Deep Affordance Foresight: Planning Through What Can Be Done in the Future

Danfei Xu[1]*, **Ajay Mandlekar**[1], **Roberto Martín-Martín**[1], **Yuke Zhu**[2], **Silvio Savarese**[1], **Li Fei-Fei**[1]

Abstract

Robotic planning in realistic environments requires searching in large planning spaces. A powerful concept for guiding the search is affordance, which models what actions can be successful in a given situation. However, the classical notion of affordance is unsuitable for planning because it only informs the robot about the immediate outcome of actions instead of what actions are best for achieving a long-term goal. In this paper, we introduce a new affordance representation and a learning-to-plan framework that enable the robot to reason about the long-term effects of actions through modeling what actions are possible in the future. We show that our method, Deep Affordance Foresight, can effectively learn multi-step tool-use tasks and quickly adapt to a new longer horizon task. More materials and appendix available at https://sites.google.com/stanford.edu/daf

1 Introduction

Planning for multi-step tasks in real-world domains (e.g., making coffee in a messy kitchen) is a long-standing open problem in robotics. A key challenge is that the tasks require searching for solutions in high-dimensional planning spaces over extended time horizons. An approach to the challenge is to reduce the search problem into a *skill planning* problem: finding a sequence of motor skills applied to objects that will bring the environment to the desired state [1–6]. However, this reduction leads to a *combinatorial space* of possible skill parameters and object states, which can still be prohibitively expensive to plan with. On the other hand, only a small subset of skills can be carried out successfully at a given state in a typical manipulation domain. Thus to be effective, it is crucial for a planner to focus only on skills that are executable in a given environment state.

The ability to reason about what actions are possible in a given situation is commonly studied through *affordances*. Classically, an affordance is the *potential for actions* that an object "affords" to an agent [7]. For example, a mug is "graspable" and a door is "openable". These affordances can be refined to consider the exact parameterization of the action that may lead to success. For example, prior works in robotics have used affordance to represent possible grasping poses based on images of objects [8–11]. However, we argue that this classical notion of affordance is *myopic* and unsuitable for skill planning. This is because an affordance only implies the potential of carrying out an action, ignoring the action's effect on the subsequent plan towards a long-term goal. Consider the scene in Fig. 1: a myopic "graspable" affordance of the tool only implies that an agent can grasp and hold the tool, but different tasks may require different grasping poses. For example, using the tool to hook the red cube requires a different pose than for pushing the blue cube out of the tube.

In this work, we propose to use a learned environment dynamics model to extend the concept of affordances to represent the *future actions* that would become feasible if a certain action is executed at the current state, thereby informing the agent the best actions to take to achieve a long-term goal. For example, given a task goal of grasping the red cube, we aim to model whether a grasping pose

*[1]Stanford Vision and Learning Lab, [2]The University of Texas at Austin.

NeurIPS 2020 Workshop on Object Representations for Learning and Reasoning, Vancouver, Canada.

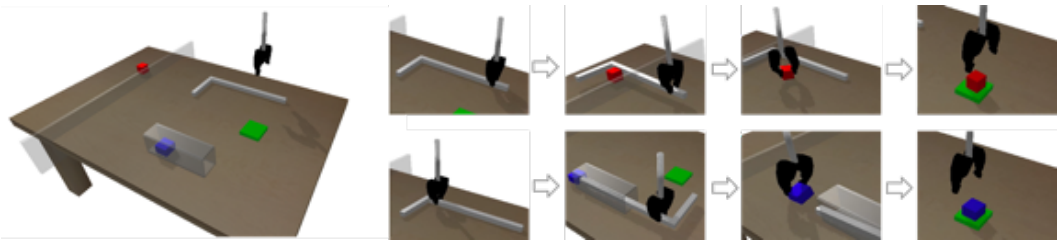

Figure 1: We evaluate our method in a tool-use domain (left). The two tasks shown on the right require the robot to use the tool differently depending on the task goal (red vs. blue on target). A virtual wall prevents the gripper from directly grasping the red cube.

would enable the robot to use the tool to hook the cube. This would subsequently depend on whether an enabled `hook` action would make a `grasp(red-cube)` action feasible.

To develop the method, we adopt a relaxed notion of affordances. Classically, an afforded action is both *feasible* (e.g., robot kinematics allows reaching the target grasping pose) and can *achieve a desired effect* (e.g., the tool being grasped stably). As discussed above, different task goals may require different action effects (e.g., different in-hand poses). Instead, we relax the definition of affordance to only model the feasibility of an action, and represent the effect of an action as the *expected affordances at future states*. In other words, we wish to model (1) what actions are feasible at a given state and (2) what actions would become feasible if an action is executed. This recursive structure allows composing chains of affordances to reason about long-horizon plans.

Concretely, we introduce Deep Affordance Foresight (DAF), a *learning-to-plan* method that incrementally builds environment models around the affordances of parameterized motor skills [12], and learns to plan for multi-step tasks through trial-and-error. DAF learns a latent dynamics model to predict future latent states conditioned on sampled skill plans and an affordance prediction model to evaluate skill affordances both at the current and future latent states. DAF can use both models together to select multi-step plans that are most likely to achieve a task goal. Moreover, DAF can be trained end-to-end from pixel observations, allowing DAF to model complex dynamics such as pouring liquid, for which manually defining an affordance is hard.

We present evaluation results on the *Tool-Use* domain as shown in Fig. 1, where a free gripper robot must use the hook-like tool to fetch the red and blue cubes and put them on the green target, evaluating the capabilities of our robot to differentiate between the same affordance (graspable) based on future task needs. In Appendix, we present result on *Kitchen* domain. It requires the robot to plan through complex dynamics such as pouring liquid to complete multi-stage tasks of serving tea or coffee, highlighting the ability of DAF of combining and reusing learned affordances for other tasks.

