.o. seg** failed to generate reasonable task plans.

Spatially-grounded transition model outperforms unstructured transition model. While DAF achieves performances that are comparable to our method in the easiest *Rearrangement* task, its performance drops significantly when planning for tasks that require more precise skill parameter selections such as putting the towel into the sink in the *Cleaning* task and pouring the food into the bowl in the *Cooking* task. Our approach with the spatially-grounded transition model outperforms DAF for more than 15% with *Seen* objects in these two environments.



(a) Cleaning the microwave oven

(b) Cooking the broccoli

Figure 4: Qualitative results of the pixel-level skill effect prediction in the Cleaning and Cooking tasks. The leftmost column is the states prior to applying the skills. The rightmost column shows a simplified view of the tree search process. Each color on the embedding map represent the predicted effect of a skill in symbolic space. The correspondence between the pixel color and its representing symbolic state is shown on the right side of each figure.

Pre-trained representation is the key for generalization to new objects. We evaluate our approach and DAF both with and without the pre-trained representation space in unseen object settings, where we replace the towels, foods, bowls and household objects with unseen instances that not included in the scene understanding dataset and the experience dataset. In Tab. 1, both models trained without the pre-trained representation (**DAF w.o. pre** and **Ours w.o. pre**) suffer a huge performance drop from $> 60\%$ success rate down to close to 0% results after Rearrangment the testing objects from seen to unseen. In contrast, both models with the pre-trained representation has a smaller performance drop (20% for **Ours** and 30% for **DAF**). **Ours** achieves the best result in both *Rearrangement* and *Cooking* tasks, which outperforms the other baselines for more than 10% in the most challenging cooking unseen food task.

Pre-trained representation improves the sample-efficiency of learning in multiple environments. In Fig. 3, we observe that our approach with the pre-trained representation outperforms the baseline methods in all three training settings with different amounts of training samples. Noticeably, with only 60% of the training data, **Ours** outperforms baselines for more than 20% in success rate. Although **DAF** also leverage the pre-trained representation, their performance doesn't show much improvement. This also highlights the effect of our spatially-grounded transition model on generalizing to different task environments.

Visualizing learned transitions. We visualize the skill effect predictions from our transition model in Fig. 4. Each colored pixel on the skill-effect prediction map shows the predicted next symbolic state when a skill is applied at that location. For example, the second step of the cleaning task shows placing the towel at different locations on the counter leads to different future states. And the only way to reach the `Soaked` state is to place the towel inside the sink. Similarly in the cooking task, only the cooking surface of the pan affords the `Cooked` state of the food. We also illustrate how our model might take into account the confidence of a transition prediction. Taking pouring an example (last row of *Cooking* in Fig. 4): Our model prefers pouring locations that are close to the left half of the bowl (higher confidence) to take into account the inertia of the food items to avoid overshooting.

5 Conclusion

We have proposed a new learning-to-plan method that leverages object representations learned through large-scale pretraining. Through a suite of tasks inspired by everyday home activities, we show that the planner can (1) solve complex multi-step manipulation tasks and (2) generalize to new task goals and objects by inheriting invariant features from the learned representation. Our method is open to many interesting future directions. (1). How to leverage neuro-symbolic skills for more efficient bilevel planning [41]. (2). How to incorporate language-based primitive skills for more generalizable manipulation [25].

References

[1] S. Srivastava, C. Li, M. Lingelbach, R. Martín-Martín, F. Xia, K. E. Vainio, Z. Lian, C. Gokmen, S. Buch, K. Liu, et al. Behavior: Benchmark for everyday household activities in virtual, interactive, and ecological environments. In *Conference on Robot Learning*, pages 477–490. PMLR, 2022.

[2] T. Lozano-Pérez and L. P. Kaelbling. A constraint-based method for solving sequential manipulation planning problems. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3684–3691. IEEE, 2014.

[3] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4:265–293, 2021.

[4] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling. Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448, 2020.

[5] M. Toussaint. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[6] M. A. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. 2018.

[7] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *arXiv preprint arXiv:1904.12584*, 2019.

[8] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[9] W. Yuan, C. Paxton, K. Desingh, and D. Fox. SORNet: Spatial object-centric representations for sequential manipulation. In *5th Annual Conference on Robot Learning*, 2021.

