Sequential Dexterity: Chaining Dexterous Policies for Long-Horizon Manipulation

Anonymous Author(s)
Affiliation
Address
email

Abstract: Many real-world manipulation tasks consist of a series of subtasks that are significantly different from one another. Such long-horizon, complex tasks highlight the potential of dexterous hands, which possess adaptaility and versatility, capable of seamlessly transitioning between different modes of functionality without the need for re-grasping or external tools. However, the challenges arise due to the high-dimensional action space of dexterous hand and complex compositional dynamics of the long-horizon tasks. We present Sequential Dexterity, a general system based on reinforcement learning (RL) that chains multiple dexterous policies for achieving long-horizon task goals. The core of the system is a *transition feasibility function* that progressively finetunes the sub-policies for enhancing chaining success rate, while also enables autonomous policy-switching for recovery from failures and bypassing redundant stages. Despite being trained only in simulation with a few task objects, our system demonstrates excellent generalization capability to novel objects and is able to zero-shot transfer to a real-world robot equipped with a dexterous hand. More details and video results could be found at https://sites.google.com/view/seq-dex.

Keywords: Dexterous Manipulation, Reinforcement Learning, Long-Horizon Manipulation

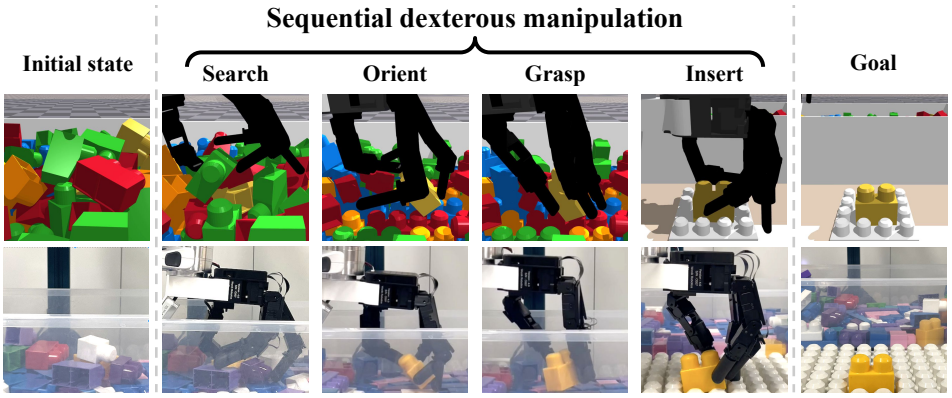


Figure 1: We present **Sequential Dexterity**, a system that learns to chain multiple versatile dexterous manipulation motions for tackling long-horizon tasks (e.g., building a block structure from a pile of blocks), which is able to zero-shot transfer to the real world.

1 Introduction

Many real-world manipulation tasks consist of a sequence of smaller but drastically different subtasks. For example, in the task of Lego structure building (Fig.1), the task involves *searching* within a box to locate a specific block piece. Once found, the piece is then *oriented* and *grasped* firmly in hand,

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

setting it up for the final *insertion* at the goal location. Such a task demands a flexible and versatile manipulator to adapt and switch between different modes of functionality seamlessly, avoiding re-grasping or use of external tools. Furthermore, it requires a long-horizon plan that considers the temporal context and functional relationship between the subtasks in order to successfully execute the entire sequence of tasks. These requirements motivate the use of dexterous hand, which has the potential to reach human-level dexterity by utilizing various hand configurations and their inherent capabilities. However, fully utilizing dexterous hands to achieve long-horizon, versatile tasks remains an outstanding challenge, calling for innovative solutions.

Recent developments in dexterous manipulation have made significant strides in areas such as object grasping [1–3] and in-hand manipulation [4–9]. However, these works primarily investigate single-stage skills, overlooking the potential of sequencing multiple dexterous policies for long-horizon tasks. A naive way to chain multiple dexterous policies together is to simply execute a single-stage skill one after the other. While the simple strategy works in some scenarios [10–12], a subtask in general can easily fail when encountering a starting state it has never seen during training. Regularizing the state space between neighboring skills can mitigate this out-of-distribution issue [13, 14], but long-horizon dexterous manipulation requires a comprehensive optimization of the entire skill chain, due to the complex coordination between non-adjacent tasks. For instance, as depicted in Fig. 1, the robot needs to strategize in advance when orienting the block, aiming for an optimal object pose that facilitates not only the immediate subsequent grasping but also the insertion task in the later stage of the task.