2 Method

This section describes the affordance-based planning problem and introduces the learning-to-plan method Deep Affordance Foresight. We include detailed discussions on related methods in Appendix.

Problem setup. We consider partially observable domains with observation space O, state space S, parameterized skills Π (described later), and transition dynamics $\mathcal{T} : S \times \Pi \to Dist(S)$. We assume a finite set of goals G. Each $g \in G$ is a binary condition function $g : S \to \{0, 1\}$ indicating if a state is in a goal state set S_g. The objective is to reach the goal by the end of an episode.

Following prior work [13], we define a parameterized skill [12] by a policy $\pi(s, \theta)$ modulated by a set of parameters $\theta \in R^D$. For example, a grasping skill (π) can be parameterized by 3D grasping positions (θ), and the policy can execute a planned grasping motion. An important feature of motion planning-based skills that we leverage in this work is that we can check if a skill is *feasible to execute* before executing it. The feasibility can be determined through robot kinematic constraints or if a skill motion plan would result in unintended collision between the robot and the environment. For example, in the setup shown in Fig.1, grasping the red cube directly is infeasible due to the kinematic constraint defined by the virtual wall, and grasping the blue cube would collide the gripper with the pipe, which is also infeasible. Skill feasibility checkers are commonly used to prune skill samples in solving a larger task-and-motion-planning (TAMP) problem [1–6]. TAMP methods typically require

knowledge of ground truth states and an environment dynamics model. Instead, we leverage skill feasibilities to develop a method that can learn to plan in an environment with unknown dynamics.

2.1 Planning with Affordances

Here we formally define our affordance representation and introduce a planning problem setup.

Definition 1 (Affordance $\mathcal{A}$): *Given a skill (π, θ), we define an affordance as $\mathcal{A}_{\pi,\theta} = \{s \in S | (\pi, \theta)$ is feasible at $s\}$. We use $\mathcal{A}_{\pi,\theta}(s) = \mathbb{1}[(s, \pi, \theta) \in \mathcal{A}_{\pi,\theta}]$ to denote if state s affords (π, θ).*

To formalize a planning problem using $\mathcal{A}$, we first show how to compute the probability of *plan completion* from some initial state distribution. A length-N plan p belongs to the set $\mathcal{P}_N = \{(\pi_i, \theta_i)\}_{i=1}^N | (\pi_i, \theta_i) \in \Pi, N \in \mathbb{Z}^+\}$. A particular plan $p \in \mathcal{P}_N$ is then a sequence of parametrized skills $\{(\pi_1, \theta_1), \ldots, (\pi_N, \theta_N)\}$. Without loss of generality, we assume fixed plan length and omit the subscript N. Given a plan p, we denote the induced state distribution at each step i as $Z_i(\cdot; p)$. Given an initial state distribution $Z_0(\cdot; p)$, $Z_{i>0}(\cdot; p)$ can be expressed recursively as:

$$Z_i(s'; p) \propto \sum_{s \in S} \mathcal{T}(s' | s, \pi_i, \theta_i) Z_{i-1}(s; p) \mathcal{A}_{\pi_i, \theta_i}(s) \tag{1}$$

where (π_i, θ_i) is the skill at step i of plan p. We can compute the probability of completing the plan p (being able to execute each skill in the plan) starting from Z_0 as:

$$C_{plan}(p = \{(\pi_1, \theta_1), \ldots, (\pi_N, \theta_N)\}) = \sum_{s \in S} Z_{N-1}(s; p) \mathcal{A}_{\pi_N, \theta_N}(s) \tag{2}$$

Next we show how to construct plans towards a goal $g \in G$. The key idea is to reinterpret g using affordance. Recall that g is a binary function on whether a state belongs to its goal state set S_g. We say that a plan $p = \{(\pi_i, \theta_i)\}_{i=1}^N$ is directed towards goal g if the last skill in the plan can be executed in a goal state, i.e., $A_N \subseteq S_g$.

Definition 2 (Goal-directed plans $\mathcal{P}_g$): *Given a goal $g \in G$ and its goal state set S_g, we define the goal-directed plans of g as $\mathcal{P}_g \subseteq \mathcal{P}$ such that $\forall \{(\pi_i, \theta_i)\}_{i=1}^N \in \mathcal{P}_g, \mathcal{A}_{\pi_N, \theta_N} \subseteq S_g$.*

Finally, the problem of searching for a best skill plan towards goal $g \in G$ is:

$$\arg\max_{p \in \mathcal{P}_g} C_{plan}(p) \tag{3}$$

While it is possible to find exact optimal solutions by computing Eq. (3) from state space S and transition function T, we aim at realistic domains in which we have access to neither. In the following, we present a method that learns to plan in an unknown environment by modeling affordances.

2.2 Deep Affordance Foresight (DAF)

We base our learning-to-plan method on a model-based reinforcement learning (MBRL) formulation. To behave in an environment with unknown dynamics, an MBRL agent needs to learn both a dynamics model and a cost function to predict plan-induced future states and evaluate plan costs. Our method can be viewed as building a *partial model* [14] of the environment based on affordances. Prior works on building partial models have shown remarkable results on learning dynamics [15–18]. For example, PlaNet [18] combines the dynamics and cost modeling by predicting multi-step future rewards through a learned dynamics model. However, these works rely on modeling task-specific quantities such as rewards. In contrast, our affordance modeling is *task-agnostic*: a grasping skill is afforded regardless of the final task goal. This allows our method to share learned affordance models among plans with different goals, which can improve sample efficiency and task performance.

