2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

MOSAIC: Skill-Centric Manipulation Planning with Physics Simulation

Itamar Mishani, Yorai Shaoul, and Maxim Likhachev
Robotics Institute, School of Computer Science
Carnegie Mellon University, United States
{imishani, yoraish, maxim}@cs.cmu.edu

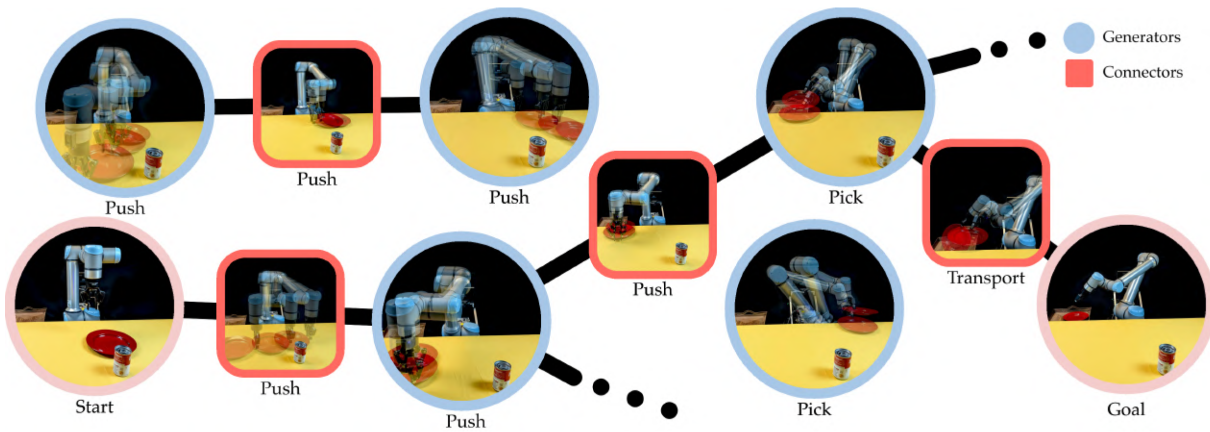


Fig. 1: MOSAIC solves long-horizon manipulation tasks by generating local skill trajectories (circles) and connecting those with connector skills (squares). MOSAIC capitalizes on the skills themselves to guide the exploration process toward regions where they are likely to succeed – enabling effective composition of generic local skills to solve complex tasks.

***Abstract*— Planning long-horizon manipulation motions using a set of predefined skills is a central challenge in robotics; solving it efficiently could enable general-purpose robots to tackle novel tasks by flexibly composing generic skills. Solutions to this problem lie in an infinitely vast space of parameterized skill sequences – a space where common incremental methods struggle to find sequences that have non-obvious intermediate steps. Some approaches reason over lower-dimensional, symbolic spaces, which are more tractable to explore but may be brittle and are laborious to construct. In this work, we introduce MOSAIC, a skill-centric, multi-directional planning approach that targets these challenges by reasoning about which skills to employ and where they are most likely to succeed, by utilizing physics simulation to estimate skill execution outcomes. Specifically, MOSAIC employs two complementary skill families: *Generators*, which identify "islands of competence" where skills are demonstrably effective, and *Connectors*, which link these skill-trajectories by solving boundary value problems. By focusing planning efforts on regions of high competence, MOSAIC efficiently discovers physically-grounded solutions. We demonstrate its efficacy on complex long-horizon problems in both simulation and the real world, using a diverse set of skills including generative diffusion models, motion planning algorithms, and manipulation-specific models. Visit `skill-mosaic.github.io` for demonstrations.**

I. INTRODUCTION

Recent progress in robotics has enabled learning complex manipulation skills that can be executed reliably under certain controlled scenarios. However, deploying robots in unstructured environments such as homes and offices requires solving longer-horizon tasks using skills not necessarily trained for those conditions. For example, a robot tidying a cluttered table must autonomously discover how to reorient plates for stable grasps, avoid wet areas, and coordinate sweeping motions. Such scenarios demand reasoning over long horizons and composing a diverse set of imperfect skills–not only sequencing them but also identifying conditions under which they are likely to succeed.

Existing approaches typically represent two extremes. On the one hand, policy learning methods succeed in short-horizon tasks but struggle with long-horizon reasoning and skill composition due to large data requirements and poor generalization without substantial retraining. On the other hand, Task and Motion Planning (TAMP) frameworks offer hierarchical reasoning but are constrained by their reliance on explicit symbolic representations, which often fail to capture critical geometric and physical details, compromising robustness in open-world settings. Surprisingly, despite their varying strategies, many existing algorithms are fundamentally similar in that they adopt a goal-directed search, exploring outward from the start state or backward from the goal. However, this approach can be inefficient when solutions depend on non-obvious intermediate steps that do not offer an immediate indication of progress toward the goal.

In this work, we introduce MOSAIC, an algorithmic framework for long-horizon planning that positions skills as key stakeholders in the planning process, enabled by a high-fidelity physics simulator that evaluates their feasibility during search. Rather than being constrained by directed search, MOSAIC explores multiple directions simultaneously, anchoring its search in regions of the state space where skills are most likely to succeed. The approach leverages *generator skills* to propose effective skill-trajectories and world configurations, while utilizing *connector skills* to generate transitions between these regions by solving boundary value problems. Using simulation as world model, MOSAIC finds solutions that are physically grounded.

Our contributions are:

- A feasibility-driven planning framework, MOSAIC, that shifts from goal-directed to a multi-directional search anchored in regions of high skill competence, along with its theoretical foundations.
- A physics-informed reasoning approach that uses high-fidelity simulation to ground planning decisions in physical reality through in-the-loop evaluation.
- Extensive validation in both simulation and the real world, demonstrating that MOSAIC outperforms existing methods in performance, generalization, and scalability on complex manipulation tasks.