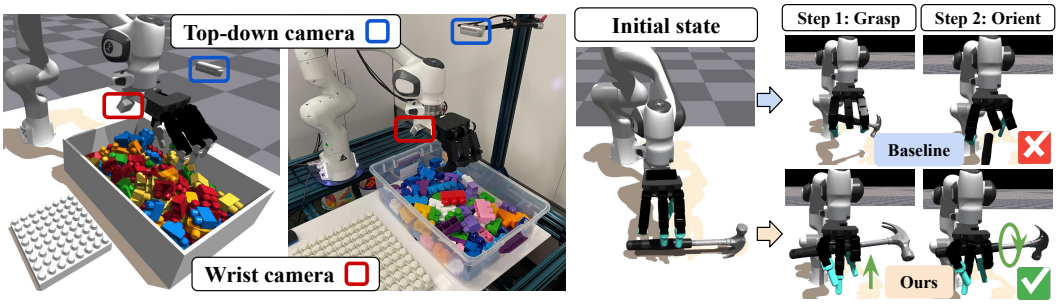
This paper proposes a new method to effectively chain multiple high-dimensional manipulation policies via a combination of regularization and optimization. We introduce a bi-directional process consisting of a forward initialization process and a backward fine-tuning process. The forward initialization process models the end-state distribution of each sub-policy, which defines the initial state distribution for the subsequent policy. The forward initialization process associates the preceding policy with the subsequent one by injecting a bias in the initial state distribution of the subsequent policy during training. Conversely, we also introduce a backward fine-tuning mechanism to associate the subsequent policy with the preceding one. We define a *Transition Feasibility Function* which learns to identify initial states from which the subsequent policy can succeed its task. The transition feasibility function is used to fine-tune the preceding policy, serving as an auxiliary reward signal. With the backward fine-tuning process, the transition feasibility function effectively backpropagates the long-term goals to influence the earlier policies via its learning objectives, thereby enabling global optimization across the entire skill chain. Once the policies are trained and deployed, the transition feasibility functions can be repurposed to serve as stage identifiers that determine the optimal timing for policy switching and the optimal subsequent policy to switch to. The transition feasibility function substantially improves the robustness of task execution and increases the success rate. Our experimental results demonstrate that the bi-directional optimization process notably enhances the performance of chaining multiple dexterous manipulation policies, which can further zero-shot transfer to a real-world robot arm equipped with a dexterous hand to tackle challenging long-horizon dexterous manipulation tasks.

In summary, the primary contributions of this work encompass:

- The first to explore long-horizon dexterous manipulation with policy chaining.
- A **bi-directional optimization framework** that significantly enhances the success rate, efficiency, and robustness of chaining multiple dexterous policies.
- Our framework exhibits state-of-the-art results in two multi-stage dexterous manipulation tasks and facilitates seamless zero-shot transfer to a real-world dexterous robot system.

2 Related Work

Dexterous manipulation. Dexterous manipulation represents a long-standing area of research in robotics [15–19]. With its high degree of freedom, a dexterous hand can execute a variety of manipulation skills [1–9, 20–22]. Traditional algorithms have typically addressed these challenges by leveraging trajectory optimization founded on analytical dynamics modeling [17–19]. These techniques pose simplification over the active contacts between the hand and objects, limiting their effectiveness in more



(a). Building Blocks (Sim and Real) **(b). Tool Positioning**

Figure 2: Overview of the environment setups. (a) Workspace of **Building Blocks** task in simulation and real-world. (b) The setup of the **Tool Positioning** task. Initially, the tool is placed on the table in a random pose, and the dexterous hand needs to grasp the tool and re-orient it to a ready-to-use pose. The comparison results illustrate how the way of grasping directly influences subsequent orientation.

complex tasks. Conversely, deep reinforcement learning have exhibited the capacity to learn dexterous skills without any assumptions [7, 8, 21]. Despite their notable flexibility in learning dexterous primitives, these methods predominantly focus on singular, isolated tasks. Our work prioritizes the chaining of multiple dexterous primitives into a comprehensive, long-horizon manipulation process.

Long-horizon robot manipulation. Training a robot to perform long-horizon manipulation tasks from scratch is challenging, primarily due to the cumulative propagation of errors throughout the task execution process. Established methods tackle these tasks by breaking them down into simpler, reusable subtasks [23]. Typically, these algorithms comprise a set of low-level sub-policies, which can be obtained through various means, such as unsupervised exploration [24–28], learning from demonstration [29–32] and pre-defined measures [11, 33–37]. Despite their distinct merits, these works do not address the specific challenge of long-horizon manipulation in the context of dexterous hands. This challenge largely stems from the compounded complexity produced by the extensive state space of a hand coupled with the extended scope of long-horizon tasks. Therefore, even when provided with high-level plans, ensuring a seamless transition between dexterous policies remains a formidable challenge.

Skill-chaining. Prior policy-chaining methods focus on updating each sub-task policy to encompass the terminal states of the preceding policy [10, 13]. However, given the high degree of freedom characteristic of a hand, the terminal state space undergoes limitless expansion, thereby complicating effective training. Closely related to our work is Lee et al. [14], wherein a discriminator is learned to regulate the expansion of the terminal state space. Nevertheless, its uni-directional training process restricts optimization to adjacent skills only, disregarding the influence of long-term goals on early non-adjacent policies. In contrast, our bi-directional training mechanism enables the backpropagation of the long-term goal reward to optimize the entire policy chain. Our concept of backward fine-tuning draws significant inspiration from goal-regression planning in classical symbolic planning literatures [38] (also known as pre-image backchaining [39–43]). However, these works assume access to a set of pre-defined motion primitives, which is hard to obtain in dexterous manipulation setups. Our work focuses on the learning and chaining of a sequence of dexterous policies from scratch, targeting the accomplishment of long-horizon task objectives.

3 Problem Setups

We study the task of chaining a sequence of dexterous policies to accomplish long-horizon manipulation tasks, examples of which include Lego-like building blocks or picking up and positioning a tool to a desired pose. These two tasks both require the use of multiple dexterous skills to complete, making them highly suitable for studying long-horizon dexterous manipulation with skill chaining.

