Generative Factor Chaining: Coordinated Manipulation with Diffusion-based Factor Graph

Anonymous Authors.

***Abstract*—In the realm of challenging long-horizon planning tasks involving multiple manipulators, existing methods encounter computational scalability issues or require an impractical amount of training data. To address these limitations, we present Generative Factor Chaining (GFC), a novel approach based on modularized generative models for learning and composing skills in complex tasks. Our proposed method treats a long-horizon planning task in a complex scene as a spatial-temporal factor graph, where nodes represent objects in the scene and factors denote constraints/skills that connect different objects. By employing the diffusion model framework, different factors can be jointly learned using individual skill data, which is readily obtainable. During inference, these factors can be flexibly composed, possibly with additional constraints, to achieve long-horizon planning. The modular design of GFC enables generalization to unseen planning tasks. We showcase the advantages of our method through real-world experiments. More details can be found at: https://sites.google.com/view/generative-factor-chaining**

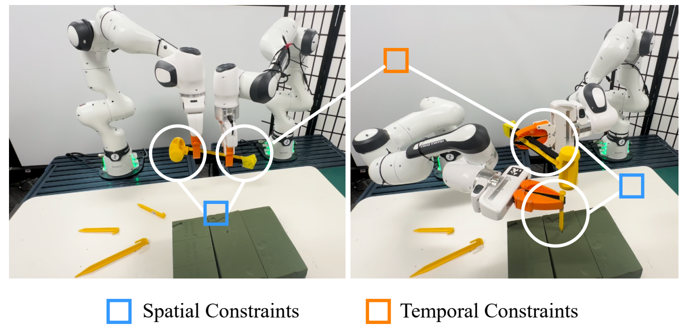

Fig. 1. **Motivational example.** Selected keyframes of a coordinated manipulation task. The goal is to pass the hammer from the left arm to the right arm and hammer the stake. A simplified spatial-temporal factor graph representation is illustrated, where squares are factors representing constraints, and circles are nodes representing object and robot states. Our method models both temporal (sequential) constraints between manipulation steps and spatial constraints for coordinated actions between multiple manipulators.

I. Introduction

Solving real-world sequential manipulation tasks requires reasoning about dependencies among manipulation steps. For example, a robot needs to grip the center or the tail of a hammer, instead of its head, in order to subsequently hammer a stake. The complexity of planning problems increases when multiple manipulators are involved, where spatial coordination constraints, in addition to sequential dependencies among manipulation steps of each arm, need to be satisfied. An example of such a scenario is shown in Figure 1, where the arm on the left has to reason about the effect of picking up the hammer with a certain pose, such that the arm on the right can coordinate to re-grasp. Subsequently, the two arms must coordinate to hammer the stake. While classical Task and Motion Planning (TAMP) methods have shown to be effective at solving sequential manipulation problems by hierarchical decomposition [15], they require comprehensive knowledge of the system state and kinodynamic model. Further, searching in such a large solution space to satisfy numerous constraints poses a severe scalability challenge. In this work, we aim to develop a learning-based planning framework to tackle complex manipulation tasks with both sequential and spatial coordination constraints.

To solve complex sequential manipulation problems, prior learning-to-plan methods have largely adopted the options framework [4] and either implicitly [12, 50] or explicitly [1, 32, 28] model the preconditions and effect of the options or primitive skills. Key to their successes are *skill chaining* functions that can determine whether executing a skill can satisfy the precondition of the next skill in the plan, and eventually the success condition of the overall task. For example, Deep Affordance Foresight [50] uses a learned skill feasibility predictor to guide the plan sampler. Generative Skill Chaining [32] constructs generative models for each skill. It then generates a plan by conditionally sampling skill sequences where each intermediate state satisfies the effect of the previous skill and the precondition of the next.

Unfortunately, despite the benefit, the use of vectorized states and the assumption of a *linear chain* of sequential dependencies severely limits the expressiveness of these methods. Consider a task where a robot is tasked to fetch two items from a box. Intuitively, the skills for fetching one object should have no influence on that of the other object. However, they will be represented together due to vectorized states and because of the linear dependency assumption, the skill-chaining methods are forced to model such sequential dependencies. In fact, we will empirically demonstrate that merely swapping the orders of skills that have no sequential dependencies in a plan can drastically affect the performance of these skill-chaining methods. Similarly, a skill that is intended to satisfy the condition of a subsequent skill a few steps later will be forced to influence the steps in between. Finally, the skill chain representation forbids these methods from effectively modeling multiple-arm manipulation tasks, where two or more concurrent skills must be planned to jointly satisfy a constraint. In this paper, we aim to relax the sequential dependency and construct a factorized state representation that can flexibly handle concurrent skill executions and facilitate generalization.

To this end, we introduce Generative Factor Chaining (GFC), a learning-to-plan method that relaxes the linear de-

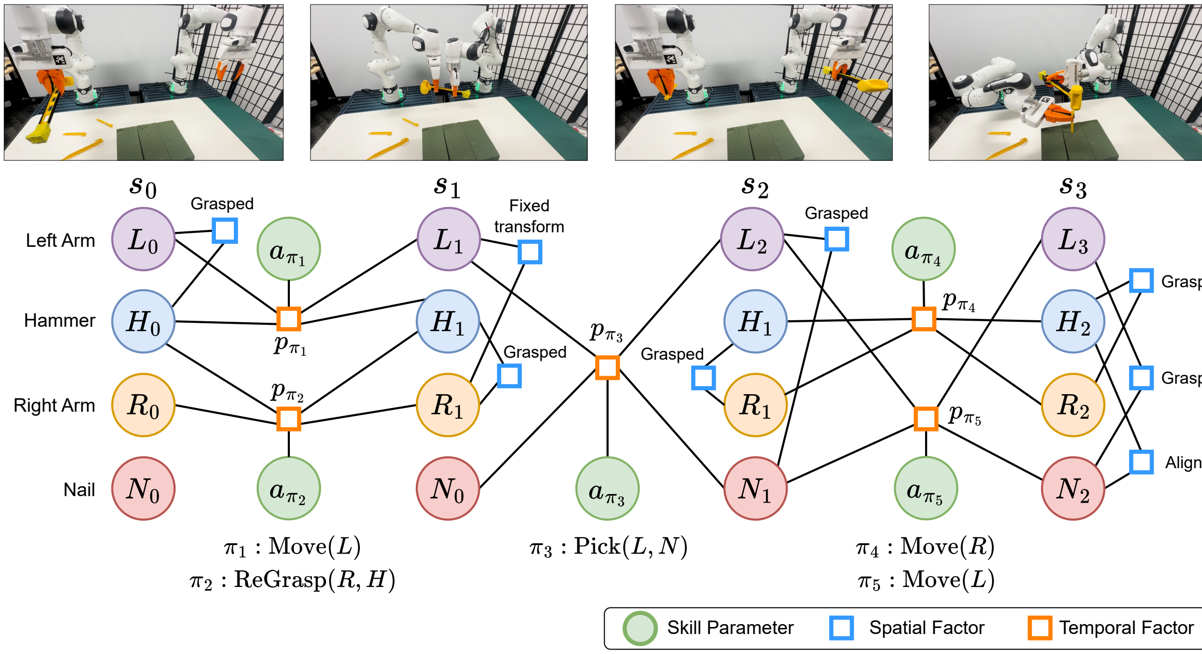

Fig. 2. **Factor graph for a multi-arm coordination task.** Our factor graph-based planning formulation is to solve for a sequence of spatial factor graphs from the initial state to a goal factor by chaining them using temporal skill factors. The above figure is an illustration of the temporal evolution of a factor graph by the execution of single or multiple skills sequentially or in-parallel. Given the hammer grasped by the left arm and a nail out of reach of the right arm, the goal is to handover the hammer to right arm followed by left arm picking up the nail. Finally, both arms coordinate to move to a position such that hammer can strike the nail. The subscript in the nodes denotes the temporal evolution of each of them.