II. RELATED WORK

We categorize prior works into four main areas: Model-Based Planning with Simulation, Task and Motion Planning, Single Policy Methods, and Skill-Based Methods.

Model-Based Planning with Simulation remains fundamental for autonomous decision-making in complex environments, enabling systematic reasoning about action consequences and exploration of alternative strategies before execution. This is particularly crucial for long-horizon manipulation tasks where myopic decisions can lead to dead ends or suboptimal solutions. To plan effectively, we need a world model that can estimate the outcomes of different actions. Physics simulators have seen remarkable advances, with modern platforms like Isaac Sim [1], Genesis [2], Sapien [3], and MuJoCo [4] now capable of modeling complex contact dynamics with unprecedented accuracy and efficiency. These advances have enabled zero-shot sim2real transfer [5] and real2sim2real [6] workflows that allow robots to perform complex tasks in the real world. A growing body of work has demonstrated the effectiveness of integrating physics simulators directly into the planning loop. Saleem and Likhachev [7], Saxena et al. [8], Saxena and Likhachev [9] developed algorithms for manipulation planning among movable objects using selective physics-based simulation evaluations, solving complex manipulation problems involving both prehensile and non-prehensile manipulation. There is also significant work on using physics simulators and differentiable physics simulators for optimization through contact-rich scenarios [10–12]. While learned world models [13, 14] show promise for efficient world modeling, physics simulators currently provide the most reliable predictions for manipulation tasks involving complex physical interactions. In this work, MOSAIC leverages physics simulation as a high-fidelity world model to ground planning decisions in physical reality. MOSAIC is designed to accommodate any world model that can evaluate skill outcomes, and will continue improving as world models increase in fidelity and efficiency.

Task and Motion Planning (TAMP) addresses long-horizon planning problems by interleaving abstract symbolic task planning with geometric motion planning. TAMP systems require users to define a symbolic task specification and a symbolic world state that can be augmented by actions [15]. These systems are powerful but require manual engineering and privileged task-specific information for determining the symbolic representations and available transitions [16]. While there is a significant body of work that aims to learn different TAMP modules to avoid manual engineering [17–24], designing appropriate learning modules that interface with existing symbolic planners and collecting training data to learn accurate symbolic abstractions is challenging. Recent hybrid approaches, such as [25], combine TAMP with learning by decomposing tasks symbolically and training policies for subtasks. MOSAIC takes a different approach and eliminates the need for symbolic task specifications. Instead, it relies on physics simulators to directly forward simulate any skills considered during its search – an attractive alternative made possible by the efficiency and high-fidelity of modern simulators.

Single Policy Methods learn to solve tasks under specified goal conditions. Diffusion models have been used for this purpose [26, 27], showing strong performance in manipulation but struggling with generalization and long-horizon reasoning. In reinforcement learning, automatic goal generation [28, 29] and option discovery [30] incorporate domain knowledge via structural priors into the learning process to guide exploration and improve sample efficiency. These methods make full-horizon learning more tractable by injecting task-specific information. However, policy learning methods typically succeed only in short-horizon tasks and struggle with the long-horizon reasoning and skill composition required for complex manipulation due to large data requirements and poor generalization without substantial retraining. In contrast, MOSAIC leverages simple, task-agnostic skill primitives to solve diverse long-horizon tasks through intelligent composition rather than end-to-end learning.

Skills-based Methods aim to compose skills (e.g., learned policies, analytic controllers) to solve long-horizon tasks. They typically address two challenges: selecting *which skills* to compose (a discrete problem) and *how to parameterize* them (often a continuous problem). A central approach is skill chaining [31], which, building on the options framework [32], incrementally discovers local policies that achieve subgoals [33–36]. Other methods [37] explicitly select skills and parameters, or learn skills as diffusion models [38, 39]. However, the sequential structure of these frameworks forces skills to define initiation, termination, and effect sets, or assumes access to plan skeletons, limiting their flexibility and reuse across tasks. Sivaramakrishnan et al. [40] relax some of these requirements by learning a goal-conditioned dynamics policy in an obstacle-free space, then building a roadmap to guide planning with the learned controller. Similarly, [41] use sampling-based planners to explore the abstract space of higher-order skills in lifelong RL.

A fundamental limitation shared by existing skill-based methods–and in most planning approaches–is their reliance on *goal-directed search*: forward methods choose skills based on reachable termination sets from the start state, while backward methods select skills based on initialization sets leading to the goal state. Both schemes constrain exploration to local neighborhoods around current or goal states, making it difficult to identify distant but executable skills that could aid task completion. This approach can be particularly inefficient when solutions depend on non-obvious intermediate steps that offer no immediate indication of progress toward the goal. Critically, skills are *imperfect*–they succeed under certain conditions but may fail in others due to environmental constraints, object configurations, or inherent limitations in their training data. To the best of our knowledge, no existing planning framework explicitly reasons about where skills are likely to succeed and directs planning efforts accordingly.

MOSAIC differs by addressing this gap through a feasibility-driven, multi-directional search that plans with imperfect skills. Rather than exploring skill sequences in a goal-directed fashion, MOSAIC solves complex long-horizon tasks by explicitly reasoning about skill competence and discovering regions where skills are most likely to succeed.

III. Problem Definition: Skill-Centric Planning

We seek to find a sequence of parameterized skills, out of a given library of available skills, whose execution would modify a world state from its current specification to a specification satisfying a goal condition.