Concretely, we jointly train a latent dynamics model and an affordance prediction model to predict skill affordances at future states. We use model-predictive control (MPC) to plan in the learned latent space and evaluate the proposed plans by computing plan completion probabilities (Eq.3) from predicted affordances. Our method iteratively collects data from environment using planning and trains the two models on the gathered data. Below we provide details of the components.

Latent dynamics model. We consider experience sequences $\{(o_t, \pi_t, \theta_t, a_t)\}_{t=1}^T$, environment observation o_t, a skill (π_i, θ_i) that the robot attempted to execute at time t, and the resulting binary affordance value a_t. Following PlaNet [18], we project observation o_t to a latent encoding z_t

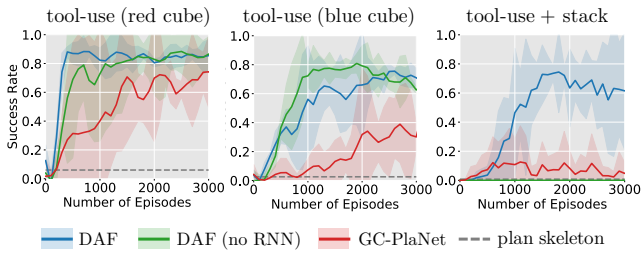

Figure 2: Results on the jointly learning the two *tool-use* tasks shown in Fig. 1 and a combined *tool-use + stack* task, where the robot has to use the tool to get both cubes and stack them on the green target. We compare our method (DAF) and a goal-conditional variant of PlaNet [18] (GC-PlaNet).

using an observation encoder $z_t = f_{enc}(o_t)$. The encoder can be a multi-layer perception for low-dimensional observations and deep CNN for image observations. We make a simplified assumption that the latent dynamics is deterministic [19] and construct a deterministic transition model $\hat{z}_{t+1} = f_{trans}(z_t, \pi_t, \theta_t)$. We also explore a recurrent transition model $h_{t+1} = f_{trans}(h_t, z_t, \pi_t, \theta_t)$ with decoder $\hat{z}_{t+1} = f_{dec}(h_{t+1})$ that shows better empirical performance on long-horizon tasks.

Learning dynamics by predicting affordances. Given latent experiences $\{(z_t, \pi_t, \theta_t, a_t)\}_{t=1}^{T}$, we train a binary classifier $\hat{a}_t = f_{\mathcal{A}}(z_t, \pi_t, \theta_t)$ to predict whether a latent state z_t affords the skill (π_t, θ_t). We train the affordance model jointly with the latent dynamics model. The simplest way is to learn from one-step transitions: predicting $\hat{a}_t$ and $\hat{a}_{t+1}$ from (z_t, π_t, θ_t) and $(f_{trans}(z_t, \pi_t, \theta_t), \pi_{t+1}, \theta_{t+1})$, respectively. However, as shown in [17, 18], the latent dynamics model learned from one-step transitions is often not accurate enough for long-horizon planning. Hence we adopt the *overshooting* [17] technique and optimize f_{trans} and $f_{\mathcal{A}}$ over multi-step affordance predictions.

Planning with MPC We use a standard model-predictive control (MPC) strategy to plan with the learned latent dynamics and affordance models. Given a goal g, the MPC planner optimizes $arg\min_{\pi,\theta} C(\{(z_i, \pi_i, \theta_i)\}_{i=1}^{H})$, with plan cost function C, goal-directed plans $(\pi_{1:H}, \theta_{1:H}) \in \mathcal{P}_g$, and the induced latent sequences $z_{1:H}$ over a planning horizon H. We use the negative of the plan completion probability defined in Eq. 3 as the plan cost. Because we assume deterministic transition, we can conveniently compute the cost as $C(\{(z_i, \pi_i, \theta_i)\}_{i=1}^{H}) = -\prod_{i=1}^{H} f_{\mathcal{A}}(z_i, \pi_i, \theta_i)$.

3 Experiments

We use the *Tool-Use* environment (Fig. 1) to analyze the key traits of our method, and present additional results on task knowledge transfer and image-input evaluation in Appendix.

Task setup We evaluate on two sets of tasks. The first is **tool-use** with two task goals: use the tool to fetch and place either the blue or the red cube on the green target. A virtual wall prevents the gripper from directly grasping the red cube - the robot must use the tool to pull it across the wall. The blue cube is in a pipe - the robot needs to use the tool to push it out of the pipe first. The two task goals are sampled randomly each episode. **tool-use + stack** is a longer task of stacking the two cubes on top of the target, which requires the robot to use the tool differently to fetch both cubes.

Architectures and baselines To isolate the effects of our method and design choices, we focus on object pose input space in this domain. We evaluate our method with recurrent dynamics (**DAF**) and MLP-based dynamics (**DAF (no RNN)**). We compare with a goal-conditional variant of PlaNet [18] (**GC-PlaNet**) by conditioning the learned reward model on a task ID. To facilitate fair comparisons, we remove the auxiliary observation model and the stochastic component in the recurrent dynamics of PlaNet, and match all other architecture choices to DAF. We also include a hard-coded baseline (**plan skeleton**) that executes ground truth plan skeletons (discrete skills) with random skill parameters.

Results Left two plots in Fig. 2 show the results of jointly learning the two *tool-use* tasks. We see that DAF converges to high success rate within 1000 episodes for both tasks. GC-PlaNet performs competitively on the red-cube task but peaks at 0.4 success rate for the blue cube task, which requires more careful grasping pose choice for poking the blue cube out of the pipe. The rightmost plot in Fig. 2 shows the results on a longer task that requires the robot to fetch the two cubes and stack them on the green target. On the left figure, we observe that DAF reaches peak performance of 0.7 success rate at episode 1500, whereas GC-PlaNet converges at 0.1 success rate. Notably, DAF without RNN dynamics falls flat on this task, echoing the findings in [17, 18] that recurrent dynamics is crucial for modeling long tasks.