pendency assumption and enables flexible plan composition by adopting a factor graph [10] representation. We represent states as spatial factor graphs and chain them using temporal skill factors to construct a spatial-temporal factor graph plan to solve for the geometrical solution and satisfy a goal condition. Further, individual spatial factors are designed as constraints, and skill factors are represented by distributions learned with diffusion models. While the distributions are trained only for the skill-level transition, they are chained using a probabilistic graphical model of the spatial-temporal factor graph plan to sample from a plan-level distribution with spatial constraints directly at inference. We empirically evaluate the factorized state representation and its applicability in handling sequential independence and improving long-horizon reasoning. In summary, our key contributions are as follows:

- We develop GFC, a learning-to-plan method that can flexibly model complex dependencies in long-horizon manipulation beyond linear sequential dependencies.
- GFC can perform zero-shot generalization to new coordination tasks with multiple manipulators by composing parallel skill chains of each arm directly at inference.
- GFC can reuse skill factor distributions to satisfy novel task constraints without any additional learning.
- On nine long-horizon single-manipulator tasks and four coordination tasks in simulation, we highlight the effectiveness of the flexible factor graph representation and CFG's spatial-temporal probabilistic chaining capability.

II. Related Work

Geometric Task and Motion Planning (TAMP). Generating motion to satisfy task and environment constraints is a fundamental challenge in long-horizon planning problems. TAMP frameworks decompose a complex planning problem into constraint satisfaction problems at task and motion levels [44, 4, 40, 33, 30] and characterize their solutions with symbolic task plan and primitive skills or local motion planning problems [35, 47, 8]. Notably, Garret et al. [15] drew connections between task-and-motion plans and dynamic factor graphs [10], where constraints are factors and robot and environment configurations are nodes. This formulation introduces considerable flexibility in reusing and composing constraint solvers across tasks. However, classical TAMP approaches rely on formulating fully-observable conditions and accurate system dynamics to generate feasible motion [21, 28, 41, 25, 49, 36, 48]. While such methods are exhaustive, their strong assumptions limit their practical applications and scalability. To this end, we opt for a learning-based solution [50, 1], while our factor graph representation and the goal of achieving compositional generalization remain heavily inspired by classical TAMP frameworks.

Learning to solve sequential manipulation tasks. There is a rich literature on learning to solve manipulation tasks with sequential dependencies. Most relevant to this work are skill-chaining methods that model pre- and post-conditions of predefined skills to search for feasible goal-reaching plans [27, 26, 1, 24, 5, 23, 9]. Recent skill-chaining methods have shown multi-task learning and generalization to new tasks [12, 50, 1, 32]. However, considering strict sequential dependencies might be limiting when independent tasks occur in a sequence. In contrast, our method adopts a factor graph representation that can flexibly model independence, skip-step dependencies, and multi-arm coordination in long-horizon tasks.

Generative models for planning. Recent advances in generative models have been adopted for imitation learning [52, 22, 37, 7, 31, 14, 39, 38], and offline reinforcement learning [20, 2] settings. In addition to the capacity to model complex state and action distributions, generative models have also been shown to encourage compositional generalization [55, 32, 13]. When trained on large multi-task datasets, Diffuser [20] and Decision Diffuser [2] have shown "trajectory stitching", the ability to combine data of different tasks. Most relevant to us are Generative Skill Chaining (GSC) [32] and Diffusion-CCSP [51], both designed to achieve systematic compositional generalization. GSC introduced a compositional diffusion model that can compose skill chains through a guided diffusion process. However, similar to other skill-chaining methods discussed above, GSC cannot discern between independent skills and considers irrelevant sequential dependencies. Diffusion-CCSP trains diffusion models to generate object configurations to satisfy multiple spatial constraints, while resorting to external solvers to plan the sequential manipulation steps that lead to the configuration. Our method can be viewed as solving the combined problem: it can generate plans to satisfy both spatial and temporal constraints represented in a factor graph.

Learning for coordinated manipulation. Coordinating two or more arms for manipulation presents numerous planning challenges [6, 34, 46], including searching in a combinatorial state-action space and solving complex constraint satisfaction problems for coordinated motion. Recent works have utilized learning-based frameworks [3, 8, 17, 16, 45]. The key challenges involve dealing with the increased exploration space in a Reinforcement Learning setting [3, 17] or mitigating compounding errors in an offline Imitation Learning setting [45, 16]. However, most existing works have focused on learning task-specific policies for single or a handful of coordinated manipulation tasks [3, 16] or require multi-arm demonstration data collected through a specialized teleoperation device [45], posing a significant scalability challenge. In contrast, our factor graph-based representation enables compositional generalization by design. We empirically demonstrate that GFC can solve new coordinated manipulation tasks through inference-time-guided diffusion.

III. Background

Diffusion Models. A core component of our method is based on distributions learned using diffusion models. A diffusion model learns an unknown distribution $p(\mathbf{x}^{(0)})$ from its samples by approximating the score function $\nabla \log p$. It consists of two processes: a *forward diffusion or noising* process that progressively injects noise and a *reverse diffusion or denoising* process that iteratively removes noise to recover clean data. The forward process simply adds Gaussian noise ϵ to clean data as $\mathbf{x}^{(t)} = \mathbf{x}^{(0)} + \sigma_t \epsilon$ for a monotonically increasing σ_t. The reverse process relies on the score function $\nabla_{\mathbf{x}} \log p_t(\mathbf{x}^{(t)})$ where p_t is the distribution of noised data $\mathbf{x}^{(t)}$. In practice, the unknown score function is estimated using a neural network $\epsilon_\phi(\mathbf{x}^{(t)}, t)$ by minimizing the denoising score matching [43] objective

$$\mathbb{E}_{t,\epsilon,\mathbf{x}^{(0)}}[\lambda(t)\|\epsilon - \epsilon_\phi(\mathbf{x}^{(t)}, t)\|^2] \quad (1)$$

where $\lambda(t)$ is a time-dependent weight. Several recent works have explored the advantages of diffusion models like scalability [19, 42, 54, 53] and the ability to learn multi-model distributions [18, 11, 29, 2]. We are particularly interested in the compositional ability [55, 52, 13, 51, 32] of these models for the proposed method.

Problem setup. We assume access to a library of parameterized skills [21] $\pi \sim \Pi$ such as primitive actions like `Pick` and `Place`. Each skill π requires a pre-condition to be fulfilled and is parameterized by a continuous parameter $a \in A_\pi$ governing the desired motion while executing the skill in a state s. For a given symbolically feasible task plan from a starting state s_0 to reach a specified goal condition s_{goal}, generated by a task planner or given by an oracle, the problem is to obtain the sequence of continuous parameters to make the plan geometrically feasible. For example, given a nail at a target location and a hammer on a table, the symbolic plan is to `Pick` the hammer and `Reach` the nail. A geometrically-feasible plan requires suitable `Pick` and `Reach` parameters such that the hammer's head can strike the nail.

