IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.0429000

LLM-Guided Tool-Aware Task and Motion Planning for Chemistry Lab Automation

BOHYEONG PARK[1], HYUNHO CHO[1], and SANGHYUN KIM[1,2], (Member, IEEE)

[1]Department of Mechanical Engineering (AgeTech-Service Convergence Major), Kyung Hee University, 1732 Deogyeong-daero, Giheung-gu, Yongin-si 17104, Gyeonggi-do, Republic of Korea (e-mail: bhe1004@khu.ac.kr; chohh7391@khu.ac.kr)
[2]Advanced Institute of Convergence Technology, 145 Gwanggyo-ro, Yeongtong-gu, Suwon-si 16229, Gyeonggi-do, Republic of Korea

Corresponding author: Sanghyun Kim (e-mail: kim87@khu.ac.kr).

This research was supported by the SMEs Technology Innovation Development Program through the Technology Innovation and Promotion Agency (TIPA), funded by the Ministry of SMEs and Startups (RS-2024-00511332); the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (RS-2024-00411007); and a Korea Basic Science Institute (National Research Facilities and Equipment Center) grant funded by the Ministry of Science and ICT (RS-2025-00564593).
Bohyeong Park and Hyunho Cho contributed equally to this work.

ABSTRACT Autonomous chemistry laboratories can accelerate material discovery and improve experimental reproducibility. However, current systems are limited by rigid tool configurations and unreliable motion planning under orientation and collision constraints. Moreover, existing LLM-based planners remain confined to protocol generation without reasoning about the execution context required at each procedural stage. This paper proposes LLM-Guided Tool-Aware TAMP (Task and Motion Planning), a hierarchical framework that unifies large language model (LLM)-based symbolic reasoning with batched joint differentiable optimization-based planning for chemistry automation. The system converts natural-language experimental instructions into validated Chemical Description Language (XDL) protocols, infers appropriate end-effectors and obstacle rearrangement strategies from procedural context, estimates the scene state through multi-view fiducial marker-based perception, and produces collision-free, constraint-satisfying trajectories by evaluating candidates in parallel. A sensor-driven skill library handles execution-level actions such as tool changing and adaptive pouring. On a 6-DoF manipulator, the proposed planner achieves a 100.0% planning success rate under orientation constraints across three tasks, improving over a sequential parameter-binding baseline from an average of 70.0%, while also reducing planning-time variance. We build a high-fidelity simulation testbed in *NVIDIA Isaac Sim* for randomized trials and module-level ablation studies, and results confirm that the full framework can execute complete chemical sequences involving tool switching, obstacle rearrangement, and multi-step reagent manipulation. The project page is available at https://rcilab.github.io/auto_chem.

INDEX TERMS Autonomous chemistry laboratory, intelligent planning, large language models, task and motion planning

I. INTRODUCTION

CHEMISTRY experiments are the cornerstone of material science discovery and hypothesis verification. Traditionally, these experiments have been conducted manually, a process that is resource-intensive and often limited by human variability in reproducibility. To address these limitations, Self-Driving Labs (SDLs) have emerged as a promising paradigm that automates the experimental cycle, from experiment selection and robotic execution to data acquisition [1], [2]. Despite this progress, existing laboratory automation pipelines are built around fixed tool configurations and protocol-specific routines, requiring manual redesign whenever new apparatus or procedures are introduced. This rigidity does not scale as chemical workflows grow increasingly diverse and non-linear, motivating a system that can autonomously adapt its tool-use configuration without manual domain redefinition.

Beyond operational flexibility, reliable physical execution remains difficult in laboratory environments. To address this challenge, constrained TAMP (Task and Motion Planning) [3], [4] has been applied to automate multi-step procedures such as pouring, stirring, and reagent transport. For example, Yoshikawa et al. [5] utilized PDDLStream [6] and projection-based sampling [7] to prevent reagent spillage under orienta-

tion constraints; however, their work reported that such task-specific constraints significantly degrade planning success rates on standard robot arms, requiring additional degrees of freedom to recover acceptable performance. This limitation is structural in approaches that independently sample each parameter and compose them through rejection: as the feasible region of the constraint manifold shrinks with decreasing kinematic redundancy, the probability that independently drawn samples jointly satisfy all constraints simultaneously decreases. For 6-DoF manipulators commonly used in collaborative robot environments, this structural dependence results in degraded planning success rates under orientation constraints and unpredictable termination time, rendering such approaches unsuitable for time-sensitive chemical workflows.

To simultaneously address the challenges of context-aware tool selection and constrained planning reliability, this paper proposes the LLM-Guided Tool-Aware TAMP framework. A lightweight large language model (LLM) serves as a context-aware decision engine that extends beyond converting natural-language instructions into executable protocols. It determines the execution context required at each procedural stage by inferring the appropriate end-effector from object geometry and task requirements, and by deciding whether and where workspace obstacles should be rearranged through joint consideration of the current and subsequent procedures. These decisions are resolved before planning, allowing the TAMP solver to start from an updated configuration and focus on computing reliable, collision-free trajectories. To this end, the framework adopts cuTAMP [8], a differentiable TAMP solver that jointly optimizes batched candidate solutions, as its planning backbone, achieving both high planning success rates and structurally bounded planning time under tight orientation constraints on 6-DoF manipulators.

To support this execution model, the framework maintains environment state consistency through a robust state grounding interface that fuses multi-view *AprilTag*-based tracking with continuous sensor telemetry, providing the planning engine with consistent world-state estimates. Execution-level actions such as tool changing and adaptive pouring are handled by a set of sensor-feedback-driven skills that reliably bridge planned trajectories to physical execution. The main contributions of this paper are as follows:

- A hierarchical decision-making pipeline based on a lightweight LLM is established for context-aware chemistry automation. The pipeline converts natural language instructions into validated Chemical Description Language (XDL) protocols, infers the appropriate end-effector from object geometry and task requirements, and determines obstacle rearrangement strategies by jointly analyzing current and subsequent procedural context, enabling dynamic tool switching (e.g., suction, multi-finger grippers) without manual domain redefinition.
- cuTAMP is adopted as the planning backbone for constrained 6-DoF laboratory manipulation, jointly optimizing batched candidate solutions through differentiable gradient descent to produce constraint-satisfying trajectories within a structurally bounded time budget. This simultaneously addresses the degradation in planning success under tight orientation constraints and the unbounded termination time that makes independent parameter sampling unsuitable for time-sensitive chemical workflows.
- These components are unified into a single end-to-end pipeline that enables fully autonomous execution of complex chemical workflows from a single natural-language command, demonstrating that operational flexibility and planning reliability can be addressed jointly rather than in isolation. The integrated framework is validated under randomized initial conditions in a high-fidelity simulation testbed built on *NVIDIA Isaac Sim*, covering module-level ablations and end-to-end performance on long-horizon workflows involving tool switching, obstacle rearrangement, and reagent handling.