[10] L. P. Kaelbling and T. Lozano-Pférez. Learning composable models of parameterized skills. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 886–893. IEEE, 2017.

[11] D. Driess, J.-S. Ha, and M. Toussaint. Deep visual reasoning: Learning to predict action sequences for task and motion planning from an initial scene image. RSS, 2020.

[12] Z. Wang, C. R. Garrett, L. P. Kaelbling, and T. Lozano-Pérez. Learning compositional models of robot skills for task and motion planning. *The International Journal of Robotics Research*, 40(6-7):866–894, 2021.

[13] J. Liang, M. Sharma, A. LaGrassa, S. Vats, S. Saxena, and O. Kroemer. Search-based task planning with learned skill effect models for lifelong robotic manipulation. *arXiv preprint arXiv:2109.08771*, 2021.

[14] C. Wang, R. Wang, A. Mandlekar, L. Fei-Fei, S. Savarese, and D. Xu. Generalization through hand-eye coordination: An action space for learning spatially-invariant visuomotor control. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8913–8920. IEEE, 2021.

[15] C. Finn and S. Levine. Deep visual foresight for planning robot motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

[16] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[17] S. Tian, S. Nair, F. Ebert, S. Dasari, B. Eysenbach, C. Finn, and S. Levine. Model-based visual planning with self-supervised functional distances. *arXiv preprint arXiv:2012.15373*, 2020.

[18] B. Wu, S. Nair, L. Fei-Fei, and C. Finn. Example-driven model-based reinforcement learning for solving long-horizon visuomotor tasks. *arXiv preprint arXiv:2109.10312*, 2021.

[19] J. Oh, S. Singh, and H. Lee. Value prediction network. *Advances in neural information processing systems*, 2017.

[20] A. Dosovitskiy and V. Koltun. Learning to act by predicting the future. *arXiv preprint arXiv:1611.01779*, 2016.

[21] D. Xu, A. Mandlekar, R. Martín-Martín, Y. Zhu, S. Savarese, and L. Fei-Fei. Deep affordance foresight: Planning through what can be done in the future. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6206–6213. IEEE, 2021.

[22] D. Driess, J.-S. Ha, R. Tedrake, and M. Toussaint. Learning geometric reasoning and control for long-horizon tasks from visual input. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14298–14305, 2021.

[23] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021.

[24] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[25] M. Shridhar, L. Manuelli, and D. Fox. Cliport: What and where pathways for robotic manipulation. In *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021.

[26] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1048–1055. IEEE, 2019.

[27] J. Pari, N. Muhammad, S. P. Arunachalam, L. Pinto, et al. The surprising effectiveness of representation learning for visual imitation. *arXiv preprint arXiv:2112.01511*, 2021.

[28] S. Cabi, S. G. Colmenarejo, A. Novikov, K. Konyushkova, S. Reed, R. Jeong, K. Zolna, Y. Aytar, D. Budden, M. Vecerik, et al. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *arXiv preprint arXiv:1909.12200*, 2019.

[29] K. Kase, C. Paxton, H. Mazhar, T. Ogata, and D. Fox. Transferable task execution from pixels through deep planning domain learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10459–10465. IEEE, 2020.

[30] D. Xu, R. Martín-Martín, D.-A. Huang, Y. Zhu, S. Savarese, and L. F. Fei-Fei. Regression planning networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[31] S. Mukherjee, C. Paxton, A. Mousavian, A. Fishman, M. Likhachev, and D. Fox. Reactive long horizon task execution via visual skill and precondition models. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5717–5724. IEEE, 2020.

[32] A. H. Qureshi, A. Mousavian, C. Paxton, M. C. Yip, and D. Fox. Nerp: Neural rearrangement planning for unknown objects. *arXiv preprint arXiv:2106.01352*, 2021.

[33] B. Ames, A. Thackston, and G. Konidaris. Learning symbolic representations for planning with parameterized skills. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 526–533. IEEE, 2018.

[34] A. Curtis, X. Fang, L. P. Kaelbling, T. Lozano-Pérez, and C. R. Garrett. Long-horizon manipulation of unknown objects via task and motion planning with estimated affordances. *arXiv preprint arXiv:2108.04145*, 2021.