Learning for skill chaining. Existing works on this problem model the planning problem as a "chaining" problem: They first model the pre-conditions and effect state distributions for every skill $\pi \sim \Pi$ from the available data and a symbolic *plan skeleton* $\Phi_K = \{\pi_1, \pi_2, ..., \pi_K\}$ consisting of K-skills is constructed. With this model, they search for the given skill sequence (plan) such that each skill satisfies the pre-conditions of the next skill in the plan. STAP used learned policy priors, system dynamics, and value functions to perform data-driven optimization with shooting and the cross-entropy maximization method. In GSC, the policy and transition model is formulated as a diffusion model based distribution $p_\pi(s, a_\pi, s')$ which allows for flexible forward and backward chaining. While the forward chain ensures dynamics consistency in the plan by a forward rollout of a trajectory $\tau = \{s_0, a_{\pi_1}, s_1, a_{\pi_2}, s_{goal}\}$ associated with $\Phi_2 = \{\pi_1, \pi_2\}$ using

$$p_\tau(\tau|s_0) \propto p_{\pi_1}(s_0, a_{\pi_1}, s_1)\; p_{\pi_2}(a_{\pi_2}, s_{goal}|s_1),$$

the backward chain ensures that the goal is reachable from the intermediate states via an alternate representation

$$p_\tau(\tau|s_{goal}) \propto p_{\pi_2}(s_1, a_{\pi_2}, s_{goal})\; p_{\pi_1}(s_0, a_{\pi_1}|s_1).$$

Following the above setup, the resulting forward-backward combination can be simply represented as

$$p_\tau(\tau|s_0, s_{goal}) \propto \frac{p_{\pi_1}(s_0, a_{\pi_1}, s_1) p_{\pi_2}(s_1, a_{\pi_2}, s_{goal})}{\sqrt{p_{\pi_1}(s_1) p_{\pi_2}(s_1)}} \quad (2)$$

The use of the diffusion model to represent individual transitions further allows constraint-guided sampling to handle unseen constraints at inference.

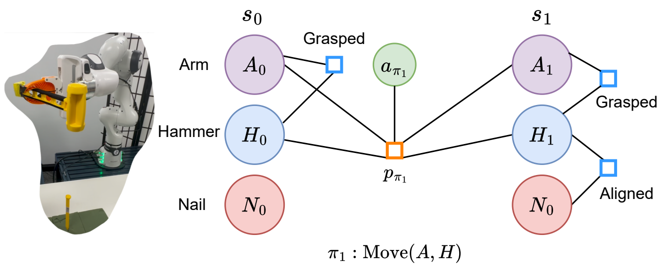

Fig. 3. **Scene as spatial factor graph.** We show how a state of the environment consisting of a gripper arm, hammer, and nail is formulated as a spatial factor graph with existing `Grasped` factor between the arm and hammer nodes. **Skill as temporal factor** The figure shows a spatial-temporal factor graph of executing a `Move` skill (move A such that A aligns with B). We illustrate the spatial-temporal sequence of the spatial factor graphs connected with temporal skill factors. The execution of the skill π_1 modifies the existing factor(s) between the nodes of interest (A, H) and adds new factor(s).

IV. Method

We aim to solve unseen long-horizon planning problems by exploiting the inter-dependencies between the objects important for the task at hand in the scene. Our method adopts factor graphs to represent states and realize their temporal evolution by the application of skills. While previous works have considered *vectorized state* representations making it difficult to decouple spatial-independence, we focus on *factorized state* representations such that the state of the environment is entirely modular, containing information about all the objects in the scenario and the task-specific constraints between them. We use a spatial-temporal factor graph to represent sequential and coordinated manipulation plan. This representation factorizes states by individual objects' and robot's states and allows us to combine skills from multiple arms on single/multiple objects simultaneously to achieve unseen collaborative tasks directly at inference. Further, we transform our spatial-temporal factor graph into a probabilistic graphical model by representing temporal factors as skill-level transition distributions and spatial factors as constraint-satisfaction distributions. We eventually compose all the factors to construct a plan-level distribution directly at inference and use it to sample optimal node variables for the whole plan at once using reverse diffusion sampling.

A. Representing States, Skills, and Plans in Factor Graphs

States as factor graphs. We define a factor graph $\mathcal{G} = \{\mathcal{V}, \mathcal{F}\}$ of a state s consisting of the decision variables $\mathcal{V}$ and factors $\mathcal{F}$. Every robot and object in the scene can be considered as a decision variable node $v \in \mathcal{V}$ in the factor graph containing their respective observations and jointly representing the state of the environment. Following the definition of a standard factor graph [10], the inter-dependencies between the nodes are represented by factors $f \in \mathcal{F}$ in the form of certain relationships or constraints. For example, if we consider state s_0 illustrated in Figure 3, the nodes of a factor graph are {`arm`, `hammer`, `nail`} and the factors existing in this scene are {`Grasped(arm, hammer)=True`}. With an abuse of notation, we will consider $s \equiv \mathcal{G} = \{\mathcal{V}, \mathcal{F}\}$ for the rest of the paper.

Skills as temporal factors. We augment the definition of a parameterized skill for our factor graph representation. We define the preconditions of a skill as a set of nodes and factors. For example, a precondition of the skill that moves the hammer in hand to align with a nail `Move`(A, H) is that there exists a `Grasped` factor between the arm node A and the hammer node H. The skill is considered feasible *iff* the precondition nodes and factors are present in the factor graph of the current state. A new state and its factor graph $s' \equiv \mathcal{G}' = \{\mathcal{V}, \mathcal{F}'\}$ is created when a skill is applied. s' is created by modifying s with the effect of skill π, which changes the state of the nodes involved and, optionally, add or remove their factors. For example, the skill `Move`(A, H) modifies the state of the arm node A and the hammer node H and creates an `Aligned` constraint factor between the hammer H and the nail N. Finally, a temporal factor representing the state transition caused by executing the skill is created between s and s'. The continuous parameter of the skill a_{π_1} is added as a node to the factor, as it governs the behavior of the skill. An illustration of this two-state spatial-temporal graph is illustrated in Figure 3. The skill definitions can be extracted from standard PDDL symbolic skill operator with minor adaptations, following the duality of factor graphs and plan skeletons [15].

Plans as spatial-temporal factor graphs. Note that the single-step transition described above already constitutes a plan and thus an optimization problem: satisfying the `Aligned`, `Grasped`, and the transition dynamics constraints by finding the correct `Move` parameters a_{π_1}. Executing a plan causes temporal evolution to the initial state factor graph, creating a spatial-temporal factor graph. Each skill in a plan introduces additional nodes and factors to the factor graph, with added complexity for optimization. The learning-based optimization setup will be described in Section IV-C.

B. Representing Coordination in Factor Graphs

A key advantage of the factor graph representation is the ability to model multi-arm coordination tasks by connecting the temporal chains of each arm using spatial constraints. Such tasks often require skills to be simultaneously executed on each arm to operate on different or the same objects. We consider two cases for parallel skill execution, where multiple robots are operating on: (1) independent objects and (2) the same object, leading to independent and dependent temporal chains respectively. With our factorized state representation, we can independently control the execution of individual skills correlated with the nodes of interest. When the current state s satisfies the pre-condition of all required skills, the execution results in a factor graph created by applying the union of the effects of all the skills to the current factor graph.

Example. We consider a scenario shown in Figure 4 (Left). The left gripper arm L_0 is holding the pink cup C_0 (with factor `Grasped` between them) and the right gripper arm R_0 is holding the green cup M_0 (with factor `Grasped` between them). Both the grippers can independently execute the skill `Move` and modify separate factors in the factor graph ($f_1^{\pi_1}$ and $f_2^{\pi_2}$). Now, to successfully execute a skill `Pour`, one can