Constructing a structure of blocks. This long-horizon task includes four different subtasks: **searching** for a block with desired dimension and color from a pile of cluttered blocks, **orienting** the block to a favorable position, **grasping** the block, and finally **inserting** the block to its designated position on the structure. This sequence of actions repeats until the structure is completed according to

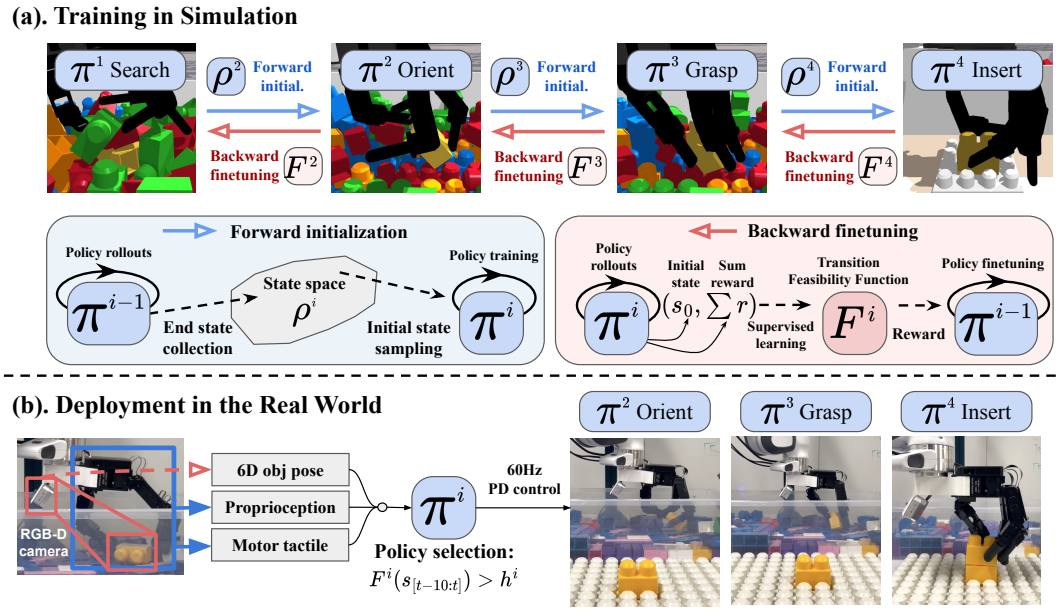


Figure 3: Overview of **Sequential Dexterity**. (a) A bi-directional optimization scheme consists of a forward initialization process and a backward fine-tuning mechanism based on the transition feasibility function. (b) The learned system is able to zero-shot transfer to the real world. The transition feasibility function serves as a policy-switching identifier to select the most appropriate policy to execute.

the given assembly instructions. The block set, initially arranged in a random configuration, comprises eight distinct types (different shapes, masses and colors), totaling 72 blocks. We operate under the assumption of having access to an assembly manual that outlines the sequence and desired positioning of each block piece on the board. The environment provides the robot with two RGB-D camera views—one from a top-down camera over the box and the other from the wrist-mounted camera (as is shown in Fig. 2(a)). No other sensors are used in either simulation or the real world.

Tool positioning. This task involves two subtasks: **grasping** a tool with a long handle (e.g., hammer, spatula) from a table and **in-hand orienting** it to a ready-to-use pose (e.g., make the flat side of the hammerhead face the nail, as is shown in Fig. 2(b)). The environment provides the robot with the 6D pose of the target tool. For more details on the task setups, please refer to Appendix. F.

4 Sequential Dexterity

We propose a bi-directional optimization process to tackle long-horizon dexterous manipulation tasks. Our approach contains three main components: (1) Training dexterous sub-policies (Sec. 4.1), (2) Chaining sub-policies through fine-tuning (Sec. 4.2), (3) Improving system robustness through automatic policy-switching (Sec. 4.3).

4.1 Learning dexterous sub-policies

Training a dexterous manipulation policy from scratch for solving long-horizon tasks, like building a block tower (Fig. 1), is significantly challenging given the high degree of freedom associated with a dexterous hand (evidenced by the result of RL-scratch in Tab. 1 and Tab. 2). As such, we first decompose a long-horizon task into a K-step sequence of sub-tasks $G=(g^1,g^2,\ldots,g^K)$ and train each sub-policy π^i with Proximal Policy Optimization (PPO) [44] algorithm. We formulate each sub-task as a Markov Decision Process (MDP) $\mathcal{M}=(\boldsymbol{S},\boldsymbol{A},\pi,\mathcal{T},R,\gamma,\rho)$, with state space $\boldsymbol{S}$, action space $\boldsymbol{A}$, policy of the agent π, transition distribution $\mathcal{T}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t,\boldsymbol{a}_t)$ ($\boldsymbol{s}_t\in\boldsymbol{S}$, $\boldsymbol{a}_t\in\boldsymbol{A}$), reward function R, discount factor $\gamma\in(0,1)$, and initial state distribution ρ. The policy π outputs a distribution of motor actions $\boldsymbol{a}_t$ based on the current state inputs $\boldsymbol{s}_t$. The goal is to find the optimal policy π to maximize the sum of rewards $\mathbb{E}_\pi[\sum_{t=0}^{T-1}\gamma^t r_t]$ ($r_t=R(\boldsymbol{s}_t,\boldsymbol{a}_t,\boldsymbol{s}_{t+1})$) in an episode with T time steps.

However, due to the large state space of a dexterous hand, it is difficult to accurately sample the potential initial states for training individual sub-policies. Take the insertion task as an example (Fig. 3 sub-policy π^4), randomly sampling the initial hand configuration and object's in-hand pose does not assure a physically stable grasp. However, we provide a critical observation that the successful end state of prior sub-task π^{i-1} inherently provides plausible initial states for π^i to start with. Inspired by this, we propose a forward initialization training scheme (Fig. 3 (a)). Given a long-horizon task $G=(g^1,g^2,...,g^K)$, our framework sequentially trains the sub-policies according to the task's chronological order. After training each sub-policy π^i, we start policy rollouts and collect a set of successful terminal states $\{\boldsymbol{s}_T^i\}$, which is later used as the initial state distribution ρ^{i+1} for training the succeeding policy π^{i+1}. Such forward training method ensures the validity of the initial states and makes the learning of dexterous policies effective. More details of training sub-policies can be found in Sec. 4.4.