References

[1] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *ICRA*, 2011.

[2] L. P. Kaelbling and T. Lozano-Pérez, "Integrated task and motion planning in belief space," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1194–1227, 2013.

[3] L. P. Kaelbling and T. Lozano-Pérez, "Pre-image backchaining in belief space for mobile manipulation," in *Robotics Research*, pp. 383–400, Springer, 2017.

[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*, vol. 30, pp. 440–448, 2020.

[5] M. Toussaint, "Logic-geometric programming: An optimization-based approach to combined task and motion planning,"

[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. J. Gibson, "The theory of affordances," *Hilldale, USA*, vol. 1, no. 2, 1977.

[8] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *arXiv preprint arXiv:1703.09312*, 2017.

[9] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 4243–4250, IEEE, 2018.

[10] R. Detry, D. Kraft, O. Kroemer, L. Bodenhagen, J. Peters, N. Krüger, and J. Piater, "Learning grasp affordance densities," *Paladyn, Journal of Behavioral Robotics*, vol. 2, no. 1, pp. 1–17, 2011.

[11] P. Mandikal and K. Grauman, "Dexterous robotic grasping with object-centric visual affordances," *arXiv preprint arXiv:2009.01439*, 2020.

[12] B. Da Silva, G. Konidaris, and A. Barto, "Active learning of parameterized skills," in *International Conference on Machine Learning*, pp. 1737–1745, 2014.

[13] 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)*, pp. 526–533, IEEE, 2018.

[14] E. Talvitie and S. P. Singh, "Simple local models for complex dynamical systems," in *Advances in Neural Information Processing Systems*, pp. 1617–1624, 2009.

[15] A. Dosovitskiy and V. Koltun, "Learning to act by predicting the future," *ICLR*, 2017.

[16] J. Oh, S. Singh, and H. Lee, "Value prediction network," in *Advances in Neural Information Processing Systems*, pp. 6118–6128, 2017.

[17] B. Amos, L. Dinh, S. Cabi, T. Rothörl, S. G. Colmenarejo, A. Muldal, T. Erez, Y. Tassa, N. de Freitas, and M. Denil, "Learning awareness models," *International Conference on Learning Representations*, 2018.

[18] 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*, pp. 2555–2565, PMLR, 2019.

[19] L. Buesing, T. Weber, S. Racaniere, S. Eslami, D. Rezende, D. P. Reichert, F. Viola, F. Besse, K. Gregor, D. Hassabis, *et al.*, "Learning and querying fast generative models for reinforcement learning," *arXiv preprint arXiv:1802.03006*, 2018.

1 Additional Related Work

1.1 Affordance

Affordances have a rich history in fields such as robotics, psychology, computer vision, and reinforcement learning (RL). In robotics, many have used affordance as a representation prior, e.g., predicting grasping poses [1–4], traversable regions [5], and exploration [6]. As discussed in the introduction, such a notion of affordance cannot be easily adapted for planning due to its myopic nature.

Other works have explored learning affordance with respect to a task goal. For example, task-aware grasping [7–10] predicts grasping poses in anticipation of a task goal (e.g., tool-use [8]). However, each learned affordance representation is tied to a specific task goal (e.g., a specific way of using the tool). In contrast, our affordance representation can be flexibly composed to reason about diverse long-horizon plans with different task goals.

Theoretical works in RL have formalized affordance for sequential decision making [11–14]. Closest to us is Khetarpal *et al.* [14] that introduces the notion of "intent". Intent specifies the desired future state distribution of an afforded action. By modeling the satisfiability of intents, they build partial models of environments that allow efficient planning. A key limitation of this work is that the intents are complex functions that are hand-defined, e.g., a "Move Left" intent checks the agent's x-position change in a grid world. Such detailed conditions are tedious and difficult to specify robotics domains. For example, in our *Tool-Use* domain, one would need to define different intents for grasping the tools at different locations. This also requires specifying the precise relative poses between the gripper and the object. In contrast, our affordance represents action feasibilities, which can be checked via robot kinematics or a crude collision detector and shared across all tasks in the same domain.

1.2 Task and Motion Planning

Our definition of affordance is closely related to "preconditions" or "preimages" in Task and Motion Planning (TAMP) [15–20]. Most TAMP methods require fully-specified planning space and dynamics models. Recent works proposed to build dynamics models by characterizing the preconditions and effects of skills [21–24]. For example, Kaelbling *et al.* [21] proposes to learn the preimage of a skill given desired effects through trial-and-error. However, they still require predefined planning spaces such as object poses. Our method plans in a learned latent space with image input. This enables our method to model complex dynamics such as pouring liquid, for which manually designing a planning space would be challenging.

Our planning formulation is heavily inspired by works from Konidaris and colleagues [25–28], which aim to build compact symbolic environment models by capturing skill pre-condition and effect distributions through interaction. The resulting symbolic models are provably both necessary and sufficient to verify whether a skill plan is *sufficing* [27], meaning that the plan is executable (analogous to plan completion probability defined in the main text and leads to a goal (analogous to our *goal-directed plans*). However, these symbolic representations, once built, are confined to a fixed domain. A recent work [29] attends to this limitation by building symbols on an agent-egocentric space that facilitates cross-domain generalization. Our work offers a different perspective and propose a latent planning formulation that exploits the generalization ability of deep neural networks. Our idea of composing affordance for planning is also related to option chaining [30, 31], although we do not explicitly construct skill trees.