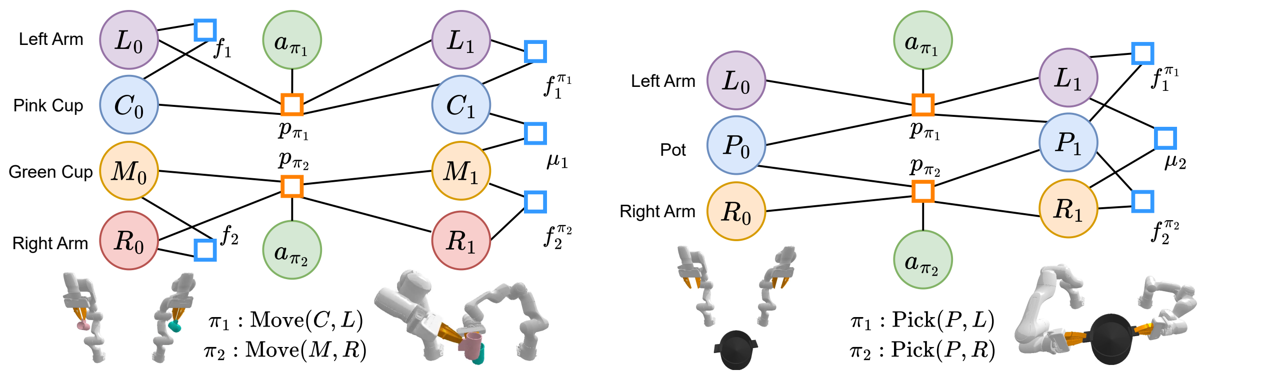

Fig. 4. (Left) **Parallel independent chaining** The figure shows the execution of two skills (π_1 and π_2) in-parallel on two independent sets of nodes (L, C and R, M) to modify their existing factors (`Grasped`). The two independent executions can be connected via external factors μ_1 (`FixedTransform`) introducing spatial dependencies between nodes C and M. (Right) **Parallel dependent chaining** The figure shows overlapping nodes of interest while parallel execution of two skills. The pot is to be picked by using both the arms simultaneously. The effect of this are resulting factors (`Grasped`) between (L, P and R, P) and external factor μ_2 (`FixedTransform`) between L and R. Overlapping nodes satisfy both skill's temporal effects.

add a constrained relationship factor (μ_1) between the two object nodes representing a set of favorable transforms that satisfy the precondition of `Pour`. Such an ability to augment constraints flexibly allows zero-shot coordination planning for unseen tasks at test time. Similarly, we can consider another scenario as shown in Figure 4 (Right), where there are two arms and a pot with no existing factors. With our construction of parallel chaining, a single object (P_0) can be picked by two grippers (L_0 and R_0) by only using information about how it can be picked with one arm (i.e. π_1 and π_2) and augmenting the spatial constraint factor (μ_2) between the two gripper nodes and the object node. This spatial factor will represent a fixed transform for subsequent planning.

C. Generative Factor Chaining

Now we have a formulation to construct a symbolic spatial-temporal factor graph plan for a task and chain them using spatial factor and temporal skill factors sequentially or in parallel. To make this plan geometrically feasible, we must find the optimal node variable values. While the underlying optimization problem can be solved using classical factor graph solvers in theory, it is difficult to model the transition dynamics for complex manipulation tasks in practice. Further, combining learning-based forward dynamics and sampling can also be employed, however, they provide less flexibility and modularity as shown by prior work [32]. In this work, we resort to a generative modeling formulation based on diffusion models. The key idea is to leverage the expressive generative model to capture the transition dynamics and exploit the flexible sampling strategies and compositionality of diffusion models. Our method, termed Generative Factor Chaining (GFC), can flexibly compose spatial-temporal factor distributions to sample optimal node variable values for the complete plan. Below we will first lay out the necessary formulation that bridges factor graph and probability distribution, and then we will describe GFC and its inference.

Probabilistic graphical model formulation. Solving for the whole spatial-temporal plan requires us to compute the joint distribution of all the decision variable nodes, in particular the skill action parameters. While prior work [32] formulates this simply as the joint distribution $p(s, a, s')$ of the vectorized states and skill parameters, we build a probabilistic graphical model with temporal skill-level distributions and spatial constraint satisfaction distributions. For an arbitrary spatial-factor graph with state-decision variables $\mathcal{V} = \{V_1, V_2, \dots, V_n\}$ and factors $\mathcal{F} = \{f_1, f_2, \dots, f_m\}$,

$$p(V_1, V_2, \dots, V_n) \propto \Pi_{\mathcal{F}} f_j(S_j),$$

where $S_j \subseteq \mathcal{V}$ and a factor f_j is included *iff* there is an edge between f_j and any one of $V_j \in \mathcal{V}$ which also implies $V_j \in S_j$. Further, we can flexibly consider additional constraint factors $\mu \in \mathcal{M}$ in such factor graph as

$$p(V_1, V_2, \dots, V_n) \propto \Pi_{\mathcal{F}} f_j(S_j) \ \Pi_{\mathcal{M}} f_\mu(S_\mu) \tag{3}$$

Now we can use the distribution $p(V_1, V_2, \dots, V_n)$ to be a distribution of the state $p(s)$ with all the spatial factors, and represent the temporal skill factor distribution of k^{th}-skill π_k as the joint distribution:

$$p_{\pi_k}(s, a, s') \equiv p_{\pi_k}(S_{\pi_k}, a, S'_{\pi_k}), \quad S_{\pi_k} \subseteq \mathcal{V}^{\pi_k}_{pre}$$

which is executable *iff* the skill's pre-condition $s^{\pi_k}_{pre} \equiv \{\mathcal{V}^{\pi_k}_{pre}, \mathcal{F}^{\pi_k}_{pre}\}$ is satisfied by the current state i.e. $\mathcal{V}^{\pi_k}_{pre} \subseteq \mathcal{V}$ and $\mathcal{F}^{\pi_k}_{pre} \subseteq \mathcal{F}$. Once executed, it leads to the transitioned state S'_{π_k}, a factor graph of the modified object nodes and spatial factors. For an example, one can consider Figure 4 (left) where the pre-condition of `Move`(C_0, L_0) is $\mathcal{V}^{\pi_k}_{pre}$=$\{C_0, L_0\}$ and $\mathcal{F}^{\pi_k}_{pre}$=$\{$`Grasped`$(C_0, L_0)\}$. Once executed, it results in adding a new factor $\{$`FixedTransform`$(C_1, M_1)\}$. Based on the above formulation of a short-horizon transition distribution, we extend to construct a trajectory-level distribution. We leverage the modularity of factored states by replacing states s with a set of decision variables S_{π_k} in the interest of skill π_k. This allows us to chain multiple skills in series and parallel. In such a scenario, the denominator term exists only for certain decision nodes *iff* they are common in two consecutive skills. For the sake of simplicity, we will formulate the probabilistic model for the two chains shown in Figure 4 by following the forward-backward analysis introduced by GSC and discussed in section III. We can write the top chain as:

$$p_{\pi_1}(L_0, C_0, a_{\pi_1}, L_1, C_1)p_{\pi_2}(R_0, M_0, a_{\pi_2}, R_1, M_1)p_{\mu_1}(C_1, M_1) \tag{4}$$

showing the independence of factors. Similarly, the bottom chain can be constructed as:

$$\frac{p_{\pi_1}(L_0, P_0, a_{\pi_1}, L_1, P_1)p_{\pi_2}(R_0, P_0, a_{\pi_2}, R_1, P_1)}{\sqrt{p_{\pi_1}(P_1)p_{\pi_2}(P_1)}}p_{\mu_2}(L_1, R_1) \tag{5}$$

where the factors are dependent on each other. It is worth noting that the augmented constraint factors p_μ work as a weighing function and can be more precisely represented by $p_\mu(S_\mu) \equiv p_\mu(y = 1|S_\mu)$ for some constraint-satisfaction index y. For example, after picking the pot in Figure 4 (right), the two arms must satisfy a fixed target transform (say T_p) while planning bimanually. Mathematically, $p_\mu\Big(\text{distance}(T_c, T_p) = 0|\{L_1, R_1\}\Big) = 1$ for any arbitrary transform T_c while planning.