The remainder of this article is organized as follows: Section II reviews related work in autonomous chemistry, constrained TAMP, and LLM-based planning, while defining the position of this work. Section III details the proposed framework architecture and its core components. Section IV presents the experimental results and performance evaluations conducted in a purpose-built simulation testbed based on *NVIDIA Isaac Sim*. Section V discusses the assumptions and limitations that bound the interpretation of these results. Finally, Section VI offers concluding remarks and discusses future research directions.

II. RELATED WORK

A. AUTONOMOUS CHEMISTRY AND SELF-DRIVING LABORATORIES

Self-Driving Labs (SDLs) have emerged as a core technology for accelerating material discovery and ensuring experimental reproducibility. While early SDL research focused on automation tailored to specific experiments, the paradigm has recently shifted toward building flexible laboratories using general-purpose robots. Burger et al. [9] demonstrated autonomous photocatalyst optimization by introducing a mobile robot into the laboratory, and Fakhruldeen et al. [10] established a modular integration standard for heterogeneous robotic platforms through the ARChemist architecture. Lunt et al. [11] further extended this direction by integrating multiple robots to automate solid-state chemistry workflows.

Despite these advances, existing SDL systems share a common structural limitation: they are largely built around protocol-specific equipment and manually designed execution routines, requiring substantial reconfiguration when new procedures or apparatus are introduced. This rigidity extends to the manipulation layer as well. Kennedy et al. [12] and Huang et al. [13] developed sensor-feedback-based pouring skills, and Yoshikawa et al. [14] automated electrode polishing, yet these efforts assume fixed end-effectors

tailored to predefined tasks, and incorporating a new tool typically requires manual domain redefinition rather than autonomous adaptation. Recent efforts have also sought to improve the reliability of individual laboratory instruments through perception-based feedback. Khan et al. [15] integrated a YOLOv8-based computer vision model with the Opentrons OT-2 liquid handling robot to enable real-time detection of pipette tips and liquid volumes, demonstrating that closed-loop quality control can meaningfully improve accuracy in liquid handling tasks. However, this approach remains scoped to a single, purpose-built instrument and a fixed operational domain, and does not address the broader challenge of adapting tool-use configurations across diverse procedural contexts. In summary, while automation of individual experimental procedures has progressed considerably, adaptability to diverse and evolving chemical workflows remains limited when the set of required tools, vessel geometries, and procedural sequences is not known in advance.

B. TASK AND MOTION PLANNING FOR CONSTRAINED LABORATORY MANIPULATION

Complex chemistry experiments simultaneously require long-horizon task planning and precise motion planning under strict physical constraints. PDDLStream, proposed by Garrett et al. [6], has become a representative framework for combining symbolic logic with sampling-based planning. Existing studies have applied PDDLStream to enforce geometric constraints such as orientation maintenance during liquid transfer; however, PDDLStream operates by first constructing an optimistic plan and then sequentially evaluating the stream instances that support it, attempting to bind each continuous parameter independently without accounting for its interaction with other constraints. As a result, as kinematic redundancy decreases and the feasible region satisfying all constraints simultaneously narrows, the probability of finding a valid binding decreases structurally. Furthermore, because the number of sampling iterations required to find a valid solution is inherently stochastic, the termination time of such planners is unpredictable and unbounded, which is a critical limitation for chemistry laboratory automation, where unexpected planning delays can disrupt time-sensitive reaction conditions such as temperature regulation and reagent timing.

Optimization-based motion planning by Muchacho et al. [16] and constraint-satisfying sampling techniques by Kingston et al. [17] have been investigated to improve constraint handling, yet they do not fundamentally resolve the reliability degradation on manipulators with limited degrees of freedom, nor do they guarantee bounded planning latency. Coleman et al. [18] proposed a design to lower the barrier to entry for complex robotic software through a *MoveIt!* case study, but the underlying challenges of planning reliability and latency predictability remain unaddressed. Hybrid approaches combining optimization-based pre-grasp planning with reinforcement learning have also been explored for dynamic environments. Li et al. [19] proposed a method that integrates quasi-Newton trajectory initialization with a recurrent deep RL policy for dynamic obstacle avoidance and grasping in mobile manipulators. However, learning-based planners of this kind do not provide formal guarantees on constraint satisfaction or planning latency, as convergence depends on stochastic policy training rather than deterministic optimization over a fixed computational budget. Regardless of the underlying approach, whether sampling-based, optimization-based, or learning-based, these methods have primarily been evaluated on general pick-and-place scenarios. Laboratory manipulation poses a compounded difficulty: orientation constraints must be maintained throughout liquid transport, obstacles may need to be rearranged between procedural steps depending on task context, and tool switching must be coordinated within multi-step workflows, all on low-redundancy robots where planning reliability is already degraded. This combination of requirements has received limited attention in existing TAMP literature.

C. LARGE LANGUAGE MODELS FOR ROBOT PLANNING AND EMBODIED DECISION-MAKING

The advent of Large Language Models (LLMs) has opened new possibilities for translating abstract natural language commands into robot-executable instructions. Mehr et al. [20] proposed the Chemical Description Language (XDL) standard for the digitization of chemical literature, and Yoshikawa et al. [21] built the CLAIRify system, which uses LLMs to convert chemical texts into robot commands. Darvish et al. (ORGANA) [22] presented an interactive robotic assistant that utilizes an LLM-based planner to converse with scientists and assist with experiments, and ChemCrow [23] and Coscientist [24] demonstrated that LLMs can autonomously search for chemical tools and design experimental procedures.