Our method is related to works that learn to predict TAMP plan feasibilities from observations [32–34]. For example, Deep Visual Reasoning [34] learns to generate feasible plan skeletons for Logical Geometric Programming (LGP) solvers. A drawback of these approaches is that they rely on TAMP/LGP planners that can already solve the task to generate planning supervisions for training. In contrast, our method learns through trial-and-error.

1.3 Learning to Plan

Our method is related to learning dynamics models for model-based RL [35–38]. Most recent works have focused on building complete environment models directly from the raw observation space. However, learning to make accurate predictions with high-dimensional observations is still challenging [35, 37, 39], especially for visually complex long-horizon tasks. Instead, our method

builds a *partial model* [40] of the environment on skill affordances, which are low-dimensional and amenable to long-horizon planning.

Prior works on planning with partial models focuse on predicting either reward or quantities that are tied to a task [38, 41–43]. For example, PlaNet [38] learns to predict future rewards and observations through a latent dynamics model. It is difficult for these methods to share reward models among different tasks and transfer to new tasks. In contrast, our composable affordances are defined independent of a final task goal. This enables our learned affordance and dynamics models to be shared and reused among different tasks to improve data efficiency and task performance.

2 Algorithms

Algorithm 1 DEEP AFFORDANCE FORESIGHT

Hyperparameters:
Batch size B
Number of training iterations K
Number of rollouts R
Task horizon T
Overshooting length H
Inputs:
$z = f_{enc}(o)$ ▷ observation encoder
$\hat{z}_{t+1} = f_{trans}(z_t, \pi_t, \theta_t)$ ▷ latent transition model
$\hat{a}_t = f_{\mathcal{A}}(z_t, \pi_t, \theta_t)$ ▷ affordance model
$\pi_t \sim f_{\pi}(z_t)$ ▷ skill skeleton proposal model
$\mathcal{D}$ ▷ replay buffer with seeding data
`env` ▷ environment

Start
while *Not Converged* **do**
  `// Model fitting`
  for update steps $k \leftarrow [1...K]$ **do**
    Sample experience sequence batch $\{o^i, \pi^i_{1:H}, \theta^i_{1:H}, a^i_{1:H}\}_{i=1}^{B} \sim \mathcal{D}$
    `// Omitting batch index for clarity`
    $z_1 \leftarrow f_{enc}(o)$
    for $h \leftarrow [1...(H-1)]$ **do**
      $z_{h+1} \leftarrow f_{trans}(z_h, \pi_h, \theta_h)$
    end for
    Train $f_{\mathcal{A}}(z_{1:H}, \pi_{1:H}, \theta_{1:H})$ against affordance label $a_{1:H}$ with BCE loss.
    Train $f_{\pi}(z_{1:H})$ against skills that were executed with discrete-distribution MLE.
  end for
  `// Experience collection`
  for rollout iteration $r \leftarrow [1...R]$ **do**
    $o, g \leftarrow$ `env.reset()` ▷ get initial observation o and goal g
    for rollout step $t \leftarrow [1...T]$ **do**
      $\pi, \theta \leftarrow$ PLANWITHAFFORDANCE ▷ plan for goal g
      $a \leftarrow$ `env.skill_is_executable`(π, θ)
      if $a = 1$ **then**
        $o \leftarrow$ `env.step`(π, θ)
      end if
      Append experience (o, π, θ, a) to $\mathcal{D}$
    end for
  end for
end while

Algorithm 2 PLANWITHAFFORDANCE

Hyperparameters: Planning horizon H, Number of skill samples N
Inputs:
$z = f_{enc}(o)$ ▷ observation encoder
$\hat{z}_{t+1} = f_{trans}(z_t, \pi_t, \theta_t)$ ▷ latent transition model
$\hat{a}_t = f_{\mathcal{A}}(z_t, \pi_t, \theta_t)$ ▷ affordance model
$\pi_t \sim f_{\pi}(z_t)$ ▷ skill skeleton proposal model
o ▷ current environment observation
$\theta \sim param(\pi)$ ▷ random skill parameter sampler
P_g ▷ set of goal-directed plans for goal g

Start
`plans` $\leftarrow []$ ▷ sampled plans
`affs` $\leftarrow []$ ▷ step-wise affordances
$z_1 \leftarrow f_{enc}(o)$ ▷ encode observation to latent
$z_1^{1:N} \leftarrow repeat(z_1, N)$ ▷ repeat latent N times

`// Shooting method with sampled skills`
for $i \leftarrow [1, ..., H]$ **do**
  $\pi_i^{1:N} \sim f_{\pi}(z_i^{1:N})$ ▷ Take skill skeleton samples
  $\theta_i^{1:N} \sim param(\pi_i^{1:N})$ ▷ Take random parameter samples for each skill
  $a_i^{1:N} = f_{\mathcal{A}}(z_i, \pi_i^{1:N}, \theta_i^{1:N})$ ▷ Compute affordance values
  `plans` $\leftarrow$ `plans` $\cup (\pi_i^{1:N}, \theta_i^{1:N})$
  `affs` $\leftarrow$ `affs` $\cup\, a_i^{1:N}$
  $z_{i+1}^{1:N} \leftarrow f_{trans}(z_i^{1:N}, \pi_i^{1:N}, \theta_i^{1:N})$ ▷ Forward latent states
end for