4.2 Policy chaining with transition feasibility function

Chaining multiple policies using forward initialization alone may not guarantee success since the previous policy π^{i-1} might reach a termination state that its successor π^i cannot solve. This issue arises because the preceding policy π^{i-1} does not take into account whether the end states are *feasible* for the subsequent policy π^i to succeed. To address this challenge, it is crucial to convey the feasibility of the following policy π^i *in reverse* to its predecessor π^{i-1}, enabling the latter to optimize toward states that π^i can handle. Based on this hypothesis, we propose a backward policy fine-tuning mechanism with a transition feasibility function (Fig. 3 (b)).

Learning transition feasibility function. The feasibility of a given state for a policy can be described as the policy's ability to succeed in the end when starting from that state. We formalize this concept by creating a function that maps the transition state $\boldsymbol{s}_0^i \in \rho^i$ (which is equivalent to $\boldsymbol{s}_T^{i-1}$) to the expected sum of reward within the sub-task execution, $\mathbb{E}_{\pi^i}[\sum_{t=0}^{T-1} r_t]$. We name this function, $F:\mathcal{S}\mapsto\mathbb{R}$, the *Transition Feasibility Function*. However, a single state $\boldsymbol{s}_T^{i-1}$ is not sufficient to differentiate the performance of π^i. In particular, the velocity of finger motion from the previous sub-task may be critical to the performance of π^i, which cannot be captured by $\boldsymbol{s}_T^{i-1}$ alone. As a result, the transition feasibility function takes a sequence of observation states $\boldsymbol{s}_{[T-10:T]}^{i-1}$ (10 steps in our experiments) as input and employs a multi-head attention network [45] to extract suitable temporal information for learning the F. The final learning objective of the transition feasibility function F^i for sub-policy π^i is:

$$\mathcal{L}^i = \mathbb{E}_{\boldsymbol{s_t}\sim\rho^i}\left[F^i(\boldsymbol{s_{[T-10:T]}}) - \mathbb{E}_{\pi^i}[\sum_{t=0}^{T-1} r_t]\right] \tag{1}$$

Backward policy fine-tuning. Once F^i is trained, we can fine-tune the prior policy π^{i-1}, by incorporating F^i as an auxiliary reward component. The fine-tuning starts with updating the second-to-the-last sub-task policy π^{K-1} and sequentially moves backward, refining each preceding policy until the first one π^1 is updated. In each fine-tuning step, we utilize F^i as an additional reward, combined with the original sub-task reward R^{i-1}, to fine-tune policy π^{i-1}. The final policy fine-tuning reward function is:

$$R^{i-1\prime}(\boldsymbol{s}_t,\boldsymbol{a}_t,\boldsymbol{s}_{t+1};F^i) = \lambda_1 R^{i-1}(\boldsymbol{s}_t,\boldsymbol{a}_t,\boldsymbol{s}_{t+1}) + \lambda_2 F^i(\boldsymbol{s}_{[T-10:T]}), \tag{2}$$

where λ_1 and λ_2 are the weighting factors. Once π^{i-1} has been refined, we execute policy rollouts to gather data, which maps from the initial state $\boldsymbol{s}_{[T-10:T]}^{i-2}$ to the accumulated reward $\mathbb{E}_{\pi^{i-1}}[\sum_{t=0}^{T-1} r_t]$ received by the policy π^{i-1} at the terminal state s_T^{i-1}. This data helps to construct a new transition feasibility function F^{i-1}, which is further used to fine-tune the preceding policy π^{i-2}. The implementation pseudocode of the bi-directional forward and backward training process is illustrated in Appendix. A.

4.3 Policy switching with transition feasibility function

A key challenge in chaining multiple dexterous policies is to determine *when* to switch to the next policy and *what* should be the next policy to execute. Prior works approach this issue by establishing a predetermined execution horizon for a policy, transitioning abruptly to the subsequent policy once

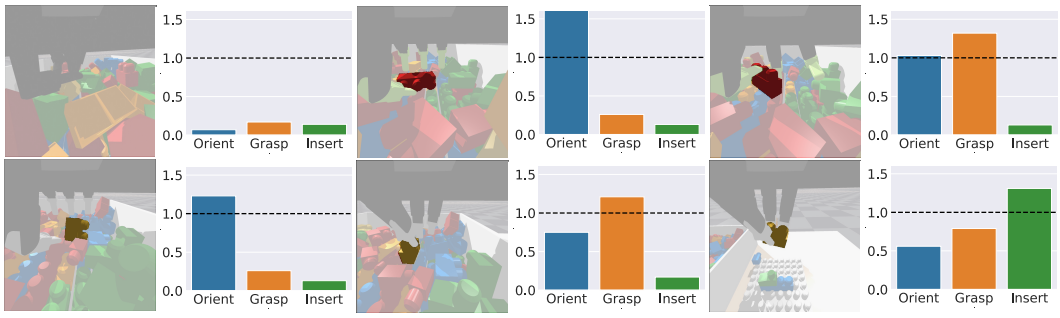


Figure 4: Examples of policy-switching with transition feasibility function. Each example contains an image from the wrist-mount camera (left) and its corresponding feasibility score c_i outputted by the transition feasibility function (right). We highlight the target block in the image for better visualization. The policy-switching process visits each sub-policy in reverse order. The first sub-policy with a feasibility score $c_i > 1.0$ is selected for execution.