Generative Factor Chaining We align towards diffusion model-based learned distributions to represent the probabilities in the formulated probabilistic graphical model. We transform the probabilities into their respective score functions $\epsilon(\mathbf{x}^{(t)}, t)$ for a particular reverse diffusion sampling step t and train it using the score-sde loss in Equation 1. Hence, for sampling a scene-graph for Equation 3, we can write as:

$$\epsilon(V_1^{(t)}, V_2^{(t)}, \ldots, V_n^{(t)}, t) = \sum_{\mathcal{F}} \epsilon_{f_j}(S_j^{(t)}, t) + \sum_{\mathcal{M}} \epsilon_{f_\mu}(S_\mu^{(t)}, t) \tag{6}$$

Similarly, we can show for the probabilistic chain in Equation 4 as:

$$\begin{gathered}\epsilon(L_0^{(t)}, C_0^{(t)}, R_0^{(t)}, M_0^{(t)}, L_1^{(t)}, C_1^{(t)}, R_1^{(t)}, M_1^{(t)}, t) = \\ \epsilon_{\pi_1}(L_0^{(t)}, C_0^{(t)}, a_{\pi_1}^{(t)}, L_1^{(t)}, C_1^{(t)}, t) + \\ \epsilon_{\pi_2}(R_0^{(t)}, M_0^{(t)}, a_{\pi_2}^{(t)} R_1^{(t)}, M_1^{(t)}, t) + \epsilon_{\mu_1}(C_1^{(t)}, M_1^{(t)}, t)\end{gathered}$$

and for the dependent factor chain in Equation 5 as:

$$\begin{gathered}\epsilon(L_0^{(t)}, P_0^{(t)}, R_0^{(t)}, L_1^{(t)}, P_1^{(t)}, R_1^{(t)}, t) = \\ \epsilon_{\pi_1}(L_0^{(t)}, P_0^{(t)}, a_{\pi_1}^{(t)}, L_1^{(t)}, P_1^{(t)}, t) + \\ \epsilon_{\pi_2}(R_0^{(t)}, P_0^{(t)}, a_{\pi_2}^{(t)} R_1^{(t)}, P_1^{(t)}, t) - \frac{1}{2}\epsilon_{\pi_1}(P_1^{(t)}, t) \\ -\frac{1}{2}\epsilon_{\pi_2}(P_1^{(t)}, t) + \epsilon_{\mu_2}(L_1^{(t)}, R_1^{(t)}, t)\end{gathered}$$

Generalization to new coordination tasks. We can realize from Equation 6 that the final score function depends on the composition of all the factors in the spatial-temporal factor graph. While factors $f \in \mathcal{F}$ are mostly modeled implicitly by the temporal skills, the external factors can be any arbitrary spatial constraints that ensure the satisfaction of the pre-condition of the subsequent skills. Hence, with new additions to the set of external factors $\mu' \in \mathcal{M}'$, one can reuse the same temporal skills with an added set of new spatial constraints.

Summary. GFC is a new paradigm to solve complex manipulation problems using spatial-temporal factor graphs. GFC can be divided into the following segments: (1) train individual skill factor distributions individually, (2) create spatial-temporal factor graph from a plan skeleton, (3) compose individual spatial and temporal factor distributions to construct a probabilistic graphical model, and (4) use the plan-level distribution to sample plan solutions. The proposed approach is modular as the individual skill factors and constraints can be flexibly connected to form new graphs. GFC can connect parallel skill chains with added spatial factors to solve coordinated manipulation problems directly at inference. Additional detail on algorithm is included in supplementary material.

V. Experiment

In this section, we seek to validate the following hypotheses: (1) GFC relaxes strict temporal dependency to allow spatial-temporal reasoning, performing better or on par with prior works in single-arm long-horizon sequential manipulation tasks, (2) GFC can effectively solve coordination tasks, and (3) GFC can flexibly plug and play skills to solve novel coordination planning problems. We perform a systematic evaluation on 9 long-horizon single-arm manipulation tasks from prior works and 4 complex multi-arm coordination tasks simulation. We also demonstrate deploying GFC on two Franka Panda manipulators in the real world.

A. Setup

Parameterized skills. We consider a finite set of parameterized skills: (1) `Pick`: Gripper picks up an object from the table and the parameters contain 6-DoF pose in the object's frame of reference, (2) `Place`: Gripper places an object at the target location and parameters contain 6-DoF pose in the place target's frame of reference, (3) `Move`: Gripper reaches a target location with an object in hand and parameters contain 6-DoF pose in the robot's frame of reference within the workspace, (4) `ReGrasp`: Gripper grasps object mid-air and parameters are same as pick, (5) `Push`: Gripper uses the grasped object to push away another object, (6) `Pull`: Gripper uses the grasped object to pull another object inwards, (7) `Strike`: Gripper strikes another object with one object in hand (e.g., a hammer), and (8) `Pour`: Gripper rotates the object in hand in a pouring fashion. While the first six skills can be trained as diffusion models, the last two are defined using designed motions with pre-condition as external dependency constraints, for example, `FixedTransform` (μ) in Figure 4. Following the baseline methods, all skills are trained and executed in the local frame of the robot of interest.

Relevant baselines and metrics: Our proposed method is based on factorized states and supports long-horizon planning for collaborative tasks directly at inference via probabilistic chaining. In this context, we consider prior method based on probabilistic chaining with vectorized states (**GSC** [32]) as a baseline. We further consider discriminative search-based approaches for solving long-horizon planning by skill chaining with uniform priors (**Random CEM**) and learned policy priors (**STAP** [1]). Since all prior works use sequential

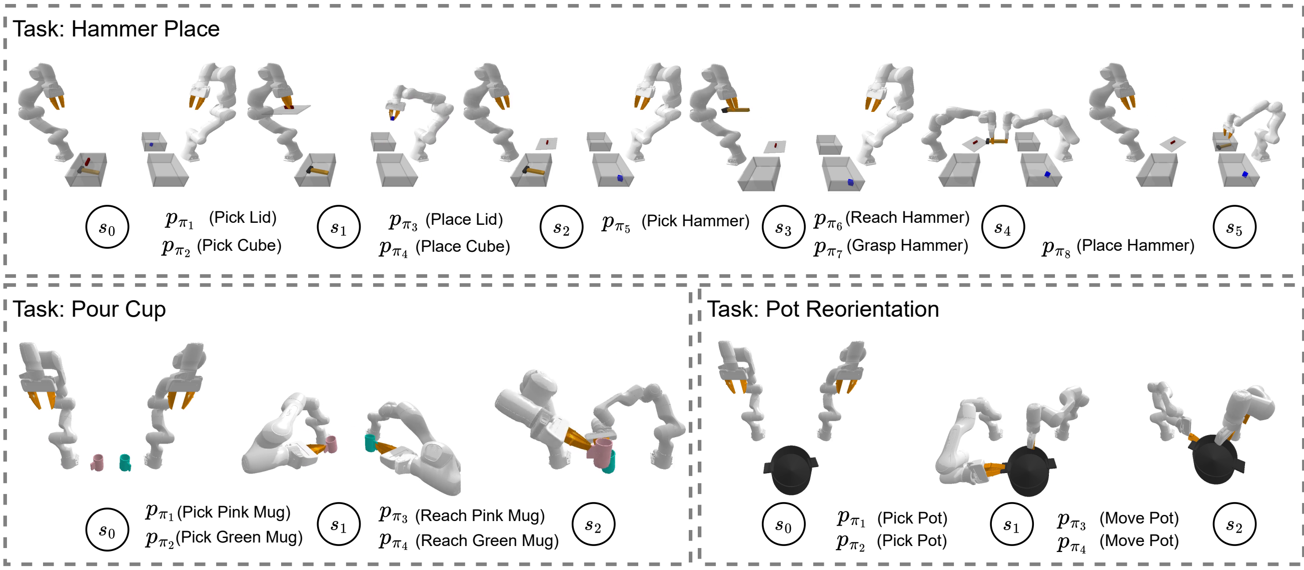

Fig. 5. **Optimal plan execution with sampled parameters.** We qualitatively evaluate the performance of our method in finding the optimal node variable values and skill parameters to solve for the plan and achieve the goal condition. We show successful solutions in simulation for the *Hammer Place* task (Top), *Pour Cup* task (bottom left), and *Bimnual Reorientation* task (bottom right).

planning, we compare the performance of the proposed method on the sequential version of the parallel skeleton.