While these systems demonstrate strong capabilities in semantic decomposition and high-level task sequencing, their use of LLMs generally remains confined to text-based procedure generation or human decision-making assistance. The generated plans are typically handed off to a separate, manually configured execution pipeline, and the LLM itself is not required to reason about the physical execution environment, such as which end-effector is geometrically appropriate for a given vessel, or whether workspace obstacles must be rearranged before a procedure can proceed. As a consequence, geometric validity, collision avoidance, and orientation feasibility are not addressed at the reasoning stage and must be delegated entirely to downstream planning modules. Recent work has sought to close this gap through simulation-based feasibility verification. Qu et al. [25] proposed TRTP, a three-stage framework that couples vision-language models with digital twin simulation to evaluate spatial reachability and physical feasibility of LLM-generated action sequences before execution. While this closed-loop architecture improves plan reliability, the feasibility check is performed in a separate simulation module rather than being resolved at the symbolic reasoning stage; end-effector selection and context-dependent obstacle rearrangement remain outside the scope of the language model's decision-making. This separation

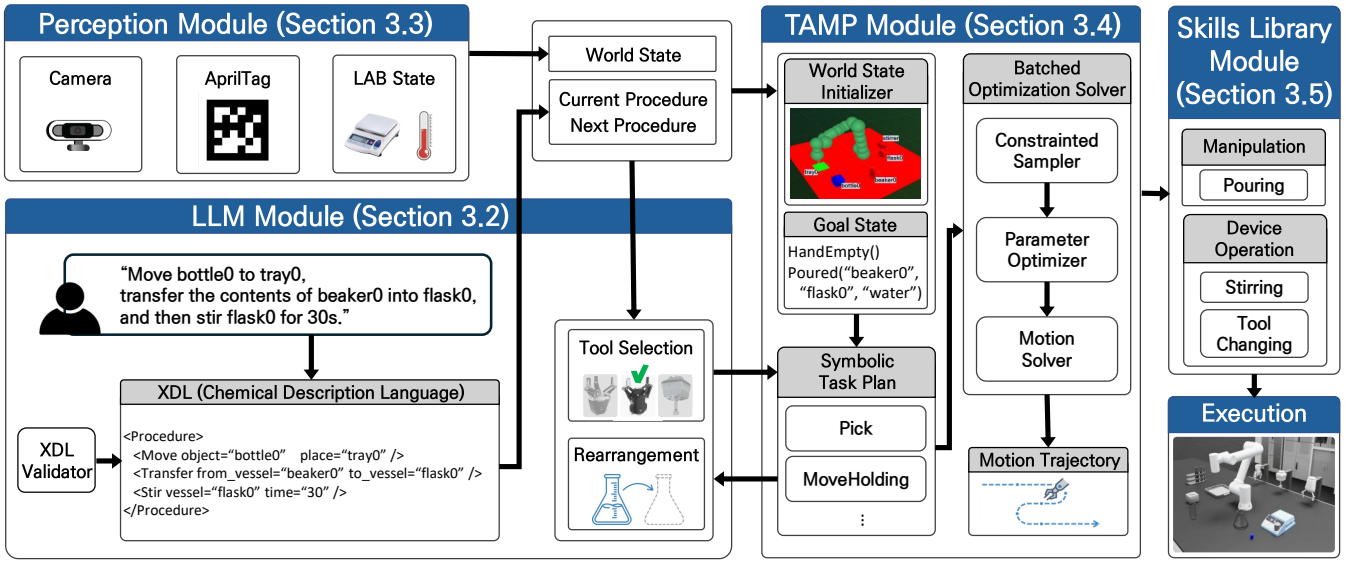


FIGURE 1. Architectural overview of the LLM-Guided Tool-Aware TAMP framework. The hierarchy illustrates the integration of symbolic reasoning (LLM Module) with physical execution (TAMP Module) supported by a multi-modal perception interface (Perception Module) and a robotic skill library (Skill Library Module).

means that existing LLM-based robot planning systems can formulate what to do, but lack the ability to determine the execution context required to carry it out, namely which tool is appropriate for the current task and whether the workspace must be reconfigured before execution proceeds.

D. POSITION OF THIS WORK

Existing work has addressed individual aspects of autonomous chemistry from complementary directions. SDL and chemistry automation systems have demonstrated automated experimental execution, but remain largely protocol-specific and assume fixed tool configurations that require manual redesign for new procedures. Constrained TAMP approaches have tackled motion planning under geometric constraints, but exhibit reliability degradation on low-redundancy manipulators and do not guarantee bounded and predictable planning latency for time-sensitive chemical workflows. LLM-based robot planning systems have enabled high-level semantic reasoning for chemical procedures, but remain confined to protocol generation without determining the execution context required to carry out each procedure, namely tool selection and obstacle rearrangement decisions.

Limited attention has been paid to jointly addressing context-aware symbolic reasoning for end-effector selection, obstacle rearrangement, and procedural sequencing together with reliable, bounded-time constrained execution on general-purpose laboratory manipulators. The proposed framework is positioned at the intersection of these three directions, where it unifies LLM-based context-aware decision-making with batched joint differentiable optimization-based constrained planning within a single end-to-end system for flexible and reliable chemistry laboratory automation.

III. METHODS

Section II identified three coupled limitations that prevent existing approaches from achieving flexible and reliable chemistry laboratory automation: existing automation pipelines require manual redesign whenever new apparatus or procedures are introduced; constrained motion planning on low-redundancy manipulators suffers from degraded reliability and unpredictable termination time under orientation constraints; and LLM-based robot planning systems remain confined to protocol generation without determining the execution context required to carry out each procedure, namely tool selection and obstacle rearrangement decisions. To address these limitations, the proposed framework comprises four mutually necessary modules: the LLM Module, TAMP Module, Skill Library Module, and Perception Module. The LLM Module jointly addresses the first and third limitations by converting natural-language instructions into validated XDL protocols and determining the execution context required at each procedural stage, including end-effector selection and obstacle rearrangement, without requiring manual domain redefinition. The TAMP Module addresses the second by adopting cuTAMP [8] as its planning backbone, jointly optimizing batched candidate solutions through differentiable gradient descent to produce constraint-satisfying trajectories within a structurally bounded time budget. The Skill Library Module complements the TAMP Module by handling execution-level actions that are not amenable to geometric planning alone, such as adaptive pouring and temperature regulation, ensuring robust physical execution under real-world uncertainty. The Perception Module underpins all three by continuously sensing the laboratory environment and constructing a World State, a structured representation of object poses, environment geometry, and apparatus configurations that conforms

to the symbolic and geometric world definition required by the TAMP Module. This representation is supplied to each module as required.