Formally, let $\mathcal{Q}_\mathcal{R} \subseteq \mathbb{R}^n$ denote the configuration space of a robot $\mathcal{R}$ with n degrees-of-freedom (DOF), and let $\mathcal{X} \subseteq \mathbb{R}^m$ be the state space of the planning problem, often called the *world state*, where $n \leq m$. To facilitate set membership checks for states (e.g., whether a state is a goal state or belongs to some equivalence class), let us define *binary conditions* $\xi : \mathcal{X} \to 0, 1$ with $\xi \in \Xi$. We define a *trajectory* $\boldsymbol{\tau}$ as a mapping $\boldsymbol{\tau} : [0, 1] \to \mathcal{X}$, and T as the trajectory space. The space of *parameterized skills*, i.e., motor controllers that map skill parameters θ to trajectories $\boldsymbol{\tau} \in \mathrm{T}$ [42], is termed the skill space $\mathcal{A}$.

Our definition of *parameterized skills* is related to the *options* framework [32]. A skill σ is a temporally extended action, represented as a tuple $(\pi_\sigma, \Theta_\sigma)$, where π_σ is the skill policy that returns the probability of taking action a in state x given parameters $\theta \in \Theta_\sigma$ via $\pi_\sigma(a|x, \theta)$. Actions a may denote low-level control commands (e.g., joint velocities) or higher-level motion segments (e.g., waypoint sequences). The outcome of a skill is a trajectory $\boldsymbol{\tau} \in \mathrm{T}$.

Crucially, this does not imply that skills must always be executed in an open-loop fashion. Trajectories serve as the planning representation of skills, but execution can remain closed-loop. Skills may be realized as reactive policies (e.g., diffusion policies [27]), in which case their trajectories are obtained by rollout; as deterministic planners, where trajectories are generated procedurally; or as hybrid methods that combine both. Thus, skills provide a unified interface for planning—via trajectories—while retaining flexibility in execution. We next distinguish between two types of skills tailored to our framework.

Definition 1. ***Generators****, denoted as $\mathcal{G}_i : \Theta_{\mathcal{G}_i} \to \mathrm{T}$, are parameterized skills that generate trajectories without requiring specified start and goal states.*

Definition 2. ***Connectors****, denoted as $\mathcal{C}_j : \Xi \times \Xi \times \Theta_{\mathcal{C}_j} \to \mathrm{T}$, are conditional parameterized skills that generate trajectories conditioned on specified start and goal conditions.* [1]

With these definitions, we formally define the skill space as $\mathcal{A} = \bigcup_i \mathcal{G}_i \cup \bigcup_j \mathcal{C}_j$. Given a start state x_{start} and goal condition function $\xi_{\text{goal}} : \mathcal{X} \to \{0, 1\}$ the objective of the planning problem is to find a sequence of N skills $\{\sigma_i \mid \sigma_i \in \mathcal{A}\}_{i=1}^N$ and associated parameters $\{\theta_i \mid \theta_i \in \Theta_{\sigma_i}\}_{i=1}^N$ that produce the sequence of trajectories $\Pi = \{\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_N\}$ that satisfies the following conditions:

$$\boldsymbol{\tau}_1(0) = x_{\text{start}} \tag{1}$$

$$\xi_{\text{goal}}(\boldsymbol{\tau}_N(1)) = 1 \tag{2}$$

$$\boldsymbol{\tau}_i(1) = \boldsymbol{\tau}_{i+1}(0) \quad \forall i \in \{1, \ldots, N-1\} \tag{3}$$

That is, the solution sequence must satisfy three key properties: the initial state of the first trajectory must be equal to the start state (Eq. 1), adjacent trajectories must connect continuously (Eq. 3), and the final state of the last trajectory must satisfy the goal condition (Eq. 2). Skills produce only *valid* trajectories $\boldsymbol{\tau}_i$: not causing collisions between the robot and static obstacles and respecting robot dynamics. Additional constraints (e.g., motion smoothness) and optimization objectives (e.g., execution time) may be imposed on the solution.

A. Skill-centric Manipulation Planning

Robotic manipulation refers to the process of altering the state of objects through physical interaction. A fundamental characteristic of manipulation planning is that the system is typically underactuated – the number of actuated DOFs is smaller than the dimension of the state space $\mathcal{X}$. For each movable object $\mathcal{O}_i$ we define a configuration space $Q_{\mathcal{O}_i} \subseteq SE(3)$. Given a robot's configuration space $\mathcal{Q}_\mathcal{R}$, the complete state space is then defined as the product of all configuration spaces, $\mathcal{X} = \mathcal{Q}_\mathcal{R} \times Q_{\mathcal{O}_1} \times \cdots \times Q_{\mathcal{O}_k}$, where k is the number of movable objects. Since movable objects are included in the state space and are not directly actuated, the state space dimension m is generally greater than the robot's configuration space dimension n, resulting in an underactuated system. Therefore, generating a new state during planning requires specifying not only the robot's configuration but also the configurations of all movable objects. To accurately capture how robot actions affect the environment, we rely on a model–in our case, a physics simulator–that can predict the outcomes of these interactions.

IV. Mosaic

The core concept of MOSAIC is the construction of a directed multigraph, which we call a *mosaic graph*. In this graph, nodes are tuples of skills, parameters and the estimated generated trajectories, and edges are tuples of skills, parameters, and boundary conditions. Nodes are created with *generator* skills, which produce local behavior trajectories, and edges are created with *connector* skills, which link the generated local trajectories. Since constructing this graph

[1] A common start and goal condition for connector skills is an equality condition to given states x' and x''. Therefore, for brevity when the context is clear, we interchangeably write $\mathcal{C}_j : \mathcal{X} \times \mathcal{X} \times \Theta_{\mathcal{C}_j} \to \mathrm{T}$ and imply the conditions $\mathbb{1}_{\{x'\}}$ and $\mathbb{1}_{\{x''\}}$.