the maximum step count is attained [14, 46, 10]. The pre-scheduled policy transitions worked in some scenarios, but they are not suitable for dexterous manipulation that involves non-prehensile maneuvers and in-hand manipulation. For instance, if a robot is reorienting an object in-hand, a premature policy switch before the object is stabilized could result in task failure. The key to tackling this issue is to automatically figure out the optimal switch time such that the transition state will lead to success of next policy. The transition feasibility function provides exactly the information we need for identifying the switch timing. As such, during execution, we repurpose our trained transition feasibility functions as a policy-switching identifier. At each time step, the transition feasibility function of the next sub-policy will output a feasibility score $c_t^{i+1} = F^{i+1}(\boldsymbol{s}_{[t-10:t]})/h^{i+1}$, where h^{i+1} is a threshold hyperparameter defined based on the reward of successful task executions. The ideal time of policy switching can then be defined as the moment when $c_t^{i+1} > 1$.

Simply executing sub-policies sequentially may not guarantee successful task execution since the robot sometimes needs to recover using a previous policy and sometimes needs to bypass a future policy if the sub-task has already been achieved. Thus, effective policy switching requires the robot to not only consider the current policy and its successor, but also the entire skill chain. To achieve this, we group the learned transition feasibility function $(F^2, F^3 ..., F^K)$ as a stage estimator. At each policy-switching step, we calculate the feasibility score from the final transition feasibility function of the entire task $c_t^K = F^K(\boldsymbol{s}_{[t-10:t]})/h^K$, sequentially moving backward. The sub-policy, for which the first feasibility score $c_t^i > 1$, is considered the next policy for execution. If none of the feasibility scores satisfy, the robot will restart from the beginning of the entire task. Leveraging the learned transition feasibility function in this manner enhances the robot's robustness against unexpected failures during policy execution, while also allowing it to bypass redundant stages, thus promoting efficient task execution.

4.4 Implementation details

RL reward. Training sub-policies require pre-defined sub-task rewards $\{R^i\}_{i=1}^K$. Establishing such rewards can be complex as the optimal sub-goals that would most contribute to the overall task accomplishment may not be readily apparent. However, we pose a critical finding that the backward fine-tuning mechanism can transmit the goal of the entire task to each sub-task. For instance, the transition feasibility function of the inserting policy informs the grasping policy about the in-hand object pose that would be most beneficial for the insertion. Furthermore, such backward transmission can influence all preceding sub-policies, enabling the entire policy chain to optimize for the overall task goal. This mechanism alleviates the burden of reward shaping and allows us to use standard sub-task rewards that are agnostic to the final task goal for training sub-policies. For instance, the sub-task reward of grasping is defined as whether the target object has been lifted. The specifics of how the object is held in hand are automatically managed during the backward fine-tuning process. The detailed descriptions of each sub-task reward are documented in the Appendix. D.

State-action space. The state space for the sub-policies is built around the perspective of the hand. It integrates proprioception and motor tactile [47, 48] information from the 16-degree-of-freedom

	Trained					Unseen			
	Block 1	Block 2	Block 3	Block 4	Block 5	Block 6	Block 7	Block 8	ALL
RL-scratch	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
Curriculum RL	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
V-Chain [28]	$0.15_{\pm 0.02}$	$0.09_{\pm 0.04}$	$0.11_{\pm 0.04}$	$0.10_{\pm 0.04}$	$0.08_{\pm 0.02}$	$0.08_{\pm 0.03}$	$0.03_{\pm 0.02}$	$0.04_{\pm 0.02}$	$0.08_{\pm 0.02}$
Policy-Seq [10]	$0.20_{\pm 0.04}$	$0.14_{\pm 0.02}$	$0.15_{\pm 0.03}$	$0.23_{\pm 0.03}$	$0.15_{\pm 0.03}$	$0.17_{\pm 0.00}$	$0.16_{\pm 0.01}$	$0.12_{\pm 0.02}$	$0.16_{\pm 0.02}$
T-STAR [14]	$0.19_{\pm 0.04}$	$0.18_{\pm 0.02}$	$0.11_{\pm 0.01}$	$0.27_{\pm 0.02}$	$0.17_{\pm 0.04}$	$0.25_{\pm 0.02}$	$\mathbf{0.26}_{\pm \mathbf{0.03}}$	$0.10_{\pm 0.03}$	$0.19_{\pm 0.03}$
Ours w/o temporal	$0.47_{\pm 0.06}$	$0.44_{\pm 0.07}$	$0.43_{\pm 0.00}$	$0.49_{\pm 0.04}$	$0.40_{\pm 0.04}$	$0.51_{\pm 0.04}$	$0.18_{\pm 0.01}$	$\mathbf{0.16}_{\pm \mathbf{0.03}}$	$0.38_{\pm 0.04}$
Ours	$\mathbf{0.61}_{\pm \mathbf{0.03}}$	$\mathbf{0.55}_{\pm \mathbf{0.01}}$	$\mathbf{0.52}_{\pm \mathbf{0.03}}$	$\mathbf{0.63}_{\pm \mathbf{0.03}}$	$\mathbf{0.51}_{\pm \mathbf{0.06}}$	$\mathbf{0.53}_{\pm \mathbf{0.06}}$	$0.22_{\pm 0.02}$	$\mathbf{0.16}_{\pm \mathbf{0.01}}$	$\mathbf{0.46}_{\pm \mathbf{0.03}}$