Data collection and skill model training. To train skill models, we collect individual skill transition data in a single-manipulator setting and independent of any other skills in the library. We use an oracle controller in the robot frame of reference to sample suitable grasp positions for picking from the table and store the successful ones to train for `Pick` skill. We use several picked objects to reach randomly sampled target locations within the workspace of the robot for collecting the data for `Move` skill. Further, since it is not trivial to generate data for `Regrasp`, an inverse of the transition data from `Pick` and `Move` datasets is used. We predominantly use these three skills for all the considered tasks. For each transition, we collect the initial state of all the decision variables, the sampled action parameter, and the final states of the modified decision variable. While we collect observations for all the decision variables, the ones independent to the skill of interest are used as conditioning variables while training. We train the skill factor distributions using the denoising score-matching loss as discussed in section III. Our diffusion model architecture uses an identical Transformer backbone as GSC, except it predicts only for nodes of interest at the current step. Additional detail on model training and architecture is included in supplementary material.

Real robot setup: Similar to prior learning-based methods [1, 32], GFC trained in simulation can be used to perform sequential manipulation tasks in a real robot environment. We take RGB-D images from a Kinect Azure camera and perform semantic segmentation to locate objects and estimate their poses with the corresponding point cloud. The estimated state is then used to construct the environment for planning. Additional qualitative results on robot hardware are provided in the supplementary material.

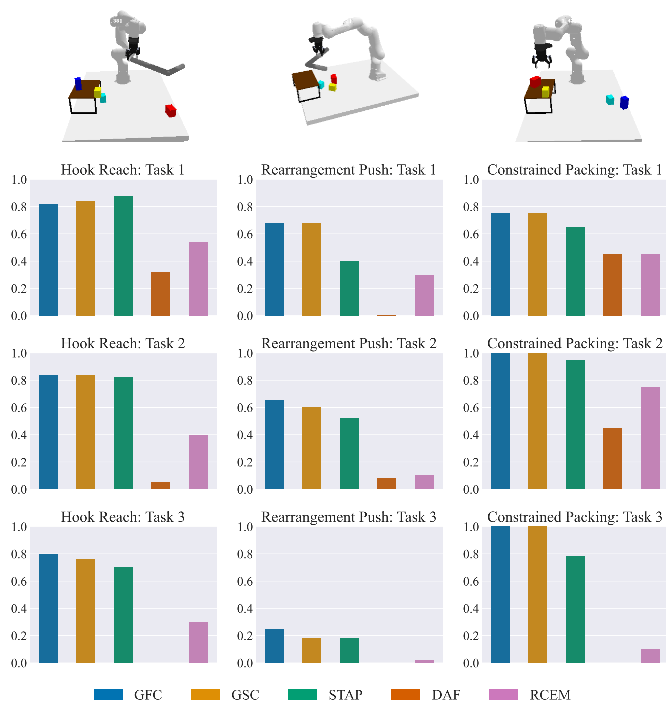

Fig. 6. **Results on single-manipulator task suite [1, 32].** GFC consistently achieves success rate that are as good as prior works while outperforming them as the horizon length increases. It leverages the factorized states to discern between independent skill executions. Like all the baseline algorithms, GFC success rate is reported with 100 trials.

B. Key Findings

GFC relaxes strict linear dependency assumptions. We first evaluate GFC on single-manipulator long-horizon tasks introduced by STAP [1] and also used by GSC [32]. These tasks consider manipulation by reasoning about the usage of a tool (a hook) to manipulate blocks out of or into the

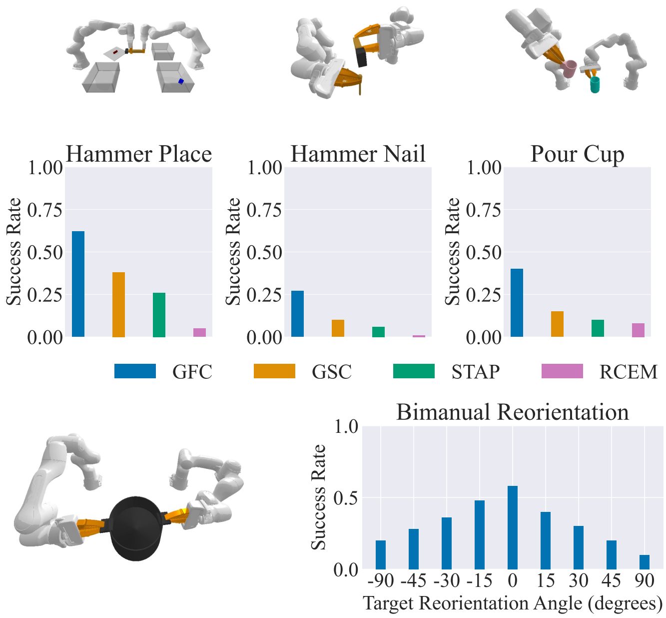

Fig. 7. **Results on coordinated two-manipulator tasks.** We consider four tasks to evaluate parallel skill execution and spatial-temporal composition capability of GFC. After testing on 100 scenarios, we significantly outperform baselines with sequential chaining for *Hammer Place Hammer Nail* and *Pour Cup* tasks. No other methods can support the *Bimanual Reorientation* task. We included a detailed evaluation on task success rate given different reorientation goals. A plan is successful if the goal condition is satisfied like `AinB`(Hammer, Box), `onTopOf`(Hammer, Nail), `FixedTransform`(Pink Cup, Green Cup) and `FixedTransform`(Pot, Ground)

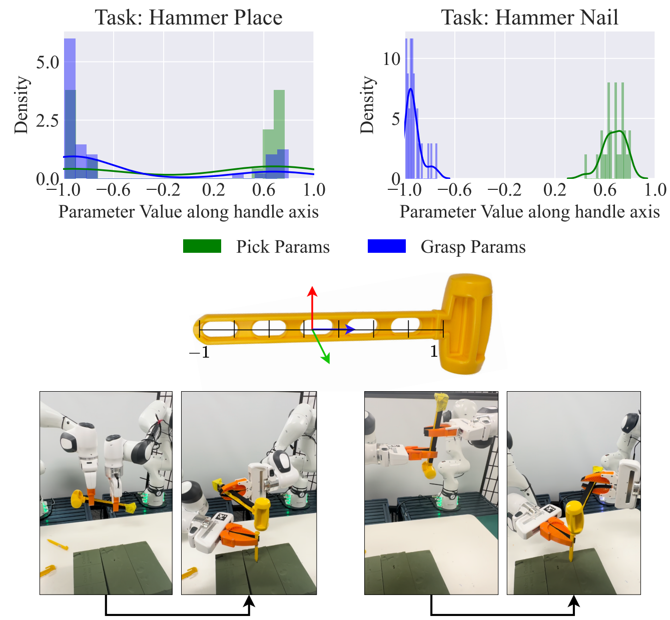

Fig. 8. **Analysis of sampled `Pick` and `Grasp` skill parameters** We show that the planner is able to reason about the long-horizon action dependency of `Pick` and `Grasp` skills. (Top) While we see that *Hammer Place* can be solved by pick/grasp at head/tail and vice versa, to satisfy the precondition of `Strike` in *Hammer Nail*, the hammer must be grasped near tail so must be picked near head. (Bottom) We further show orientation reasoning as well, where the hammer can either be grasped on the same side or the flip side.

robot workspace (sample initial states shown in Figure 6). *Hook Reach* is to hook the cube in order for the arm to grasp and move the block to a target. *Rearrangement Push* requires placing a cube such that it can be pushed beneath a rack using the tool. *Constrained Packing* is to place four cubes on a constrained surface without collisions. While these tasks are originally designed to highlight linear sequential dependencies, there are steps with indirect dependencies or independence that only GFC can effectively model because of the factorized states. For example, in *Rearrangement Push*, the picking pose of the cube should not affect the tool use steps. As shown in Figure 6, we observe that the performance of GFC is consistently on-par with the baseline for tasks with strict linear dependencies such as *Hook Reach* and on-par or better for tasks with more complex dependency structures such as *Rearrangement Push*. This validates our hypothesis that GFC effectively models sequential dependencies, in addition to independence and skipped-step dependencies in long-horizon tasks.