Algorithm 1: MOSAIC

Input: Start state $x_{\text{start}} \in \mathcal{X}$
Goal termination condition function $\xi_{goal} : \mathcal{X} \rightarrow \{0, 1\}$
Skill library $\Sigma = \{\sigma\}_{m=1}^{M}$
Oracle O for skill and trajectory selection
Output: Sequence of skills and their parameters Π

```
1  M = DirectedMultigraph()
2  G ← {G_i ∈ Σ}                              // Generators container
3  C ← {C_j ∈ Σ}                              // Connectors container
4  while M.NODES = ∅ do
5      for σ ∈ G do
6          θ ← O.SampleParameters(σ)
7          τ ← σ(θ) // Estimatd, valid trajectories
8          if τ ≠ ∅ then
9              M.AddNode((σ, θ, τ), Cost(τ))
10         end
11     end
12 end
13 τ_start ← {x_start}
14 M.AddNode((∅, ∅, τ_start), 0) ; // Add start state as a node,
       with zero cost
15 while ¬M.HasPath(x_start, {x | ξ_goal(x) = 1}) do
       // Oracle selects next skill to apply
16     σ ← O.ChooseSkill(Σ, M)
17     θ ← O.SampleParameters(σ)
18     if σ ∈ C then
           // Apply connector skill
19         ξ_0, ξ_1 ← O.ChooseCondsToConnect(M)
20         τ ← σ(ξ_0, ξ_1, θ)
21         if τ ≠ ∅ then
22             M.AddEdge((σ, ξ_0, ξ_1, θ), Cost(τ))
23         end
24     else
           // Apply generator skill
25         τ ← σ(θ)
26         if τ ≠ ∅ then
27             M.AddNode((σ, θ, τ), Cost(τ))
28         end
29     end
30 end
   // Least-cost path to any goal state
31 return M.ShortestPath(x_start, {x | ξ_goal(x) = 1})
```

naively by alternating between generator and connector skills can result in excessive computational overhead (Sec. V-D), the MOSAIC algorithm employs a guidance module, called the *oracle*, which orchestrates the construction process by selecting appropriate skills and determining which parts of the mosaic graph to connect. In this section, we first outline the algorithm and then provide an overview of the oracle.

A. Algorithmic Approach

The MOSAIC algorithm (Alg. 1) finds a sequence of skills whose application to the given start state x_{start} results in a state satisfying the goal condition ξ_{goal}. MOSAIC does so by exploring the search space under the guidance of an oracle O, using a skill library Σ composed of skills σ that act as connectors ($\sigma \in \mathcal{C}$), generators ($\sigma \in \mathcal{G}$), or both.

The algorithm starts by initializing the graph (line 1) and invoking all available generators – adding all valid skills and their associated parameters as disconnected nodes in the mosaic graph (lines 4–12). If no nodes were added to the graph due to generator failures, the algorithm repeatedly queries the generators with different parameters until it finds at least one valid node.

The main loop of MOSAIC (lines 15–30) begins with the oracle selecting a skill to invoke next (line 16), which can be either a generator or a connector, and samples its parameters. If the selected skill is a connector, the oracle assigns it a start condition and a goal condition (line 19). Most often, the connector skill chooses two skill trajectories $\boldsymbol{\tau}_0$ and $\boldsymbol{\tau}_1$ from the mosaic and sets $\xi_0 := \mathbb{1}_{\{\boldsymbol{\tau}_0(1)\}}$ and $\xi_1 := \mathbb{1}_{\{\boldsymbol{\tau}_1(0)\}}$ to attempt connecting them. Otherwise, the connector will attempt to connect a chosen node to the goal condition by setting $\xi_1 := \xi_{\text{goal}}$. Next, the selected skill σ is invoked, and only the skills and their parameters that produce valid trajectories are added to the mosaic graph – as edges in the case of a connector (line 20) or as nodes in the case of a generator (line 25). If a skill may lead to more than one trajectory due to stochasticity of its corresponding policy, multiple trajectories are generated, e.g., via neural network batch inference or parallel computation on CPU cores, and MOSAIC leverages this information to estimate a confidence value (reflected in the cost value) associated with the selected skill by considering the fraction of invalid trajectories in the batch (lines 4–12, 18–24 and 25–29). Finally, if the mosaic graph contains a path between the start state and a state that satisfies the goal condition, the algorithm returns (line 31).

B. Oracle

MOSAIC explores the vast space of parameterized skills (i.e., expands its implicit multigraph) using its guidance module, which we term the *oracle*. The oracle's role is to orchestrate the search by balancing exploration–using *Generators* to discover new "islands of competence"–and exploitation–using *Connectors* to build edges between existing islands and find a solution. By making principled decisions at each step, the oracle directs the search toward robust, task-relevant skills.

The oracle within MOSAIC can be implemented in a variety of ways, and may be specialized with task-specific knowledge or leverage advancements in other fields (e.g., the oracle can be a large language model). Aiming to remain independent from specific tasks in this work, we propose an effective domain-independent statistical oracle that makes decisions based on the evolving structure of the search graph.