`// Evaluating the cost of each plan using affordances`
for $k \leftarrow [1, ..., N]$ **do**
  $(\pi_{1:H}, \theta_{1:H}) \leftarrow$ `plans`$[k]$ ▷ k-th plan in `plans`
  $a_{1:H} \leftarrow$ `affs`$[k]$ ▷ k-th plan-wise affordances in `affs`
  $c_k \leftarrow \begin{cases} -\prod_{t=1}^{H} a_t & \text{if } (\pi_{1:H}, \theta_{1:H}) \in \mathcal{P}_g \\ \infty & \text{otherwise} \end{cases}$ ▷ goal-directed plan cost (Eq.3)
end for
$k \leftarrow \arg\min_{k=\{1...N\}}(c_k)$ ▷ get the lowest-cost plan index
$\pi^*, \theta^* \leftarrow$ `plans`$[k][0]$ ▷ first skill of the chosen plan

return π^*, θ^*

3 Additional Details and Results on Tool-Use

Parameterized skills The agent is provided with four parameterized motor skills: `grasp`, `place`, `hook`, and `poke`. `grasp` executes top-down grasps parameterized by 3D grasping locations relative to the target object. `place` sets a grasped object onto a surface parameterized by the relative 2D location between the object and the surface. Both `hook` and `poke` moves the object-in-hand along a trajectory with parameterized start and end positions. The motion trajectories are generated using RRT-based [44] motion planners. We use additional "no-op" skills to specify goals. no-op skills are skills that have affordance sets but do not incur changes to the environment if executed. Each goal (*red-on-target, blue-on-target, stack-red-blue-on-target*) is associated with a no-op *goal skill* for which the affordance set is equal to the goal state set. In other words, the goal skills are only afforded at their corresponding goal states.

Skill feasibilities Skill feasibilities are determined by pre-defined workspace constraints (e.g., the gripper cannot go beyond the virtual wall) and PyBullet's built-in collision detector for checking if a motion plan would cause unintended collisions between the robot and the environment. For example,

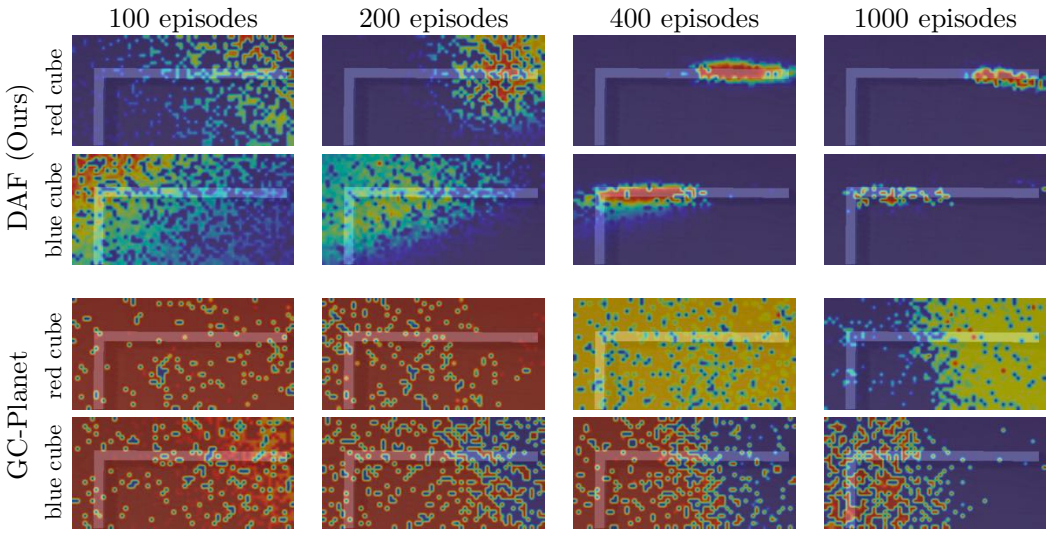

Figure 1: Visualizing the plan score predictions over the course of learning the two *tool-use* tasks. Each column shows the prediction made by models trained with N number of actively collected episodes. Each pixel of the heat map shows the predicted score of a plan that starts with the `grasp` skill parameterized by the corresponding x, y location.

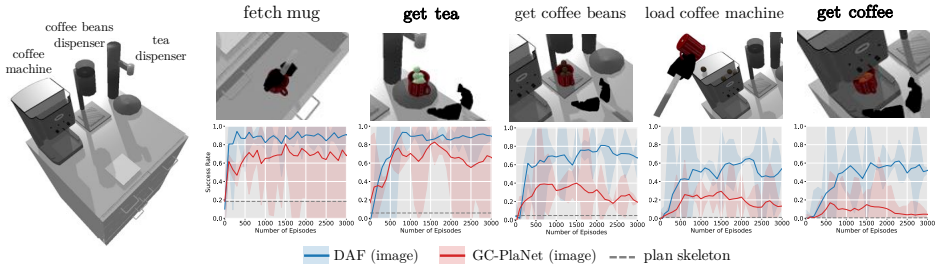

Figure 2: Setup (left) and results on key stages of the *Kitchen* task domain, which features challenging dynamics such as liquid-like objects (load coffee machine). We compare our method DAF and GC-PlaNet on learning to achieve the two final goals, *get coffee* and *get tea*, with raw image input.

we consider a grasping skill that would result in the gripper colliding with the table as infeasible. Conversely, grasping skills that do not touch any object at all are considered to be feasible. In the real world, collision detection can be implemented through a depth-based octomap [45].

Architectures and baselines To isolate the effects of our method and design choices, we focus on object pose input space in this domain. We evaluate our method with recurrent dynamics (**DAF**) and MLP-based dynamics (**DAF (no RNN)**). All other components, f_{enc}, $f_{\mathcal{A}}$, and f_{trans}, are MLPs. We compare with a goal-conditional variant of PlaNet [38] (**GC-PlaNet**) by conditioning the learned reward model on a task ID. To facilitate fair comparisons, we remove the auxiliary observation model and the stochastic component in the recurrent dynamics of PlaNet, and match all other architecture choices to DAF. These components are orthogonal to the comparison and adding them to our framework will be explored in future work. We also include a hard-coded baseline (**plan skeleton**) that executes ground truth plan skeletons (discrete skills) with random skill parameters in open-loop to highlight that the tasks we consider require intelligent skill parameter selection in conjunction with the correct skill sequence to solve consistently.