GFC can solve complex coordinated manipulation tasks. Here, we aim to validate that GFC can effectively model and solve different types of coordinated manipulation tasks. We present results on tasks with increased collaboration challenges. First, we consider tasks that require coordination but can be serialized into interleaved skill chains and solved by prior skill-chaining methods. *Hammer Place*, as shown in Figure 5, is for one arm to pick a hammer from a box, hand it over to the second arm to be placed into another box. *Hammer Nail* is for the first arm to pick up the hammer, handover to the second arm, and pick up a nail. Both arms then move to positions such that the head of the hammer is aligned with nail for the subsequent hammering step. The task is illustrated in Figure 2 (bottom). As shown in Figure 7, our method significantly outperforms all baselines in both tasks. The proportional gap is larger in the more challenging *Hammer Nail* task, which includes additional spatial and temporal constraints such as the hammer must be re-grasped towards the tail end for the subsequent hammering step, and the hammer and nail must be aligned for a successful strike. This demonstrates that GFC can effectively model and resolve both spatial and temporal constraints in complex tasks.

Finally, we consider the *Bimanual Reorientation* (Figure 5) task where two arms must simultaneously operate on the same object of interest. The task is for both arms to pick a pot, lift it up, and try to rotate it to a target reorientation angle (about z-axis) as illustrated for a 45-deg angle. The tasks must be solved via parallel skill chaining with spatial constraints and hence none of the prior baselines can be used. The factor graph includes a spatial fixed transform constraint between both the arms and hence the subsequent skills operate while satisfying the constraint. Figure 7 (bottom) shows a detailed task success rate breakdown given different orientation goals. The task poses a considerable challenge as it requires GFC to reason about suitable `Pick` pose that would not violate either robot's kinematic constraint at the final reoriented placing pose (temporal constraint), while simultaneously satisfying the fixed transform constraint of holding the pot (spatial constraint).

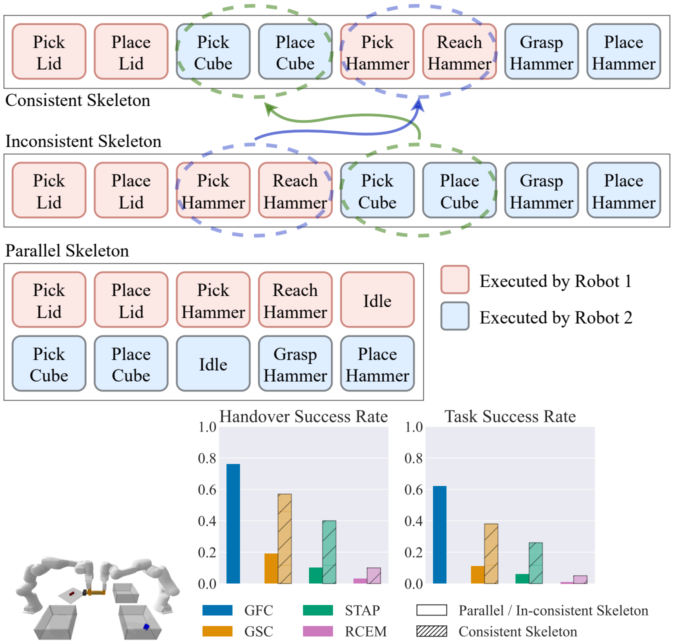

Fig. 9. **Linear chaining has limitations.** The above figure describes the three types of ablation skeletons prepared for evaluating the disadvantages of imposing strict linear dependency. The performance in-consistent chain is significantly lower than that in the consistent chain for baselines while GFC uses the parallel skeleton.

GFC can generalize to unseen coordination plans. Here, we show that GFC can solve unseen coordinated manipulation tasks by reusing skills and enforcing additional spatial constraints at inference time. The *Pour Cup* task is to `Pick` one cup with each arm, `Move` to position the two arms, and `Pour` the content of one into the other. GFC can directly reuse `Pick` and `Move` skill models and adapt the `Strike` skill model for the `Pour` step by introducing a new spatial constraint. Unlike hammer that can strike from either face of the head, the cups can only be poured using the open top and not the closed bottom. The constraint can be directly added as a factor and optimized globally through guided diffusion process. We note that other methods can also be adapted to solve new tasks. For examples, GSC can linearize the skill chain and add additional goal constraints, and STAP can similarly plan through its forward dynamics. GFC supports such zero-shot generalization by design and facilitates chaining multiple independent temporal chains to satisfy desired spatial factors. Detailed quantitative comparison is shown in Figure 7.

GFC can handle independence and inconsistent skill orders. Here, we analyze how *independent* steps in a sequential manipulation chain affects the performance of each method. We consider *Hammer Place*, where the order of transporting the cube and handing over hammer is interchangeable. As illustrated in Figure 9, we consider a *consistent* plan skeleton where sequentially-dependent steps for the two main objectives, i.e., (1) opening lid then transporting cube and (2) picking, handing over, and placing hammers, are completely sequentially. We also consider an *inconsistent* plan skeleton where the steps are interleaved. We show the handover success and overall task success in Figure 9 (Bottom). A successful handover requires choosing non-overlapping parameters for `Pick` and `Regrasp` skills along with a `Move` parameter which allows grasping. While this increases the difficulty leading to lower scores in the handover success rate, even with a minor distraction in *inconsistent* skeleton, the previous approaches failed to propagate the skipped-step dependencies as evident from the task success rate.

Analyzing long-horizon reasoning ability in coordinated manipulation. As discussed before, in order to perform a successful handover in *Hammer Place*, a planner must generate suitable `Pick` and `Move` skill parameters such that the hammer becomes graspable and a suitable parameter for `ReGrasp` skill exists for the other arm. Additional task constraint is imposed in the *Hammer Nail* task, where the robot must grasp the hammer by the tail end to perform a successful `Strike`. We conducted a focused analysis of such reasoning capabilities quantitatively and qualitatively.

We observe in Figure 8 (top left) that while *Hammer Place* task can be solved by picking or grasping on any end of the hammer handle, *Hammer Nail* requires more constrained parameter sampling as shown in Figure 8 (top right). Further, it is worth noting that along with the parameter selection along the handle axis, the method also samples suitable orientation (same or flip side) for grasping as shown by two examples in Figure 8 (bottom) leading to different strike positions.

VI. Limitations and Future Directions

While powerful, GFC presents many opportunities for future works. First, our proposed method does not generate high-level task plans. While it is possible to directly perform a symbolic search with the skill operators we adopted, we consider solving the full TAMP problem in our framework an important future direction. Second, it is important to note that our method assumes full observability and operates in a low-dimensional state space. A future direction is to extend GFC to a learned latent space. Finally, similar to prior works [50, 1, 32], our approach relies on a fixed set of primitive skills. Future work can explore integrating learned skills or trajectory generators for additional generality and scalability.

VII. Conclusion

We presented GFC, a learning-to-plan method to solve complex coordinated manipulation planning problems with given skeleton plans. GFC can flexibly plan for multiple arms sequentially or in parallel operating on one or more objects in the scene. It uses spatial factor graph to represent states and uses temporal skill factors to chain them. The final spatial-temporal plan is solved by composing the spatial factor distributions and diffusion model-based skill distributions. We compose short-horizon skills to sample from a plan-level probabilistic graphical model distribution. GFC is shown to solve sequential and coordinated tasks directly at inference and reason about long-horizon action dependency across multiple temporal chains. The proposed framework is flexible, scalable and generalizes well to unseen multiple-manipulator tasks.

References

[1] Christopher Agia, Toki Migimatsu, Jiajun Wu, and Jeannette Bohg. Taps: Task-agnostic policy sequencing. *arXiv preprint arXiv:2210.12250*, 2022.