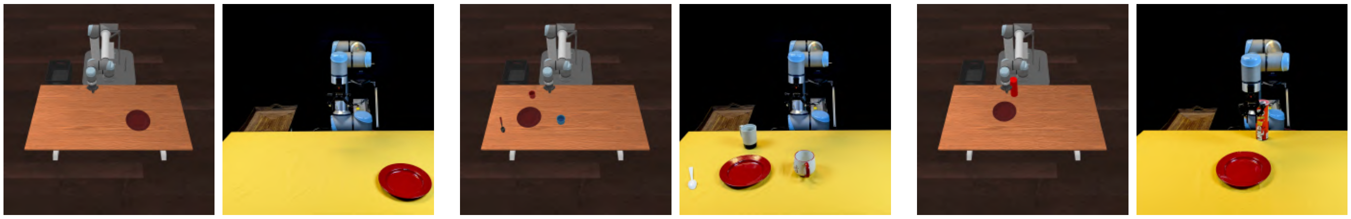
Our proposed oracle module is general and can be used with any library of skills. Specifically, when invoked, our oracle first decides whether to choose a Generator or a Connector based on the graph's connectivity; the probability of selecting a Connector increases as the ratio of nodes N to edges E in the graph grows. Once a skill type is chosen, a specific skill σ is selected by maximizing a score $U(\sigma_i)$ that balances its past performance with an exploration bonus:

$$U(\sigma_i) = \alpha s_{\sigma_i} + (1-\alpha)\sqrt{\ln \frac{\sum_j (t_{\sigma_j} + 1)}{t_{\sigma_i} + 1}} + n$$

with the s_σ being the success rate of skill σ, t_σ its invocation count, α a weighting parameter, and $n \sim \mathcal{N}(0, 1)$ a noise term for stochasticity.

When σ is a generator, it unconditionally creates a new node in the MOSAIC graph, and when it is a Connector, the oracle provides it with boundary conditions to connect. In the latter, the oracle may randomly select a node to connect locally to one of its neighbors[2] – promoting unbiased exploration, or exploit the structure of the graph by selecting

[2]We define a distance between nodes x and x' as the pose difference of robots and objects in the final state in x and initial state in x'.



(a) Transport. (b) Transport in Clutter. (c) Transport Among Movable Objects.

Fig. 2: Simulation setups and their real-world counterparts. Across all scenarios, the robot must place the plate into the bin. In Scenario 1: Transport (a), the robot must push the plate to the edge in order to pick it up. Scenario 2: Transport in Clutter (b) includes additional objects on the table. The robot must move the plate to the edge without displacing other objects. Scenario 3: Transport Among Movable Objects (c) allows the robot to interact with more objects on the table. The robot discovered the need to clear space for manipulating the plate by moving the chips can elsewhere.

nodes already reachable from the start state or the goal. To avoid redundant effort, the oracle penalizes pairs with repeated failed connection attempts by artificially increasing their distance, discouraging their selection in future nearest-neighbor searches. This design provides an effective, statistically-grounded guidance strategy that requires no task-specific tuning.

C. Theoretical Analysis: Probabilistic Completeness

In this section, we formally define probabilistic completeness (PC) in the context of skill-centric planning and provide a proof sketch demonstrating why MOSAIC is PC. For the purposes of these proofs, we assume deterministic skill-trajectory generation, i.e., each skill and parameter pair $\langle \sigma, \theta \rangle$ maps to a consistent trajectory (or a batch of trajectories). To define PC with respect to a given skill library, we first introduce the concept of a *feasible* solution:

Definition 3. *Given a start state x_{start}, a goal condition function $\xi_{goal} : \mathcal{X} \rightarrow \{0, 1\}$, and a library of skills $\Sigma = \{\sigma_m \in \mathcal{A}\}_{m=1}^{M}$, a solution trajectory is said to be **feasible** under Σ if it is decomposable into a sequence of trajectory segments, $\Pi = \{\boldsymbol{\tau}_1, \dots, \boldsymbol{\tau}_N\}$ such that $\forall \boldsymbol{\tau}_i \in \Pi, \exists \sigma_i \in \Sigma$ with parameters $\theta \in \Theta_{\sigma_i}$ that can generate $\boldsymbol{\tau}_i$.*

Given the definition of feasibility, a PC algorithm must guarantee that, asymptotically, it will discover a sequence of trajectories that compose a feasible solution if one exists:

Definition 4. *Let $\Pi_k^{\boldsymbol{ALG}}$ be the set of trajectories discovered by a skill-centric planning algorithm **ALG** at its k-th iteration. Additionally, let Π^* be the set of all feasible solutions for the planning problem. We call an algorithm **probabilistically complete** if $\lim_{k\to\infty} P(\exists \Pi \in \Pi_k^{ALG} \mid \Pi \in \Pi^*) = 1$.*

To establish PC, MOSAIC must satisfy two conditions. As the number of iterations $k \rightarrow \infty$, it must (1) invoke all generator skills ($\mathcal{G} \in \Sigma$) with every parameter configuration to explore all potential trajectory segments, and (2) attempt to connect every disconnected node using all connector skills ($\mathcal{C} \in \Sigma$) with all parameters. This is ensured by the fact that the oracle assigns non-zero probabilities to selecting any skill and sampling parameters randomly. The oracle's sampling strategy guarantees that, given infinite time, all possible skill combinations and parameter configurations will be explored. Thus, all generators and connectors are eventually used across all configurations, guaranteeing that MOSAIC explores all trajectories and connections, and finds a solution if one exists.

V. EXPERIMENTAL ANALYSIS

We evaluate MOSAIC in diverse simulated environments and validate on real UR10e robot, using RealSense for RGB-D pose estimation. We test three manipulation scenarios of increasing complexity, providing only the goal condition ("plate in bin") and a set of generic skills. Figure 2 depicts the setups; the accompanying video shows the hardware executions.

Scenario 1: Basic Transport places a plate deep on the table where it cannot be grasped directly due to its geometry. The planner must discover a sequence that first reorients the plate (e.g., by sliding it outward), then grasps it from the side, and places it in the bin. *Scenario 2: Transport in Clutter* extends Scenario 1 by adding static obstacles around the plate. The planner must discover a skill sequence that navigates through clutter without collisions. *Scenario 3: Transport Among Movable Objects* introduces a second movable object that may obstruct plate access. The planner can choose to work around this object or manipulate it, expanding the search space to include two-object interactions.