Table 1: Results for the Building Blocks task

Allegro Hand as well as the target object's 6D pose in the reference frame of the wrist-mounted camera. During simulation-based sub-policy training, we augment this state space with additional information, such as the velocity of each hand joint and the target object. In real-world deployments, these states are abstracted via policy distillation [6, 7, 49]. The action space of our system includes 16-dimensional hand joints and 3D wrist translation of the robot arm. To concentrate the sub-policy learning on critical aspects of the manipulation task, when the target's (either object or goal location) relative pose to the wrist camera is detected more than 5 centimeters from the hand, we employ a motion-planning-based operational space controller (OSC [50]) to move the end-effector to a position 5 centimeters above the target. Such a target-centric action space assists in aligning the manipulator appropriately before initiating the manipulation task. More details of the state-action space are introduced in Appendix. C.

5 Experiments

5.1 Experiment setups

Environment setups. The environment is initialized with a Franka Emika robot arm equipped with an Allegro Hand as the end-effector. In the Building Block task, we placed a box of blocks (eight categories, in total 72 pieces) and a building board on the table. For the tool positioning task, a long-handled tool is placed on the table. The control frequency is 60 Hz for both the robot arm and the hand. The real-world hardware mirrors the simulation setups. We coordinate the use of top-down and wrist-mount cameras to access the 6D pose and segmentation mask of the object in the real-world: we use the top-down camera once at the beginning, use the wrist-mount camera once at the beginning of each maneuver in **orienting**, and use the wrist-mount camera continuously in **searching**, **grasping** and **inserting**; More details of the environment setups are in Appendix. F.

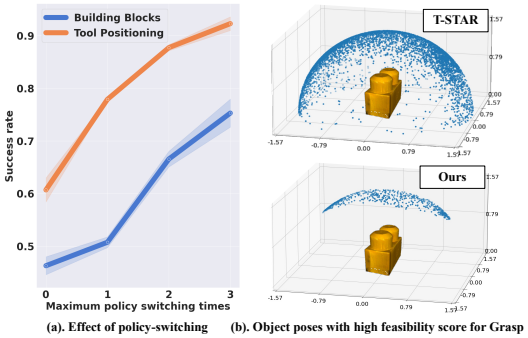


Figure 5: (a) Performance improvement of Ours given 0/1/2/3 maximum policy-switching times. (b) Visualization of object poses with high feasibility score for the Grasp sub-policy in Building Blocks task. The x, y, and z axes are the roll, yaw, and pitch of the object, respectively. In Ours, each point in the diagram represents a pose that is regarded as feasible by the transition feasibility function ($c^i > 1.0$). For T-STAR, we use the poses that are judged as successful by its discriminator.

Baseline methods. We compare our approach with the following baselines: 1) RL-scratch is vanilla PPO algorithm [44] learns the task from scratch. 2) Curriculum RL follows a procedure training strategy to expand from the first skill to the entire task. 3) V-Chain [28] combines skill-chaining with the value function from the PPO policy. 4) Policy-Seq [10] focuses on the forward initiation process in skill-chaining. 5) T-STAR [14] incorporates a discriminator to regularize the terminal states.

5.2 Results

Bi-directional optimization framework is key for chaining multiple dexterous policies. In Tab. 1 and Tab. 2, our approach learned with bi-directional optimization (Ours and Ours w/o temporal) outperforms prior uni-directional skill-chaining methods (V-Chain, Policy-Seq and T-STAR) significantly, with more than 20% improvement in task success rate in two long-horizon tasks. We further analyze what really matters for successful policy chaining. We visualize the transition feasibility score of the

	Trained	Unseen		
	Hammer	Spatula	Spoon	ALL
RL-scratch	$0.17_{\pm 0.05}$	$0.06_{\pm 0.03}$	$0.10_{\pm 0.01}$	$0.11_{\pm 0.03}$
Curriculum RL	$0.29_{\pm 0.02}$	$0.17_{\pm 0.01}$	$0.16_{\pm 0.08}$	$0.21_{\pm 0.04}$
Policy-Seq [10]	$0.43_{\pm 0.01}$	$0.29_{\pm 0.06}$	$0.24_{\pm 0.04}$	$0.32_{\pm 0.02}$
T-STAR [14]	$0.47_{\pm 0.01}$	$0.40_{\pm 0.03}$	$0.26_{\pm 0.04}$	$0.37_{\pm 0.03}$
Ours w/o temp.	$0.77_{\pm 0.03}$	$0.54_{\pm 0.07}$	$0.40_{\pm 0.04}$	$0.57_{\pm 0.05}$
Ours	$\mathbf{0.81}_{\pm \mathbf{0.01}}$	$\mathbf{0.57}_{\pm \mathbf{0.04}}$	$\mathbf{0.43}_{\pm \mathbf{0.08}}$	$\mathbf{0.60}_{\pm \mathbf{0.04}}$

Table 2: Results for the tool positioning task

	Single Block 1	Single Block 4	Double Block 1
RL-scratch	0/10	0/10	0/10
Policy-Seq [28]	0/10	2/10	0/10
T-STAR [14]	3/15	5/18	0/13
Ours	**12/20**	**10/20**	**5/15**

Table 3: Real world results in the Building Blocks task. Single/Double refers to building one single block or stacking two blocks.

grasping sub-policy (T-STAR's result is calculated from its discriminator) in Fig. 5(b). We found our approach with backward fine-tuning scheme correctly transits the goal of the succeeding *inserting* skill to prior *grasping* and encourages the policy to grasp the block when its studs face up, which facilitates the insertion. T-STAR with the uni-directional learning process, however, suggests many states where the block's studs are facing horizontally, thereby bringing challenges for subsequent insertion.