[35] R. Chitnis, T. Silver, J. B. Tenenbaum, T. Lozano-Perez, and L. P. Kaelbling. Learning neuro-symbolic relational transition models for bilevel planning. *arXiv preprint arXiv:2105.14074*, 2021.

[36] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, et al. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018.

[37] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee. Transporter networks: Rearranging the visual world for robotic manipulation. *Conference on Robot Learning (CoRL)*, 2020.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, 2017.

[39] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. 2016.

[40] J. J. Kuffner and S. M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2. IEEE, 2000.

[41] T. Silver, A. Athalye, J. B. Tenenbaum, T. Lozano-Perez, and L. P. Kaelbling. Learning neuro-symbolic skills for bilevel planning. *arXiv preprint arXiv:2206.10680*, 2022.

6 Appendix

6.1 Experiment setups

6.1.1 Task setups.

For the *Rearrangement* task, the initial state is `OnTop(A, sink)` $\wedge$ `OnTop(B, oven)`, the goal of the task is `OnTop(B, sink)` $\wedge$ `OnTop(A, oven)`. Here `A`, `B` are placeholders for objects and we use objects within the training data (*Seen*) and those not included in the training set (*Unseen*) to evaluate generalization. For the *Cleaning* task, the initial state is `OnTop(towel, oven)` $\wedge$ $\neg$`Soaked(towel)` and the goal state is `InSide(towel, microwave)` $\wedge$ `Soaked(towel)`. This requires the agent to first soak the towel with water in the sink then place the towel into the microwave oven for cleaning the dirt marks inside. *Rearrangement* and *Cleaning* share the same environment and training datasets. And we use the same learned model to plan for both tasks. For the *Cooking* task, the initial state is `OnTop(A, table)` $\wedge$ $\neg$`Cooked(A)` and the goal state is `OnTop(A, bowl)` $\wedge$ `Cooked(A)`. This requires the agent to first cook the food item `A` with a pan then serve it on a bowl.

6.1.2 Dataset details.

Objects. We mainly consider five object categories in our experiments: 27 food objects, 18 household objects, 20 bowl containers, two cleaning towels, and a cooking pan. The food items can be set to `Cooked`. The cleaning towels can be `Soaked` indicating whether they have been soaked in the sink. The rest of the objects categories can only experience kinematic state changes. During the data collection of the representation learning dataset, we divide the first three categories into a *Seen* object set used for synthesizing the representation learning dataset (20 food instances, 13 household objects, 15 bowls) and set aside the rest as *Unseen* set for evaluation. In this work, we focus on exploring the model's category-level generalization, which means both *Seen* and *Unseen* instances are within known object categories. For the collection of the experience dataset, we only use a small subset of *Seen* objects (1 food, 2 household objects, 1 bowl). The goal is to evaluate the sample efficiency and generalization capability of each method subject to a limited amount of experience data for learning environment dynamics.

Environment layouts. For all tasks, we randomize the initial positions of task-relevant objects in confined areas. In the *Cooking* task, we further create three different layouts by swapping the location of the objects (Fig. 3). This creates a harder learning problem as the model has to adapt to different initial configurations.

Exploration data. With the pre-defined parameterized skills, we let the agent explore the environment with randomly selected skill and skill parameters. The collected exploration dataset is used for training the transition model $f_{\mathcal{T}}$. To improve the diversity of the data, we apply sampling heuristics such as the same skill won't be sampled consecutively and the skill parameters are only sampled within the segmentation mask of the target object. For each environment, 500 exploration trajectories with the sequence length of 20 are sampled.

6.2 Implementation Details

This section describes the implementation details of the pruning process of the spatially-grounded transition model $f_{\mathcal{T}}$ and the search-based task planning framework proposed in the main paper.