A. Skill Library

We tested MOSAIC with a library of four skills that can serve as generators ($\mathcal{G}$), connectors ($\mathcal{C}$), or both: **Push** ($\mathcal{G}+\mathcal{C}$) uses a learned diffusion policy [27] to generate pushing motions up to 25cm with 70% success rate. As a generator, it creates new world states with local push trajectories; as a connector, it moves objects between poses across world states. This skill uses a motion planner as a subroutine to arrive at pre-push configurations. **Pick** ($\mathcal{G}$) computes grasp poses based on object geometry and arm kinematics, then executes screw-based approach-grasp-retract trajectories. **Transport** ($\mathcal{C}$) moves grasped objects to satisfy goal conditions using motion planning [43, 44]. The skill fails if the object is not grasped or the goal isn't met. **Rearrange** ($\mathcal{C}$, Scenario 3 only) combines pick-and-place and push policies to reposition multiple objects by determining the appropriate policy for each object. During planning, the skills are rolled-out in the Sapien physics engine [3, 45] to verify

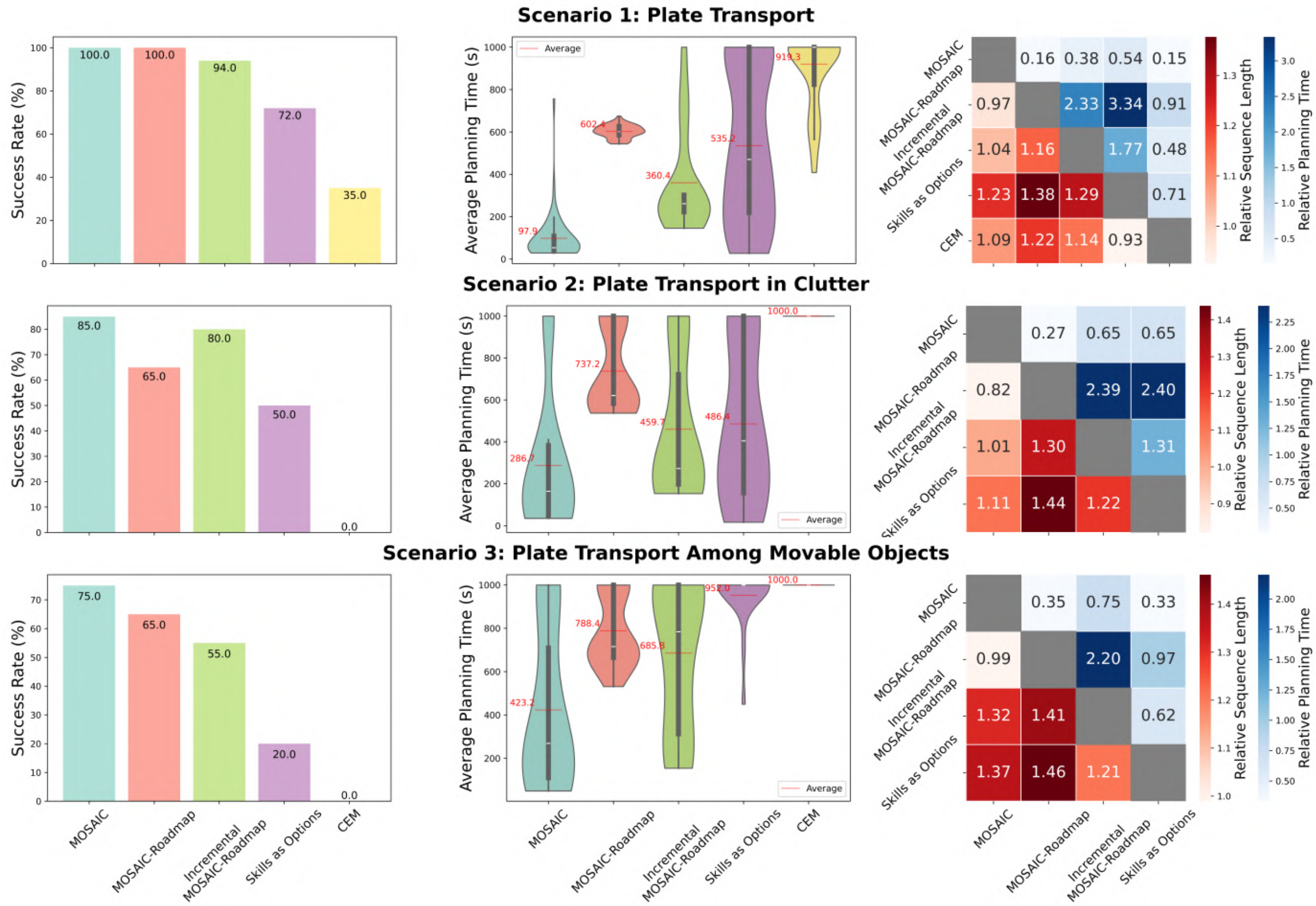


Fig. 3: Algorithms comparison across experimental scenarios. **Left**: Success rates. **Middle**: Planning time density with median, IQR, and average. **Right**: Head-to-head comparison on tests both algorithms solved. Upper-right shows relative planning times; lower-left shows relative sequence lengths. Each cell compares the "row algorithm" to the "column algorithm." For MOSAIC, lower values are better in the first row, and higher values are better in the first column.

contact outcomes and prune infeasible plans. Additional implementation details appear in the Appendix (Sec.**??**).