Transition feasibility function significantly improves performance of long-horizon dexterous manipulation. In Tab. 1 and Tab. 2, the models learned with the transition feasibility function (Ours and Ours w/o temporal) outperforms the one using the PPO-trained value function (V-chain) for more than 30% in task success rate. This result implies that the value function of PPO policy fails to model the feasibility of subsequent policy, which further affects policy chaining results.

Temporal inputs facilitate handling high-dimensional state spaces, particularly for dexterous manipulation. In Tab. 1, by training the transition feasibility function to extract temporal information from a sequence of history states, Ours exceeds Ours w/o temporal for 8% in task success rate. This result highlights the importance of extracting velocity and temporal information for chaining dexterous policies that contain dynamic finger motions.

Ability to switch sub-policy optimally and autonomously is essential for succeeding long-horizon tasks. Fig. 5(a) illustrates the performance improvement of enabling automatic policy-switching. Only with a maximum allowance of three switching times, Ours can improve more than 30% in task success rate. This result concludes that it is crucial to have the capability of switching forward and backward by leveraging the transition feasibility function of each sub-policy. Such policy-switching ability further contributes to our real-world results in Tab. 3, which allows the policy to handle a challenging 8-step long-horizon task (Double Block 1) with more than 30% task success rate (10 maximum switching times for all methods in the real-world experiments). Please refer to our Supplementary video for more results.

Real-world results. In Tab. 3 real-world experiments, our approach has more than 30% success rate improvements compared to prior methods. This result is consistent with the results in simulation. Ours has a 33% success rate in building a double-block structure, while the other baselines have a 0% success rate. This result highlights the robustness of our model when tackling long-horizon dexterous manipulation tasks. For more details of the real-world setups, please refer to Appendix. B.

6 Conclusion

We present Sequential Dexterity, a system developed for tackling long-horizon dexterous manipulation tasks. Our system leverages a bi-directional optimization process to chain multiple dexterous policies learned with deep reinforcement learning. At the core of our system is the Transition Feasibility Function, a pivotal component facilitating a gradual fine-tuning of sub-policies and enabling dynamic policy switching, thereby significantly increasing the success rate of policy chaining. Our system has the capability to zero-shot transfer to a real-world dexterous robot, exhibiting generalization across novel object shapes.

Our study has several limitations. First, we encounter difficulties in simulating a contact-rich insertion process which necessitates an additional manually designed pressing motion to completely insert the blocks during real-world deployment. Second, our design choice of motor tactile does not yield a significant improvement in performance, as observed in Appendix Tab. 8. Future research could explore the potential of sensor-based tactile signals for managing contact-rich tasks, as proposed in [8, 22].

References

[1] Z. Q. Chen, K. Van Wyk, Y.-W. Chao, W. Yang, A. Mousavian, A. Gupta, and D. Fox. Learning robust real-world dexterous grasping policies via implicit shape augmentation. *arXiv preprint arXiv:2210.13638*, 2022.

[2] A. Wu, M. Guo, and C. K. Liu. Learning diverse and physically feasible dexterous grasps with generative model and bilevel optimization. *arXiv preprint arXiv:2207.00195*, 2022.

[3] Y. Qin, B. Huang, Z.-H. Yin, H. Su, and X. Wang. Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation. In *Conference on Robot Learning*, pages 594–605. PMLR, 2023.

[4] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[5] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. *arXiv preprint arXiv:2210.13702*, 2022.

[6] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal. Visual dexterity: In-hand dexterous manipulation from depth. *arXiv preprint arXiv:2211.11744*, 2022.

[7] T. Chen, J. Xu, and P. Agrawal. A system for general in-hand object re-orientation. *Conference on Robot Learning*, 2021.

[8] Z.-H. Yin, B. Huang, Y. Qin, Q. Chen, and X. Wang. Rotating without seeing: Towards in-hand dexterity through touch. *arXiv preprint arXiv:2303.10880*, 2023.

[9] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. *arXiv preprint arXiv:2203.13251*, 2022.

[10] A. Clegg, W. Yu, J. Tan, C. K. Liu, and G. Turk. Learning to dress: Synthesizing human dressing motion via deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 37(6):1–10, 2018.

[11] Y. Lee, S.-H. Sun, S. Somasundaram, E. S. Hu, and J. J. Lim. Composing complex skills by learning transition policies. In *International Conference on Learning Representations*, 2019.

[12] X. B. Peng, M. Chang, G. Zhang, P. Abbeel, and S. Levine. Mcp: Learning composable hierarchical control with multiplicative compositional policies. *Advances in Neural Information Processing Systems*, 32, 2019.

[13] G. Konidaris and A. Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. *Advances in neural information processing systems*, 22, 2009.

[14] Y. Lee, J. J. Lim, A. Anandkumar, and Y. Zhu. Adversarial skill chaining for long-horizon robot manipulation via terminal state regularization. *arXiv preprint arXiv:2111.07999*, 2021.

[15] J. K. Salisbury and J. J. Craig. Articulated hands: Force control and kinematic issues. *The International journal of Robotics research*, 1(1):4–17, 1982.

[16] M. T. Mason and J. K. Salisbury Jr. Robot hands and the mechanics of manipulation. 1985.

[17] I. Mordatch, Z. Popović, and E. Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pages 137–144, 2012.