6.2.1 Inference with learned spatially-grounded transition model

Given the state input s_t at time step t, we first search over skill selection $\pi \in \Pi$ and target object $x \in s_t$. For each pair (π, x), the segmentation decoder network f_{seg} first localizes the target object by predicting a binary segmentation mask $\mathbf{M}$ from the object embedding x. The skill-effect network f_{aff} takes the skill and object feature as inputs and predicts a dense pixel-wise skill effect map

Algorithm 1: Next State Sampling with Learned $f_{\mathcal{T}}$

Output: $\mathcal{F}$ ▷ A list of potential next states

Input: s_t, Π, $f_{\mathcal{T}} = \{f_{seg}, f_{aff}, f_{trans}\}$, f_{syb}

Initialize $\mathcal{F} \leftarrow \emptyset$

for π *in* Π **do**

  for x *in* s_t **do**

    $\mathbf{M} = f_{seg}(x)$ ▷ object segmentation mask

    $\mathbf{E} = f_{aff}(\pi, x)$ ▷ skill-effect prediction map

    $\mathbf{E} = \mathbf{M} \cdot \mathbf{E}$ ▷ mask out irrelevant pixels

    $\mathbf{S}_{t+1} = f_{trans}(s_t, \mathbf{E})$ ▷ skill-effect prediction

    $\mathbf{P}_{t+1}, \mathbf{C}_{t+1} = f_{syb}(\mathbf{S}_{t+1})$ ▷ symbolic state decoding along with confidence score

    for P_{t+1} *in* `set`$(\mathbf{P}_{t+1})$ **do**

      $\theta = \arg\max(\mathbf{C}_{t+1}[\mathbf{P}_{t+1} == P_{t+1}])$

      $\mathcal{F}$`.append`$((\pi, \theta, \mathbf{S}_{t+1}[\theta]))$

return $\mathcal{F}$

$\mathbf{E}$. We mask out the irrelevant pixel features using the predicted segmentation $\mathbf{M}$. To handle the situations where a skill may affect more than one object, the transformer network f_{trans} takes each predicted skill-effect feature and all other objects' current state feature as inputs and generates a final pixel-wise skill effect map of the next state $\mathbf{S}_{t+1}$.

The next step is to sample states from the pixel prediction $\mathbf{S}_{t+1}$. However, searching over each pixel is time consuming and most of the pixels are redundant since they would reach the same symbolic state. We empirically find that our pre-trained symbolic decoder network f_{syb} provides useful confidence signal to prune less promising paths. We infer a prediction confidence score by summing up the binary classification score of the symbolic predicates for each pixel. For each unique symbolic state $P_{t+1} \in \mathbf{P}_{t+1}$, we select the one with the highest confidence score and use its pixel location θ as the control parameters for the execution of skill π. This way, we could largely reduce the search complexity. The Appendix. 6.3 includes an empirical results on search time efficiency. The bolded variables in Alg. 1 are pixel-wise prediction results and θ is the skill parameter.

6.2.2 Tree-search task planning

With the learned state transition model, we could now solve the task planning problem with a tree search algorithm. In this work, we adopt a vanilla breadth-first search strategy to showcase the generalization capability of the transition model that is learned based on the pre-trained representation. We defer more complex search frameworks and efficient bi-level planning approaches to future works. Alg. 2 shows the pseudo-code for the search-based task planning procedure. Below we walk through the key steps in the procedure.

Given an image observation o and a symbolic goal g as inputs, the planner leverages the pre-trained observation encoder f_{enc} and symbolic decoder f_{syb} to search for a sequence of parameterized skill to reach the goal. We also provide a maximum search depth hyperparameter D to limit the boundless search space. The first step is to process image observation o into a set of object-level feature representation s_1 with the encoder network f_{enc}. The resulting feature state is then added to a priority queue $\mathcal{Q}$ along with the initialized current search depth d, confidence score c and history skill sequence $\mathcal{I}$. In each iteration, the priority queue pops out a set of leaf node states. The algorithm would exit the searching phase if the current search depth exceeds the maximum search depth. If the search continues, we then use the forward sampling process as introduced in Alg. 1 to get a set of next state predictions. Each future state candidate are added to the priority queue with the accumulated skill sequence and confidence score. If the state prediction reaches the task goal g, we add the skill sequence into a result buffer $\mathcal{R}$ and return the skill sequence that has the highest accumulated confidence score as the found task plan.