A. FRAMEWORK OVERVIEW

Fig. 1 illustrates the overall architecture and inter-module data flow of the proposed framework, in which the LLM Module, TAMP Module, and Skill Library Module operate sequentially on the basis of a shared World State continuously maintained by the Perception Module, which senses object poses and sensor data throughout the laboratory environment.

The pipeline begins with the LLM Module converting the natural-language command into an XDL protocol. Since a linguistically valid protocol may still be physically infeasible in the current workspace, the protocol is cross-checked against the current World State before being passed to the execution stage. Prior to each XDL procedure, the LLM Module infers the required end-effector, whether obstacle rearrangement is necessary, the auxiliary end-effector for rearrangement, and the target location for the relocated object by referencing the latest World State. At this stage, the LLM specifies only the target workspace grid for rearrangement, while precise geometric computation is delegated to the TAMP Module.

The TAMP Module receives these symbolic decisions, constructs a task plan, and computes a collision-free, constraint-satisfying nominal trajectory for each operator by searching for a plan skeleton and using the latest World State. However, since geometric planning alone cannot account for physical uncertainties that arise during execution, such as liquid flow and contact dynamics, the nominal trajectories are passed to the Skill Library Module. The Skill Library Module translates the nominal trajectories into physical execution through sensor-feedback-based motion primitives, correcting for uncertainties not modeled at the planning stage in real time.

Each module completes its responsibilities within a well-defined scope before passing control to the next, thereby preventing failures at one level from propagating to subsequent stages. Sections III-B through III-E describe the internal structure and design principles of each module in detail.

B. LLM MODULE

The LLM Module constitutes the symbolic reasoning layer of the proposed framework, serving as the interface through which natural-language instructions are systematically translated into executable protocols and context-dependent execution decisions are resolved prior to physical planning. Central to its design is the deliberate confinement of the LLM to discrete symbolic outputs, a constraint motivated by a fundamental limitation of current language models: continuous 3D spatial reasoning, such as deriving exact placement coordinates or collision-free waypoints, lies outside their reliable operating regime and introduces the risk of geometrically infeasible outputs propagating into downstream execution. Rather than exposing the model to such reasoning demands, the module restricts its scope to decisions that are inherently symbolic in nature, whose outputs belong to a finite, discrete set and require no continuous spatial computation: selecting an end-effector from a predefined inventory, determining whether obstacle rearrangement is necessary, and specifying a target workspace grid rather than a precise Cartesian location. The geometric realization of these abstract decisions, most notably exact placement within the designated workspace grid, is delegated to the TAMP Module, which operates at the continuous spatial level; this division of labor positions the LLM at an abstraction level commensurate with its pattern-matching capabilities, while systematically preventing hallucinated spatial quantities from entering the planning pipeline.

Organized around this design principle, the module proceeds through three sequential stages. The XDL Generator synthesizes a global experimental protocol from the input natural-language command, producing a structured symbolic representation of the intended procedure. The XDL Validator then performs three progressively stricter checks: syntactic validation, object-existence verification, and physical-feasibility assessment. By intercepting language-level inconsistencies at this stage, the validator ensures that only coherent and physically grounded protocols advance to execution. Finally, the Action Reasoner analyzes the current World State in conjunction with procedural context to infer tool selection and obstacle rearrangement decisions at each procedural step. The following subsections describe each of these stages in detail.

1) Global Experimental Protocol Synthesis

The XDL Generator functions as a schema-constrained protocol generation module that maps natural-language commands into complete, end-to-end structured XDL programs encoding the full experimental procedure in a single pass. This global synthesis approach is a direct instantiation of the discrete-output design principle: by confining the output space to a fixed XDL schema with a finite set of operators and parameter types, the generator is discouraged from producing continuous spatial quantities, substantially reducing the risk of geometrically infeasible outputs propagating into downstream execution. The discreteness of the schema also means that a lightweight *Llama 3.2 1B* model fine-tuned via Low-Rank Adaptation (LoRA)-based supervised fine-tuning (SFT) is sufficient to achieve reliable conversion without the overhead and latency of general-purpose large models. The training dataset consists of 5,000 manually constructed samples covering synonym variations for six core XDL operators, continuous numerical parameter ranges, and diverse object instances. Additional details on dataset construction and training are provided in Section IV-A3.

2) Protocol Validation

The XDL Validator serves as the safety interface between the language layer and the physical execution pipeline. While the XDL Generator produces structured protocols, a protocol that is syntactically valid may still be physically infeasible. The Validator therefore enforces three progressively stricter checks before any robot motion is executed. Syntactic valida-

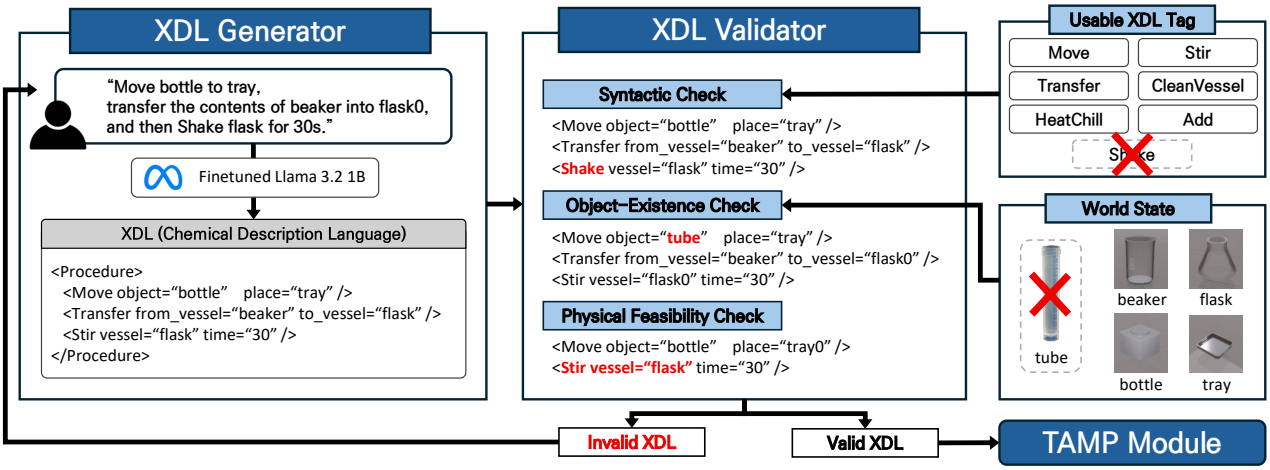


FIGURE 2. The XDL Generator and Validator pipeline within the LLM Module. A fine-tuned *Llama 3.2 1B* model converts a natural language command into a structured XDL protocol, which is then subjected to three progressively stricter checks (syntactic, object-existence, and physical-feasibility) before a valid protocol is forwarded to the TAMP Module.