[18] Y. Bai and C. K. Liu. Dexterous manipulation using both palm and fingers. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1560–1565. IEEE, 2014.

[19] V. Kumar, Y. Tassa, T. Erez, and E. Todorov. Real-time behaviour synthesis for dynamic hand-manipulation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6808–6815. IEEE, 2014.

[20] K. Zakka, L. Smith, N. Gileadi, T. Howell, X. B. Peng, S. Singh, Y. Tassa, P. Florence, A. Zeng, and P. Abbeel. Robopianist: A benchmark for high-dimensional robot control. *arXiv preprint arXiv:2304.04150*, 2023.

[21] Y. Chen, T. Wu, S. Wang, X. Feng, J. Jiang, Z. Lu, S. McAleer, H. Dong, S.-C. Zhu, and Y. Yang. Towards human-level bimanual dexterous manipulation with reinforcement learning. *Advances in Neural Information Processing Systems*, 35:5150–5163, 2022.

[22] I. Guzey, B. Evans, S. Chintala, and L. Pinto. Dexterity from touch: Self-supervised pre-training of tactile representations with robotic play. *arXiv preprint arXiv:2303.12076*, 2023.

[23] R. S. Sutton, D. Precup, and S. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

[24] J. Schmidhuber. *Towards compositional learning with dynamic neural networks*. Inst. für Informatik, 1990.

[25] P.-L. Bacon, J. Harb, and D. Precup. The option-critic architecture. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[26] O. Nachum, S. S. Gu, H. Lee, and S. Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

[27] A. Levy, G. Konidaris, R. Platt, and K. Saenko. Learning multi-level hierarchies with hindsight. *arXiv preprint arXiv:1712.00948*, 2017.

[28] V. C. Kumar, S. Ha, and C. K. Liu. Expanding motor skills using relay networks. In *Conference on Robot Learning*, pages 744–756. PMLR, 2018.

[29] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto. Robot learning from demonstration by constructing skill trees. *The International Journal of Robotics Research*, 31(3):360–375, 2012.

[30] T. Kipf, Y. Li, H. Dai, V. Zambaldi, A. Sanchez-Gonzalez, E. Grefenstette, P. Kohli, and P. Battaglia. Compile: Compositional imitation learning and execution. In *International Conference on Machine Learning*, pages 3418–3428. PMLR, 2019.

[31] Y. Lu, Y. Shen, S. Zhou, A. Courville, J. B. Tenenbaum, and C. Gan. Learning task decomposition with ordered memory policy network. *arXiv preprint arXiv:2103.10972*, 2021.

[32] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[33] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

[34] J. Oh, S. Singh, H. Lee, and P. Kohli. Zero-shot task generalization with multi-task deep reinforcement learning. In *International Conference on Machine Learning*, pages 2661–2670. PMLR, 2017.

[35] J. Merel, A. Ahuja, V. Pham, S. Tunyasuvunakool, S. Liu, D. Tirumala, N. Heess, and G. Wayne. Hierarchical visuomotor control of humanoids. *arXiv preprint arXiv:1811.09656*, 2018.

[36] J. Wu, R. Antonova, A. Kan, M. Lepert, A. Zeng, S. Song, J. Bohg, S. Rusinkiewicz, and T. Funkhouser. Tidybot: Personalized robot assistance with large language models. *arXiv preprint arXiv:2305.05658*, 2023.

[37] S. Nasiriany, H. Liu, and Y. Zhu. Augmenting reinforcement learning with behavior primitives for diverse manipulation tasks. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7477–7484. IEEE, 2022.

[38] R. Waldinger. Achieving several goals simultaneously. In *Readings in artificial intelligence*, pages 250–271. Elsevier, 1981.

[39] T. Lozano-Perez, M. T. Mason, and R. H. Taylor. Automatic synthesis of fine-motion strategies for robots. *The International Journal of Robotics Research*, 3(1):3–24, 1984.

[40] L. P. Kaelbling and T. Lozano-Pérez. Hierarchical task and motion planning in the now. In *2011 IEEE International Conference on Robotics and Automation*, pages 1470–1477. IEEE, 2011.

[41] L. P. Kaelbling and T. Lozano-Pérez. Pre-image backchaining in belief space for mobile manipulation. In *Robotics Research: The 15th International Symposium ISRR*, pages 383–400. Springer, 2017.

[42] 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.

[43] C. Agia, T. Migimatsu, J. Wu, and J. Bohg. Stap: Sequencing task-agnostic policies. *arXiv preprint arXiv:2210.12250*, 2022.

[44] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[46] E. Rosete-Beas, O. Mees, G. Kalweit, J. Boedecker, and W. Burgard. Latent plans for task-agnostic offline reinforcement learning. In *Conference on Robot Learning*, pages 1838–1849. PMLR, 2023.

[47] L. Sievers, J. Pitz, and B. Bäuml. Learning purely tactile in-hand manipulation with a torque-controlled hand. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2745–2751. IEEE, 2022.

[48] J. Pitz, L. Röstel, L. Sievers, and B. Bäuml. Dextrous tactile in-hand manipulation using a modular reinforcement learning architecture. *arXiv preprint arXiv:2303.04705*, 2023.

[49] A. A. Rusu, S. G. Colmenarejo, Ç. Gülçehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell. Policy distillation. In *ICLR (Poster)*, 2016.

[50] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987. doi:10.1109/JRA.1987.1087068.

[51] H. K. Cheng and A. G. Schwing. XMem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV*, 2022.

[52] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3343–3352, 2019.