B. Evaluation Metrics

We measure algorithm performance primarily by *success rate*, defined as the fraction of trials where the planner reaches a goal-satisfying state ξ_{goal}. Additionally, we record the *plan length*, measured as the number of skills comprising the solution, serving as a proxy for task complexity and solution quality. Lastly, we measure each algorithm's *planning time* to evaluate its efficiency.

C. Baseline Algorithms

We compare MOSAIC to four long-horizon planners that compose skill primitives: **Skills as Options** implements sequential skill chaining [31], exploring skills incrementally with each new skill starting where the previous ended. It performs breadth-first search over world states by creating an action space of start-conditioned generator skills. **CEM** [46] uses receding horizon planning, sampling skill-parameter sequences, selecting top candidates, and refining the sampling distribution. It executes the first skill of the best sequence, updates the world state, and replans until success or timeout. **MOSAIC-Roadmap** removes sequential dependencies by generating skill trajectories freely and connecting them via a PRM-inspired roadmap [47]. The algorithm first builds a roadmap using both generator and connector skills, then attempts to connect the start and goal nodes. If successful, it uses Dijkstra's algorithm [48] to find a path; otherwise, it reports failure. **Incremental MOSAIC-Roadmap** extends MOSAIC-Roadmap by iteratively adding nodes and edges to the roadmap until start and goal nodes are connected, ensuring continuous progress toward a solution even if the initial roadmap doesn't yeild a path.

D. Experimental Results

Our experiments address three key questions: (1) Can MOSAIC effectively solve long-horizon skill-centric planning problems? (2) How important is the oracle module in MOSAIC's performance? (3) How does MOSAIC's skill discovery and composition strategy compare to traditional directed approaches? The results, shown in Fig. 3, demonstrate that MOSAIC consistently achieves high success rates across all scenarios while maintaining competitive solution quality and planning times. The performance of the baselines reveals important insights about different planning approaches: *Skills as*

Options, representing current sequential methods, performs well on simple tasks but degrades significantly with complexity due to its directional exploration strategy and poor scalability. *CEM* faces similar challenges, further limited by local optimization and lack of backtracking. While MOSAIC-*Roadmap* and *Incremental* MOSAIC-*Roadmap* show benefits of flexible exploration by removing sequential dependencies, their varying success rates across scenarios highlight the limitations of fixed strategies. These results demonstrate that MOSAIC's adaptive oracle-guided exploration enables more robust and efficient planning by dynamically adjusting its search strategy, enabling it to outperform baselines in complex scenarios. Our hardware validation tests (Fig. 2) trended similarly; View videos in our supplementary materials.

VI. CONCLUSION

We present MOSAIC, a skill-centric framework for solving long-horizon manipulation tasks through the composition of generic, imperfect skills, and physics simulation. Our work makes three key contributions: First, we introduce a different approach to skill-based planning, where skills actively guide the planning process toward regions where they are likely to succeed, rather than being composed via strictly sequential, either forward or backward, chaining. Second, we demonstrate that MOSAIC's modular architecture, consisting of *generators* for producing local trajectories and *connectors* for linking them through boundary value problems, with the use of physics simulation, allows us to avoid relying on hand-coded definitions of skill preconditions and effects. Third, through experiments across multiple tasks, we show that MOSAIC consistently achieves high success rates while reducing planning time compared to existing approaches, demonstrating that its adaptive oracle-guided exploration represents a fundamental advance in robust long-horizon manipulation planning. Looking forward, promising research directions include developing more sophisticated oracle modules, integrating foundation models for high-level reasoning, extending the framework to handle partial observability, and better utilizing the parallelization capabilities of physics simulators. We believe MOSAIC offers a new paradigm for skill-centric planning that will advance progress toward more capable, adaptable robotic manipulation in unstructured environments.

REFERENCES

[1] NVIDIA, "Isaac Sim."

[2] G. Authors, "Genesis: A universal and generative physics engine for robotics and beyond," 2024.

[3] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas, and H. Su, "SAPIEN: A simulated part-based interactive environment," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[4] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[5] K. Zakka, B. Tabanpour, Q. Liao, M. Haiderbhai, S. Holt, J. Y. Luo, A. Allshire, E. Frey, K. Sreenath, L. A. Kahrs *et al.*, "Mujoco playground," *arXiv preprint arXiv:2502.08844*, 2025.

[6] M. Torne, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal, "Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation," *arXiv preprint arXiv:2403.03949*, 2024.

[7] M. S. Saleem and M. Likhachev, "Planning with selective physics-based simulation for manipulation among movable objects," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6752–6758.

[8] D. M. Saxena, M. S. Saleem, and M. Likhachev, "Manipulation planning among movable obstacles using physics-based adaptive motion primitives," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 6570–6576.

[9] D. Saxena and M. Likhachev, "Planning for manipulation among movable objects: Deciding which objects go where, in what order, and how," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 33, 2023, pp. 668–676.

[10] R. Natarajan, G. L. Johnston, N. Simaan, M. Likhachev, and H. Choset, "Long-horizon torque-limited planning through contact using discrete search and continuous optimization," in *IROS 2023 Workshop on Leveraging Models for Contact-Rich Manipulation*, 2023.

[11] A. T. Le, K. Nguyen, M. N. Vu, J. Carvalho, and J. Peters, "Model tensor planning," *arXiv preprint arXiv:2505.01059*, 2025.

[12] W. Shen, C. Garrett, N. Kumar, A. Goyal, T. Hermans, L. P. Kaelbling, T. Lozano-Pérez, and F. Ramos, "Differentiable gpu-parallelized task and motion planning," 2025.

[13] C. Li, A. Krause, and M. Hutter, "Robotic world model: A neural network simulator for robust policy optimization in robotics," *arXiv preprint arXiv:2501.10100*, 2025.