Analyzing learned affordances To get a better idea of how the affordance representations learned by DAF develop over the training process, we visualize the plan scores computed from the learned affordance values. Specifically, we visualize plan scores at the beginning of both *tool-use* tasks, since both task goals require the robot to grasp the tool. As shown in Fig. 1, each pixel in the overlaid heatmap (red-high, blue-low) indicates the normalized score of a plan that starts with a `grasp` skill

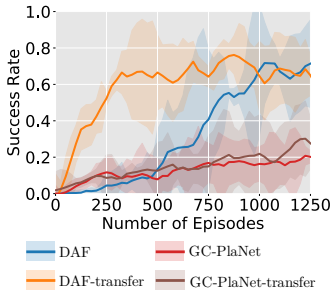

Figure 3: We compare learning the standalone *get coffee* task from scratch vs. finetuning from models pre-trained on the *get tea* task. Our task-agnostic affordance representation allows DAF to transfer knowledge about how to fetch the mug from the short get-tea task to the longer get-coffee, enabling DAF to learn a performant policy within 300 episodes.

parameterized by the corresponding x, y location with a constant z-height. Each column shows the visualization produced by models trained with certain number of actively collected episodes.

We see that DAF is able to rapidly learn meaningful affordance representations with respect to each task goal with as little as 400 actively collected episodes. In contrast, GC-PlaNet's plan score prediction remains noisy even at 1000 episodes. One may also notice that DAF continues to shrink its "good grasp" predictions at episodes 1000. This is because while the plan scores are only used to decide the next skill to execute, they are computed from the *future skills affordances* over the rest of a plan. As the training progresses, the agent starts to reach later stages of the task and get better estimates of the skill affordances later in the plan, which will in turn influence the plan scores even at the beginning of an episode. This behavior highlights the key difference between our future-aware affordance representation and traditional affordances that only model myopic effects of actions.

3.1 Results on Kitchen Domain

Compared to TAMP-like methods that require a hand-defined planning space (e.g. object poses), our method can learn end-to-end with raw image inputs. This allows our method to learn to plan through complex non-rigid dynamics such as pouring liquid. To test this capability, we task the robot to serve tea and coffee in the *Kitchen* domain as shown on the left side of Fig 2 with only visual inputs.

Setup The domain has two tasks of varying difficulties: in the simpler **get tea** task, the robot needs to open the drawer, fetch the mug, use the platform to reorient the grasp and set the mug at the correct location beneath the tea dispenser tap to get tea. In a more challenging **get coffee** task, the robot needs to fetch the mug from the drawer, use the mug to get the coffee beans from the dispenser, then pour the coffee beans into the coffee machine, and finally set the mug beneath the coffee machine dispenser to get coffee. The two goals are sampled randomly each episode. We use small spherical beads to approximate liquid dynamics.

The robot is equipped with the following parameterized skills: `grasp` is parameterized with gripper-object distance and two discrete grasping orientations: side and top; `place` with the relative location between the object to be place and a surface object; `pour` is parameterized by the relative position between the object-in-hand and the target container and a pouring angle; `open` is parameterized by a grasp location and a distance to pull along a given direction. We use no-op to skills to represent the goals.

The environment observations are RGB images rendered at 128×128 resolution from the perspective shown in Fig. 2. Accordingly, we change f_{enc} to a ResNet architecture [46] followed by a a Spatial-Softmax layer [47] and an MLP. The remaining components are the same as in *Tool-Use*.

Results As shown in Fig. 2, DAF is able to jointly learn both tasks, get tea and get coffee, with high success rate from only raw image inputs. Moreover, we observe that both DAF and GC-PlaNet can solve the simpler *get tea* tasks, with DAF having significantly lower performance variance. For the more challenging *get coffee* task, DAF learns to fill the mug with coffee beans within 500 episodes and learns to get coffee from the coffee machine at 0.6 success rate in 1500 episodes, whereas GC-PlaNet plateaus at $<$0.2 success rate.

We in addition highlight that the task-agnostic affordance representation allows DAF to share the learned affordance and latent dynamics models across tasks. To verify this claim, we compare learning the standalone *get coffee* task a) from scratch and b) finetuning from models *pretrained* on the *get tea* task. DAF should be able to transfer the affordances for opening the drawer and fetching the mug from the short *get tea* task to the longer *get coffee* task. To remove conflating factors such as image encoders, we evaluate the models on a hand-defined feature space of object poses and the number of {coffee, coffee bean, tea} beads contained in each object. As Fig. 3 shows, DAF pretrained with the *get tea* task is able to learn the *get coffee* task with only 300 actively collected episodes. In contrast, the pretrained GC-PlaNet shows no significant improvement compared to learning from scratch.

References

[1] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *arXiv preprint arXiv:1703.09312*, 2017.

[2] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 4243–4250, IEEE, 2018.

[3] R. Detry, D. Kraft, O. Kroemer, L. Bodenhagen, J. Peters, N. Krüger, and J. Piater, "Learning grasp affordance densities," *Paladyn, Journal of Behavioral Robotics*, vol. 2, no. 1, pp. 1–17, 2011.

[4] P. Mandikal and K. Grauman, "Dexterous robotic grasping with object-centric visual affordances," *arXiv preprint arXiv:2009.01439*, 2020.

[5] E. Ugur, M. R. Dogar, M. Cakmak, and E. Sahin, "The learning and use of traversability affordance using range images on a mobile robot," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 1721–1726, IEEE, 2007.