tion verifies well-formed XML structure and permitted operator tags, blocking case-sensitivity mismatches and undefined elements. Object-existence validation cross-references all referenced objects against the workspace inventory provided by the Perception Module, intercepting commands that refer to objects absent from the current environment. Physical-feasibility validation simulates the content state of each vessel across the procedure and rejects sequences that violate procedural preconditions, such as stirring an empty vessel or transferring from an empty source. If a conflict is detected at any stage, execution is halted and the offending step and error type are reported to the user, enabling targeted protocol-level revision before physical failure can occur. Fig. 2 illustrates representative examples: syntactic validation rejects undefined operator tags such as Shake, object-existence validation intercepts references to absent objects such as tube, and physical-feasibility validation rejects operations such as stirring an empty vessel.

3) Context-Aware Tool and Action Reasoning

The Action Reasoner determines the end-effector and obstacle rearrangement strategy for each procedure before the TAMP Module begins planning. This stage is critical for reducing planning failures caused by tool mismatches or obstructed workspaces. Following the symbolic-level design principle established in Section III-B, the module operates on the discretized workspace representation illustrated in Fig. 3, where the environment is partitioned into 12 workspace grids (G1–G12) based on the spatial state provided by the Perception Module. A fine-tuned *Llama 3.2 1B* model receives four inputs: the current XDL procedure, the subsequent procedure, the current obstacle layout, and the workspace grid state. From these, it infers four discrete decision variables: the required end-effector for the current operation, whether preliminary object rearrangement is necessary, the auxiliary tool for rearrangement, and the target workspace grid for the relocated object.

This design provides two main advantages. First, it enables semantic tool assignment by grounding the model's knowledge of object affordances in the physical constraints of the available end-effectors. Consequently, appropriate grippers can be assigned even for vessel types unseen during training, minimizing the need for manual domain redefinition. Second, it enables context-aware rearrangement by jointly considering the current and subsequent procedures, allowing obstacles to be moved proactively toward regions required for future steps and thereby reducing redundant rearrangements. Because auxiliary rearrangement does not demand high positional precision, the Action Reasoner predicts only the target workspace grid, while exact placement within that region is delegated to the TAMP Module. This clear division of labor ensures that the LLM handles high-level strategy while the TAMP Module ensures geometric and kinematic feasibility. For example, as illustrated in Fig. 3, when the flask obstructs access to the beaker in the current procedure, and the next procedure is a Stir operation targeting the flask, the Action Reasoner identifies the flask as an obstacle requiring rearrangement and infers G2 as the target grid, since the stirrer is located there.

C. PERCEPTION MODULE

The Perception Module serves as the primary interface for world-state grounding, supplying all downstream modules with a consistent and up-to-date representation of the laboratory environment. Its primary role is to provide the TAMP Module with real-time 6-DoF pose information of laboratory assets for collision-free motion planning, but its outputs are consumed more broadly across the framework: the LLM Module's Action Reasoner relies on the workspace grid state for context-aware tool and rearrangement decisions, and the Skill Library Module draws on continuous sensor streams for closed-loop execution of skills such as adaptive pouring and

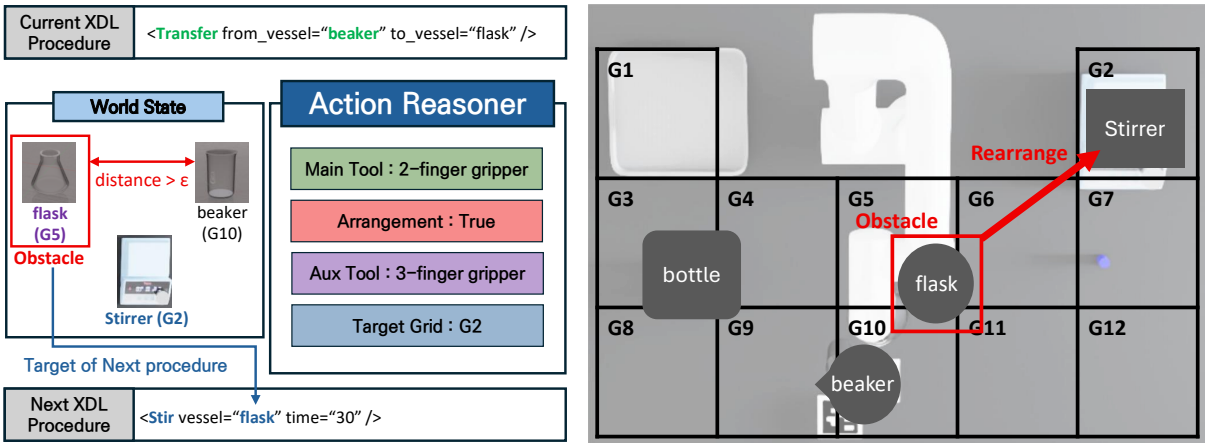


FIGURE 3. Illustration of the Action Reasoner pipeline. Given the current and next XDL procedures alongside the World State, a fine-tuned *Llama 3.2 1B* model infers four discrete outputs: the main end-effector, whether obstacle rearrangement is necessary, the auxiliary tool for rearrangement, and the target workspace grid within the discretized workspace (G1–G12).

temperature regulation.

Precise and reliable localization of apparatus is critical for ensuring safe and collision-free manipulation in an autonomous chemistry environment. The module adopts an *AprilTag*-based strategy [26] to achieve robust pose estimation at low computational latency, as illustrated in Fig. 4. Markers are rigidly affixed to each vessel, and their poses are continuously estimated from two spatially diverse, calibrated RGB cameras mounted around the workspace. The following subsections detail the localization pipeline, world-state construction, and sensor stream processing that together constitute the module's output.

1) Lab Object Localization

Each lab object is assigned a unique identifier, and its localization proceeds through a three-stage pipeline: individual pose computation in the robot base frame, dual-camera fusion for robustness against occlusions, and tag-to-vessel coordinate transformation.