[14] M. Assran, A. Bardes, D. Fan, Q. Garrido, R. Howes, M. Muckley, A. Rizvi, C. Roberts, K. Sinha, A. Zholus *et al.*, "V-jepa 2: Self-supervised video models enable understanding, prediction and planning," *arXiv preprint arXiv:2506.09985*, 2025.

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

[16] M. Crosby, F. Rovida, M. R. Pedersen, R. P. Petrick, and V. Krüger, "Planning for robots with skills," in *4th ICAPS Workshop on Planning and Robotics*, 2016.

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

[18] J. Mao, T. Lozano-Pérez, J. Tenenbaum, and L. Kaelbling, "Pdsketch: Integrated domain programming, learning, and planning," *Advances in Neural Information Processing Systems*, 2022.

[19] B. Kim, L. Kaelbling, and T. Lozano-Pérez, "Guiding search in continuous state-action spaces by learning

an action sampler from off-target search experience," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[20] Z. Wang, C. R. Garrett, L. P. Kaelbling, and T. Lozano-Pérez, "Learning compositional models of robot skills for task and motion planning," *The International Journal of Robotics Research*, 2021.

[21] T. Migimatsu, W. Lian, J. Bohg, and S. Schaal, "Symbolic state estimation with predicates for contact-rich manipulation tasks," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022.

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

[23] J. Liang, M. Sharma, A. LaGrassa, S. Vats, S. Saxena, and O. Kroemer, "Search-based task planning with learned skill effect models for lifelong robotic manipulation," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 6351–6357.

[24] X. Fang, C. R. Garrett, C. Eppner, T. Lozano-Pérez, L. P. Kaelbling, and D. Fox, "Dimsam: Diffusion models as samplers for task and motion planning under partial observability," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 1412–1419.

[25] Z. Zhou, A. Garg, D. Fox, C. Garrett, and A. Mandlekar, "Spire: Synergistic planning, imitation, and reinforcement learning for long-horizon manipulation," *arXiv preprint arXiv:2410.18065*, 2024.

[26] M. Janner, Y. Du, J. Tenenbaum, and S. Levine, "Planning with diffusion for flexible behavior synthesis," in *International Conference on Machine Learning*, 2022.

[27] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, 2024.

[28] C. Florensa, D. Held, X. Geng, and P. Abbeel, "Automatic goal generation for reinforcement learning agents," in *International conference on machine learning*, 2018, pp. 1515–1528.

[29] S. Forestier, R. Portelas, Y. Mollard, and P.-Y. Oudeyer, "Intrinsically motivated goal exploration processes with automatic curriculum learning," *Journal of Machine Learning Research*, vol. 23, no. 152, pp. 1–41, 2022.

[30] V. Veeriah, T. Zahavy, M. Hessel, Z. Xu, J. Oh, I. Kemaev, H. P. van Hasselt, D. Silver, and S. Singh, "Discovery of options via meta-learned subgoals," *Advances in Neural Information Processing Systems*, vol. 34, pp. 29 861–29 873, 2021.

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

[32] R. S. Sutton, D. Precup, and S. Singh, "between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, 1999.

[33] A. Bagaria and G. Konidaris, "Option discovery using deep skill chaining," in *International Conference on Learning Representations*, 2019.

[34] A. Bagaria, J. Senthil, M. Slivinski, and G. Konidaris, "Robustly learning composable options in deep reinforcement learning," in *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, 2021.

[35] A. Bagaria, J. K. Senthil, and G. Konidaris, "Skill discovery for exploration and planning using deep skill graphs," in *International Conference on Machine Learning*, 2021, pp. 521–531.

[36] D. Xu, A. Mandlekar, R. Martín-Martín, Y. Zhu, S. Savarese, and L. Fei-Fei, "Deep affordance foresight: Planning through what can be done in the future," in *2021 IEEE international conference on robotics and automation (ICRA)*, 2021, pp. 6206–6213.

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

[38] U. A. Mishra, S. Xue, Y. Chen, and D. Xu, "Generative skill chaining: Long-horizon skill planning with diffusion models," in *Conference on Robot Learning*, 2023.

[39] U. A. Mishra, Y. Chen, and D. Xu, "Generative factor chaining: Coordinated manipulation with diffusion-based factor graph," in *8th Annual Conference on Robot Learning*, 2024.

[40] A. Sivaramakrishnan, S. Tangirala, E. Granados, N. R. Carver, and K. E. Bekris, "Roadmaps with gaps over controllers: Achieving efficiency in planning under dynamics," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.

[41] K. Lu, A. Grover, P. Abbeel, and I. Mordatch, "Reset-free lifelong learning with skill-space planning," *arXiv preprint arXiv:2012.03548*, 2020.

[42] B. Ames, A. Thackston, and G. Konidaris, "Learning symbolic representations for planning with parameterized skills," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[43] J. Kuffner and S. LaValle, "Rrt-connect: An efficient approach to single-query path planning," *IEEE International Conference on Robotics and Automation*, 2000.

[44] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics and Automation Magazine*, pp. 72–82, 2012.

[45] J. Gu, F. Xiang, X. Li, Z. Ling, X. Liu, T. Mu, Y. Tang, S. Tao, X. Wei, Y. Yao, X. Yuan, P. Xie, Z. Huang, R. Chen, and H. Su, "Maniskill2: A unified benchmark for generalizable manipulation skills," in *International Conference on Learning Representations*, 2023.

[46] R. Y. Rubinstein and D. P. Kroese, *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer Science & Business Media, 2004.

[47] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, pp. 566–580, 1996.

[48] E. DIJKSTRA, "A note on two problems in connexion with graphs," *Numerische Mathematik*, 1959.