[6] T. Nagarajan and K. Grauman, "Learning affordance landscapes forinteraction exploration in 3d environments," *arXiv preprint arXiv:2008.09241*, 2020.

[7] H. Dang and P. K. Allen, "Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1311–1317, IEEE, 2012.

[8] K. Fang, Y. Zhu, A. Garg, A. Kurenkov, V. Mehta, L. Fei-Fei, and S. Savarese, "Learning task-oriented grasping for tool manipulation from simulated self-supervision," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 202–216, 2020.

[9] D. Song, K. Huebner, V. Kyrki, and D. Kragic, "Learning task constraints for robot grasping using graphical models," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1579–1585, IEEE, 2010.

[10] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4238–4245, IEEE, 2018.

[11] D. Abel, G. Barth-Maron, J. MacGlashan, and S. Tellex, "Toward affordance-aware planning," in *First Workshop on Affordances: Affordances in Vision for Cognitive Robotics*, 2014.

[12] D. Abel, D. E. Hershkowitz, G. Barth-Maron, S. Brawner, K. O'Farrell, J. MacGlashan, and S. Tellex, "Goal-based action priors," in *Twenty-Fifth International Conference on Automated Planning and Scheduling*, 2015.

[13] F. Cruz, S. Magg, C. Weber, and S. Wermter, "Training agents with interactive reinforcement learning and contextual affordances," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 8, no. 4, pp. 271–284, 2016.

[14] K. Khetarpal, Z. Ahmed, G. Comanici, D. Abel, and D. Precup, "What can i do here? a theory of affordances in reinforcement learning," 2020.

[15] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *ICRA*, 2011.

[16] L. P. Kaelbling and T. Lozano-Pérez, "Integrated task and motion planning in belief space," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1194–1227, 2013.

[17] 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*, vol. 30, pp. 440–448, 2020.

[18] M. Toussaint, "Logic-geometric programming: An optimization-based approach to combined task and motion planning,"

[19] 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.

[20] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *arXiv preprint arXiv:2010.01083*, 2020.

[21] L. P. Kaelbling and T. Lozano-Pérez, "Learning composable models of parameterized skills," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 886–893, IEEE, 2017.

[22] Z. Wang, C. R. Garrett, L. P. Kaelbling, and T. Lozano-Pérez, "Active model learning and diverse action sampling for task and motion planning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4107–4114, IEEE, 2018.

[23] H. M. Pasula, L. S. Zettlemoyer, and L. P. Kaelbling, "Learning symbolic models of stochastic domains," *Journal of Artificial Intelligence Research*, vol. 29, pp. 309–352, 2007.

[24] V. Xia, Z. Wang, and L. P. Kaelbling, "Learning sparse relational transition models," *International Conference on Learning Representations*, 2018.

[25] G. D. Konidaris, L. P. Kaelbling, and T. Lozano-Perez, "Constructing symbolic representations for high-level planning," 2014.

[26] G. Konidaris, L. P. Kaelbling, and T. Lozano-Perez, "Symbol acquisition for probabilistic high-level planning," AAAI Press/International Joint Conferences on Artificial Intelligence, 2015.

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

[28] 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)*, pp. 526–533, IEEE, 2018.

[29] S. James, B. Rosman, and G. Konidaris, "Learning portable representations for high-level planning," *ICML*, 2020.

[30] G. Konidaris and A. G. Barto, "Skill discovery in continuous reinforcement learning domains using skill chaining," in *Advances in neural information processing systems*, pp. 1015–1023, 2009.

[31] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto, "Robot learning from demonstration by constructing skill trees," *The International Journal of Robotics Research*, vol. 31, no. 3, pp. 360–375, 2012.

[32] A. M. Wells, N. T. Dantam, A. Shrivastava, and L. E. Kavraki, "Learning feasibility for task and motion planning in tabletop environments," *IEEE robotics and automation letters*, vol. 4, no. 2, pp. 1255–1262, 2019.

[33] D. Driess, O. Oguz, J.-S. Ha, and M. Toussaint, "Deep visual heuristics: Learning feasibility of mixed-integer programs for manipulation planning," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[34] 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," *arXiv preprint arXiv:2006.05398*, 2020.

[35] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh, "Action-conditional video prediction using deep networks in ATARI games," in *NIPS*, pp. 2863–2871, 2015.

[36] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to poke by poking: Experiential learning of intuitive physics," in *Advances in Neural Information Processing Systems*, pp. 5074–5082, 2016.

[37] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *ICRA*, pp. 2786–2793, IEEE, 2017.

[38] 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*, pp. 2555–2565, PMLR, 2019.

[39] M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller, "Embed to control: A locally linear latent dynamics model for control from raw images," in *Advances in neural information processing systems*, pp. 2746–2754, 2015.

[40] E. Talvitie and S. P. Singh, "Simple local models for complex dynamical systems," in *Advances in Neural Information Processing Systems*, pp. 1617–1624, 2009.

[41] A. Dosovitskiy and V. Koltun, "Learning to act by predicting the future," *ICLR*, 2017.

[42] J. Oh, S. Singh, and H. Lee, "Value prediction network," in *Advances in Neural Information Processing Systems*, pp. 6118–6128, 2017.

[43] B. Amos, L. Dinh, S. Cabi, T. Rothörl, S. G. Colmenarejo, A. Muldal, T. Erez, Y. Tassa, N. de Freitas, and M. Denil, "Learning awareness models," *International Conference on Learning Representations*, 2018.

[44] 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)*, vol. 2, pp. 995–1001, IEEE, 2000.

[45] S. Chitta, I. Sucan, and S. Cousins, "Moveit![ros topics]," *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CVPR*, 2016.

[47] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel, "Deep spatial autoencoders for visuomotor learning," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 512–519, IEEE, 2016.