Single-Camera Pose Computation: For each camera $c \in \{1, 2\}$, the pose of object i in the robot base frame is obtained by chaining the fixed extrinsic calibration ${}^{\mathrm{b}}T_c$ with the detected marker pose ${}^{c}T_{\mathrm{t}}^{(i)}$. To assess observation reliability, the Euclidean distance $d_c^{(i)}$ from the camera optical center to the marker is recorded as a weight metric:

$$ {}^{\mathrm{b}}T_{\mathrm{t},c}^{(i)} = {}^{\mathrm{b}}T_c \cdot {}^{c}T_{\mathrm{t}}^{(i)}, \quad d_c^{(i)} = \left\| \mathbf{t}\left({}^{c}T_{\mathrm{t}}^{(i)}\right) \right\|_2. \tag{1} $$

Inverse-Distance Weighted Fusion: When both cameras yield valid observations, the estimates are fused by assigning higher confidence to the camera closer to the marker, thereby minimizing distance-dependent projection errors. Given a regularization constant $\varepsilon = 10^{-6}$ to prevent division by zero, the normalized weights W_c and the resulting fused translation $\tilde{\mathbf{t}}$ and orientation $\tilde{\mathbf{q}}$ via Spherical Linear Interpolation (*SLERP*) [27] are computed as follows:

$$ W_c = \frac{(d_c^{(i)})^{-2}}{\sum_{j=1}^{2}(d_j^{(i)})^{-2} + \varepsilon}, \qquad \tilde{\mathbf{t}} = W_1\mathbf{t}_1 + W_2\mathbf{t}_2, \quad \tilde{\mathbf{q}} = SLERP(\mathbf{q}_1, \mathbf{q}_2, W_2). \tag{2} $$

If occlusion occurs and only one camera provides a valid observation, its estimate is used directly without fusion to maintain continuity of pose tracking.

Tag-to-Object Offset Transform: Since markers are affixed to the side surfaces of vessels rather than their geometric centers, a fixed rigid offset ${}^{\mathrm{t}}T_{\mathrm{o}}$ is applied to recover the true physical center of the object:

$$ {}^{\mathrm{b}}T_{\mathrm{o}}^{(i)} = {}^{\mathrm{b}}\tilde{T}_{\mathrm{t}}^{(i)} \cdot {}^{\mathrm{t}}T_{\mathrm{o}}. \tag{3} $$

The resulting transform ${}^{\mathrm{b}}T_{\mathrm{o}}^{(i)}$ is broadcast over the ROS 2 TF tree at 30 Hz. These localized poses form the raw spatial foundation from which the World State is assembled, as described in Section III-C2.

2) State Grounding and World Modeling

The 6-DoF poses broadcast by the Perception Module are assembled into the World State, a unified environmental representation that the TAMP Module consumes for motion planning. The World State is initialized at the beginning of each XDL procedure and updated after every stage transition, ensuring that the planner always operates on the current laboratory configuration. To populate the World State, the module partitions all objects in the workspace into two categories: target objects designated for manipulation in the current stage, and non-target objects treated as obstacles. Static elements such as tables and laboratory equipment are maintained as fixed obstacles, while movable objects such as beakers, bottles, and flasks are dynamically refreshed from the latest Perception Module output at each update. To keep planning tractable as the number of laboratory objects grows, all object models are represented using simplified geometric

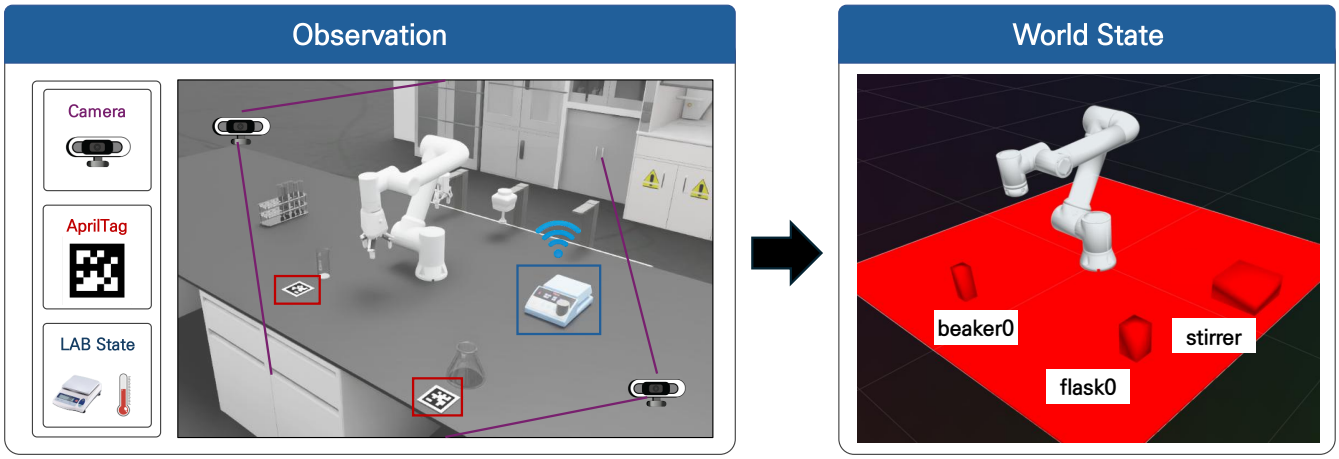


FIGURE 4. Overview of the Perception Module pipeline. Multi-view observations of the laboratory environment are fused and transformed into a World State, providing downstream modules with consistent 6-DoF pose estimates of laboratory assets.

primitives rather than full CAD geometries. Target objects are approximated as axis-aligned boxes or cylinders, while non-target obstacles are modeled as slightly enlarged primitives to provide a conservative collision margin. This abstraction reduces the computational burden of collision checking and path sampling while preserving execution feasibility, allowing the TAMP Module to scale to cluttered laboratory environments without a corresponding increase in planning overhead.

3) Multi-Modal Sensor Stream Processing

Beyond geometric world-state grounding, the Perception Module acts as a centralized telemetry hub that aggregates, filters, and distributes continuous data streams from various physical sensors. This real-time sensor processing is critical for closing the control loop in the Skill Library Module during physical execution, such as in adaptive pouring or temperature regulation. The module primarily processes two sensory modalities:

Scale Data: For precise quantitative liquid transfer, the module interfaces with an electronic precision scale. The raw weight measurements are acquired, synchronized with the system clock, and processed into a stable data stream. This filtered scale data is directly consumed by the Adaptive Pouring skill, serving as the primary feedback variable for the proportional-derivative (PD) controller to dynamically adjust the robot's wrist angle.