Algorithm 2: Search-based task planning

Output: $\{(\pi, \theta)_t\}_{t=1}^{T}$ ▷ A task plan
Input: o, g, D, f_{enc}, f_{syb}

Initialize the priority queue $\mathcal{Q} \leftarrow \emptyset$
Initialize the result buffer $\mathcal{R} \leftarrow \emptyset$
Initialize the skill sequence $\mathcal{I} \leftarrow \emptyset$
Initialize the skill sequence confidence score $c = 1$
Initialize the search depth $d = 0$
$s_1 = f_{enc}(o)$
$\mathcal{Q}$.push$((s_1, d, \mathcal{I}, c))$
while $\mathcal{Q}$ *is not empty* **do**
    $(s_t, d, \mathcal{I}, c) = \mathcal{Q}$.pop()
    if $d > D$ **then**
        break_loop()
    $\mathcal{F} =$ Sample_next_states(s_t)
    for (π, θ, s_{t+1}) *in* $\mathcal{F}$ **do**
        $\mathcal{I}$.append$((\pi, \theta))$
        $P_{t+1}, c_{t+1} = f_{syb}(s_{t+1})$
        if $P_{t+1} == g$ **then**
            $\mathcal{R}$.append$((\mathcal{I}, c \cdot c_{t+1}))$
        $\mathcal{Q}$.push$((s_{t+1}, d+1, \mathcal{I}, c \cdot c_{t+1}))$
        $\mathcal{I}$.delete$((\pi, \theta))$
return $\arg\max_c(\mathcal{R})$

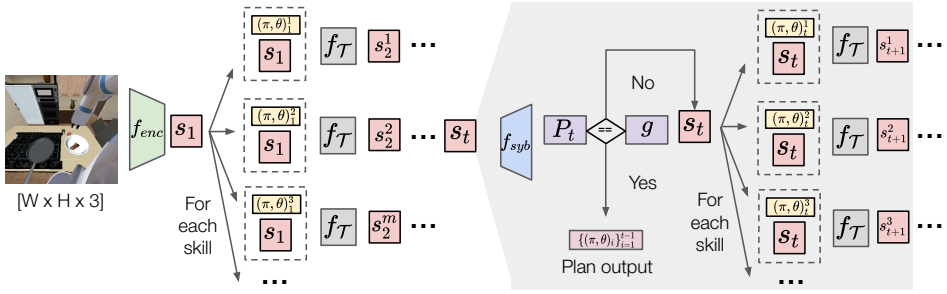


Figure 5: We develop a search-based planner based on the learned transition model. For each state node, the planner enumerates possible skill and parameter choices and traverse to their effect states. Each states is then decoded into symbolic predicates and compared with a task goal. A task is considered complete if and only if all predicate conditions are satisfied.

One of the major challenges of a vanilla searching framework is the time efficiency, especially that we want to search over both skills and parameters sequences. The unique design of our spatially-grounded transition mode can speed up the transition model and prune the less promising paths. First, the skill transition model takes the form of a fully convolutional network, with each pixel predicting the effect of a parameterized skill, the planner can get all the prediction results with a single forward pass through the network. Second, we can treat the raw output of the symbolic state prediction as a form of confidence score and find the 3D location of the pixel that has the highest confidence as the control parameters to execute the skill. This way, we could downgrade the searching complexity to state-wise because, for each unique object state, only one pixel that has the highest confidence score is added to the search tree. Downgrading the pixel-wise searching into state-wise is the key to improving our search speed and, more importantly, the classification confidence score is a byproduct of the learning process that does not require additional training or heuristics. We defer more complex pruning techniques to future works since it is not the main focus of our generalizable planning representation.

6.3 Search time breakdown

We record the search time cost of the vanilla pixel-wise searching and our pruned searching framework. The results are shown in Tab. 4. During the evaluation, we assign task goals that require different number of planning steps (from 1 to 5) in the *Rearrangement* task and use the same learned model to search for the task plans. Our approach is much more efficient than pixel-wise searching. That is because we leverage the predicted segmentation mask and classification confidence score to prune out the redundant pixels (Alg. 1), which largely decreases the search space. For each task goal, we test 20 rollouts and report the mean and 0.95 confidence interval in Tab. 4. For the searching that costs time longer than 1000 seconds, we directly report the search time as > 1000.

Table 4: Searching time cost

	Searching time breakdown (seconds)				
Plan steps	1	2	3	4	5
Pixel-wise	23.9±6.0	> 1000	> 1000	> 1000	> 1000
Ours	0.9±0.1	2.6±0.2	8.9±0.8	21.2±2.4	79.6±6.1