[2] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[3] Fabio Amadio, Adrià Colomé, and Carme Torras. Exploiting symmetries in reinforcement learning of bimanual robotic tasks. *IEEE Robotics and Automation Letters*, 4(2):1838–1845, 2019.

[4] Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[5] Akhil Bagaria and George Konidaris. Option discovery using deep skill chaining. In *International Conference on Learning Representations*, 2020.

[6] Jingkai Chen, Jiaoyang Li, Yijiang Huang, Caelan Garrett, Dawei Sun, Chuchu Fan, Andreas Hofmann, Caitlin Mueller, Sven Koenig, and Brian C Williams. Cooperative task and motion planning for multi-arm assembly systems. *arXiv preprint arXiv:2203.02475*, 2022.

[7] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric A. Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *ArXiv*, abs/2303.04137, 2023.

[8] Rohan Chitnis, Shubham Tulsiani, Saurabh Gupta, and Abhinav Gupta. Efficient bimanual manipulation using learned task schemas. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1149–1155. IEEE, 2020.

[9] Alexander Clegg, Wenhao Yu, Jie Tan, C Karen Liu, and Greg Turk. Learning to dress: Synthesizing human dressing motion via deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 37(6):1–10, 2018.

[10] Frank Dellaert. Factor graphs: Exploiting structure in robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, 4:141–166, 2021.

[11] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

[12] Danny Driess, Jung-Su Ha, and Marc Toussaint. Learning to solve sequential physical reasoning problems from a scene image. *The International Journal of Robotics Research*, 40(12-14):1435–1466, 2021.

[13] Yilun Du, Shuang Li, and Igor Mordatch. Compositional visual generation with energy based models. *Advances in Neural Information Processing Systems*, 33:6637–6647, 2020.

[14] Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and P. Abbeel. Learning universal policies via text-guided video generation. *ArXiv*, abs/2302.00111, 2023.

[15] Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4:265–293, 2021.

[16] Jennifer Grannen, Yilin Wu, Brandon Vu, and Dorsa Sadigh. Stabilize to act: Learning to coordinate for bimanual manipulation. In *Conference on Robot Learning*, pages 563–576. PMLR, 2023.

[17] Huy Ha, Jingxi Xu, and Shuran Song. Learning a decentralized multi-arm motion planner. *arXiv preprint arXiv:2011.02608*, 2020.

[18] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[20] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.

[21] Leslie Pack Kaelbling and Tomás Lozano-Pérez. Learning composable models of parameterized skills. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 886–893. IEEE, 2017.

[22] Ivan Kapelyukh, Vitalis Vosylius, and Edward Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *IEEE Robotics and Automation Letters*, 2023.

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

[24] George Konidaris, Scott Kuindersma, Roderic Grupen, and Andrew Barto. Robot learning from demonstration by constructing skill trees. *The International Journal of Robotics Research*, 31(3):360–375, 2012.

[25] George Konidaris, Leslie Pack Kaelbling, and Tomas Lozano-Perez. From skills to symbols: Learning symbolic representations for abstract high-level planning. *Journal of Artificial Intelligence Research*, 61:215–289, 2018.

[26] Youngwoon Lee, Shao-Hua Sun, Sriram Somasundaram, Edward S Hu, and Joseph J Lim. Composing complex skills by learning transition policies. In *International Conference on Learning Representations*, 2019.

[27] Youngwoon Lee, Joseph J Lim, Anima Anandkumar, and Yuke Zhu. Adversarial skill chaining for long-horizon robot manipulation via terminal state regularization. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=K5-J-Espnaq.

[28] Jacky Liang, Mohit Sharma, Alex LaGrassa, Shivam Vats, Saumya Saxena, and Oliver Kroemer. Search-based task planning with learned skill effect models for lifelong robotic manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6351–6357. IEEE, 2022.

[29] Andreas Lugmayr, Martin Danelljan, Andres Romero,

Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022.

[30] Warwick Masson, Pravesh Ranchod, and George Konidaris. Reinforcement learning with parameterized actions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[31] Utkarsh A Mishra and Yongxin Chen. Reorientdiff: Diffusion model based reorientation for object manipulation. *arXiv preprint arXiv:2303.12700*, 2023.

[32] Utkarsh Aashu Mishra, Shangjie Xue, Yongxin Chen, and Danfei Xu. Generative skill chaining: Long-horizon skill planning with diffusion models. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=HtJE9ly5dT.

[33] Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

[34] Ludwig Nägele, Alwin Hoffmann, Andreas Schierl, and Wolfgang Reif. Legobot: Automated planning for coordinated multi-robot assembly of lego structures. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9088–9095. IEEE, 2020.

[35] Soroush Nasiriany, Huihan Liu, and Yuke 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.

[36] Hanna M Pasula, Luke S Zettlemoyer, and Leslie Pack Kaelbling. Learning symbolic models of stochastic domains. *Journal of Artificial Intelligence Research*, 29: 309–352, 2007.

[37] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023.

[38] Moritz Reuss and Rudolf Lioutikov. Multimodal diffusion transformer for learning from play. In *2nd Workshop on Language and Robot Learning: Language as Grounding*, 2023. URL https://openreview.net/forum?id=nvtxqMGpn1.

[39] Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

[40] Dhruv Shah, Peng Xu, Yao Lu, Ted Xiao, Alexander Toshev, Sergey Levine, and Brian Ichter. Value function spaces: Skill-centric state abstractions for long-horizon reasoning. *arXiv preprint arXiv:2111.03189*, 2021.

[41] Tom Silver, Rohan Chitnis, Nishanth Kumar, Willie McClinton, Tomas Lozano-Perez, Leslie Pack Kaelbling, and Joshua Tenenbaum. Inventing relational state and action abstractions for effective and efficient bilevel planning. *arXiv preprint arXiv:2203.09634*, 2022.

[42] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[43] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

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

[45] Albert Tung, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Learning multi-arm manipulation through collaborative teleoperation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9212–9219. IEEE, 2021.

[46] Lucia Pais Ureche and Aude Billard. Constraints extraction from asymmetrical bimanual tasks and their use in coordinated behavior. *Robotics and autonomous systems*, 103:222–235, 2018.

[47] Nghia Vuong, Hung Pham, and Quang-Cuong Pham. Learning sequences of manipulation primitives for robotic assembly. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4086–4092. IEEE, 2021.

[48] Zi Wang, Caelan Reed Garrett, Leslie Pack Kaelbling, and Tomás 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)*, pages 4107–4114. IEEE, 2018.

[49] Victoria Xia, Zi Wang, and Leslie Pack Kaelbling. Learning sparse relational transition models. *arXiv preprint arXiv:1810.11177*, 2018.

[50] Danfei Xu, Ajay Mandlekar, Roberto Martín-Martín, Yuke Zhu, Silvio Savarese, and Li Fei-Fei. Deep affordance foresight: Planning through what can be done in the future. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6206–6213. IEEE, 2021.

[51] Zhutian Yang, Jiayuan Mao, Yilun Du, Jiajun Wu, Joshua B Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Compositional diffusion-based continuous constraint solvers. *arXiv preprint arXiv:2309.00966*, 2023.

[52] Tianhe Yu, Ted Xiao, Austin Stone, Jonathan Tompson, Anthony Brohan, Su Wang, Jaspiar Singh, Clayton Tan, Jodilyn Peralta, Brian Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[53] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *arXiv preprint arXiv:2204.13902*, 2022.

[54] Qinsheng Zhang, Molei Tao, and Yongxin Chen. gddim: Generalized denoising diffusion implicit models. *arXiv preprint arXiv:2206.05564*, 2022.

[55] Qinsheng Zhang, Jiaming Song, Xun Huang, Yongxin Chen, and Ming-Yu Liu. Diffcollage: Parallel generation of large content with diffusion models. *ArXiv*, abs/2303.17076, 2023.