Temperature Data: To ensure strict adherence to chemical reaction conditions, the module also integrates temperature readings from external laboratory equipment, such as hotplate thermocouples. Raw temperature readings are monitored and smoothed to reject transient electrical noise. This provides a stable thermal state stream, which the device operation skills utilize to verify reaction temperatures and, if necessary, suspend task execution until the target thermal conditions are met.

By consolidating and filtering these heterogeneous data streams, the Perception Module shields the high-level planning and execution modules from hardware-specific noise, providing a clean and synchronized sensory interface for complex chemical manipulation.

D. TASK AND MOTION PLANNING MODULE

The TAMP Module computes collision-free, constraint-satisfying trajectories for multi-step manipulation procedures in chemistry experiments while keeping planning latency structurally bounded and predictable, as required by time-sensitive chemical workflows. As reviewed in Section II-B, the PDDLStream-based approach [6], which attempts to bind each continuous parameter independently, is structurally ill-suited to these requirements. To address this limitation, the TAMP Module is designed as an integrated planning framework that combines cuTAMP [8] with chemistry-specific operators and an LLM-guided tool-switching mechanism.

1) Chemistry-Specific TAMP

The TAMP Module adopts cuTAMP as its planning solver, formulating the planning problem as a parallelized differentiable optimization process that simultaneously drives a large batch of candidate particles toward the constraint manifold via gradient-based optimization. Unlike approaches that attempt to bind each parameter independently, cuTAMP jointly optimizes all constraints across batched candidate solutions, enabling convergence toward the feasible region even under tight orientation constraints on low-redundancy manipulators. By executing a fixed number of optimization steps over a fixed batch of particles, the runtime remains structurally bounded and predictable, ensuring that planning delays do not disrupt time-sensitive reaction conditions such as temperature control and reagent timing.

Chemistry laboratory manipulation imposes requirements beyond those of general pick-and-place tasks. Cylindrical vessels must be grasped stably despite their geometry, liquids must be transported without spillage and transferred in pre-

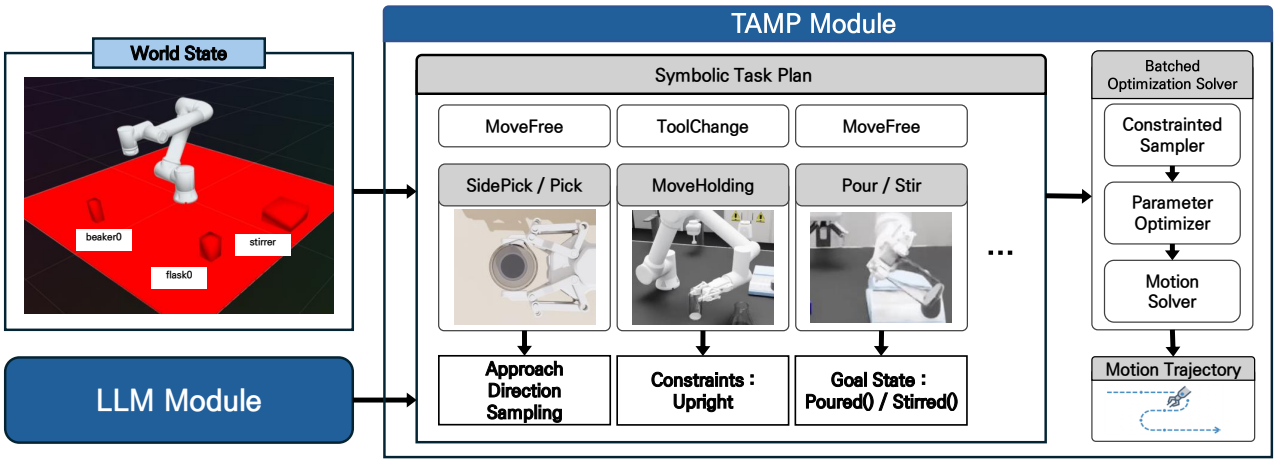
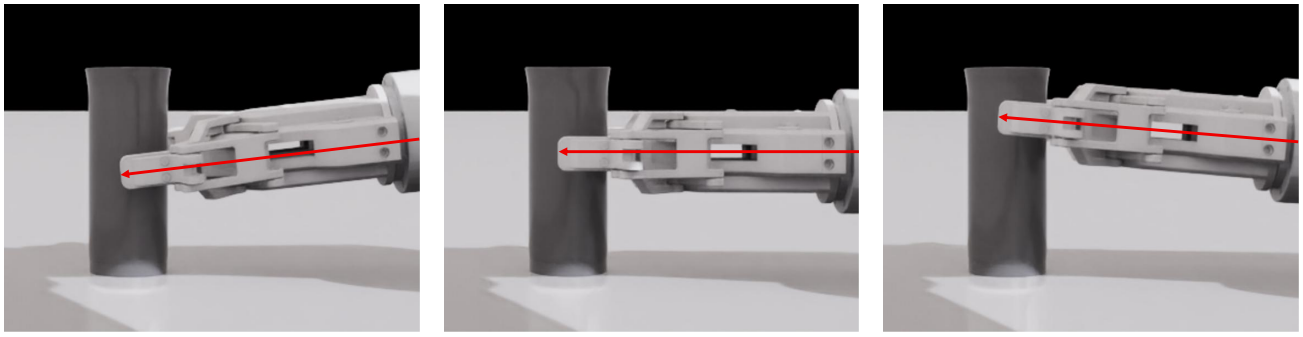


FIGURE 5. **Overview of the TAMP Module. The World State and the symbolic task plan, comprising chemistry-specific operators (SidePick, MoveHolding, Pour, Stir) alongside base operators (MoveFree, Pick), are passed to the Batched Optimization Solver, which sequentially executes a Constrained Sampler, Parameter Optimizer, and Motion Solver to produce the final motion trajectory.**



(a) (b) (c)

FIGURE 6. **Discrete yaw candidates for the SidePick operator, illustrating how gripper tilt varies across the sampled configurations. (a) $y = -\pi/2.5$, (b) $y = \pi/2$ (or $-\pi/2$), (c) $y = \pi/2.5$.**

cise quantities, and device-level operations such as stirring must be coordinated within the planning pipeline. To meet these requirements, we introduce four chemistry-specific operators in addition to the base TAMP operators: SidePick, for grasping the lateral surface of cylindrical vessels such as beakers and flasks; MoveHolding, for maintaining an upright orientation throughout the trajectory during liquid transport; Pour, for planning the tilting motion after the robot reaches the top of the target vessel; and Stir, a device-level operator for controlling the magnetic stirrer. Fig. 5 illustrates the overall structure of the TAMP Module, showing how chemistry-specific operators are composed into a symbolic task plan and resolved by the batched optimization pipeline.

The purpose of SidePick is to enable stable grasping of cylindrical vessels from any approach direction, where top-down grasping would otherwise be insufficient. In the object coordinate frame, where the z-axis is aligned with the vertical axis of the cylindrical vessel and the origin is located at the geometric center of the object, a grasp pose $\mathbf{g} = (\mathbf{t}, \mathbf{r})$ consists of translation $\mathbf{t} \in \mathbb{R}^3$ and orientation $\mathbf{r} = (r, p, y)$. To enable full-circumference side grasping of cylindrical objects, the approach angle α is uniformly sampled in the horizontal plane:

$$\alpha \sim \mathcal{U}(0, 2\pi). \tag{4}$$

The grasp orientation is determined as follows:

$$r = 0, \quad p = \alpha, \quad y \in \left\{-\frac{\pi}{2}, -\frac{\pi}{2.5}, \frac{\pi}{2.5}, \frac{\pi}{2}\right\}, \tag{5}$$

where roll r is fixed to zero such that the gripper x-axis aligns with the vertical direction, pitch p determines the approach direction around the object circumference via the approach angle α so that the gripper z-axis (finger approach direction) points horizontally toward the object center, and yaw y is discretely sampled from gripper tilt candidates. Discrete yaw sampling diversifies the gripper contact configuration on the vessel surface, thereby improving grasp stability across varying approach conditions, as illustrated in Fig. 6. The grasp height along the object z-axis is sampled within a range centered at the object origin:

$$t_z \sim \mathcal{U}\left(-\frac{h_o}{3}, \frac{h_o}{3}\right), \tag{6}$$

where the lateral position in the horizontal plane is determined by the approach direction α at a fixed radial offset from the object surface. This full-circumference sampling strategy improves search efficiency during particle initialization and enables stable grasping of rotationally symmetric vessels from any direction.

MoveHolding is an operator that enforces an upright constraint throughout the entire trajectory while the robot moves with a grasped object. It is introduced to prevent liquid spillage caused by vessel tilting during transport. The opening direction vector $\mathbf{u}_o$ of the grasped object is derived from the robot joint variables $\mathbf{q}$ and the fixed grasp transform $\mathbf{R}_{\text{grasp}}$ determined at the time of grasping:

$$\mathbf{u}_o(\mathbf{q}) = \mathbf{R}_{ee}(\mathbf{q}) \cdot \mathbf{R}_{\text{grasp}}^{-1} \cdot \mathbf{z}_w, \tag{7}$$

where $\mathbf{R}_{ee}(\mathbf{q})$ is the end-effector rotation matrix obtained from forward kinematics and $\mathbf{z}_w = [0, 0, 1]^T$. Since $\mathbf{R}_{\text{grasp}}^{-1} \cdot \mathbf{z}_w$ is a constant vector computed once at the time of grasping, this constraint can be evaluated solely from the robot joint variables without requiring real-time object pose estimation. The alignment error between this vector and the world z-axis is formulated as an upright cost J_{up}, which equals zero when the object is aligned with the vertical axis and increases as the object tilts:

$$J_{up}(\mathbf{q}) = 1 - \mathbf{u}_o(\mathbf{q}) \cdot \mathbf{z}_w. \tag{8}$$

When $J_{up} = 0$, the object is perfectly upright. For a maximum allowable tilt angle $\theta_{\max}$, the constraint is satisfied when

$$J_{up}(\mathbf{q}) \leq 1 - \cos(\theta_{\max}), \tag{9}$$

and the same constraint is also applied to the Place operator.[1]

Pour is an operator responsible for the tilting motion after the robot has reached the top of the target vessel. It is designed to decouple the transport phase from the actual pouring action, allowing the solver to focus on geometric validity and collision avoidance while delegating precise quantitative transfer to the Skill Library Module. The TAMP Module generates a geometric trajectory that tilts the opening of the vessel toward the target vessel, ensuring a valid starting pose and collision avoidance for the pouring action. This trajectory serves as the initial condition for the Adaptive Pouring skill described in Section III-E1, which dynamically adjusts the wrist angular velocity through PD control using real-time weight feedback from an electronic scale until the target amount is reached. Upon completion of the Pour operator, the state of the corresponding vessel transitions to Poured.

Stir is a device-level operator that transmits the target RPM and stirring duration to the magnetic stirrer. By directly invoking the device operation skill in the Skill Library Module without involving motion planning, it reflects the design principle that device-level operations are handled entirely at the execution layer. The stirrer operates at the specified RPM for the set duration, and upon completion of the Stir operator, the state of the corresponding vessel transitions to Stirred.

[1]In our experiments, $\theta_{\max}$ was set to 5° based on preliminary trials confirming that no liquid spillage occurred below this threshold for the vessel geometries used in the evaluation.

Algorithm 1 LLM-Guided Adaptive TAMP with Tool Switching

Require: XDL Procedure X_i, Next Procedure X_{i+1}, World State S, Current Tool T
Ensure: Motion Trajectory τ

```
[T_m, b_r, T_a, G*] ⇐ LLM_Reasoning(X_i, X_{i+1}, S)
if b_r is true then                                   ▷ Rearrangement required
    if T ≠ T_a then
        Execute_Tool_Change(T → T_a)
        T ⇐ T_a
    end if
    S ⇐ Update_World_State()                          ▷ Ground perception before rearrangement
    τ_a ⇐ cuTAMP_Planner(Pick-MoveHolding-Place, S, G*)
    Execute(τ_a)
end if
if T ≠ T_m then                                       ▷ Tool change for main task if needed
    Execute_Tool_Change(T → T_m)
    T ⇐ T_m
end if
S ⇐ Update_World_State()                              ▷ Final state update for main task
τ_m ⇐ cuTAMP_Planner(X_i, S, T_m)
return τ_m
```

If task planning or motion planning fails during any XDL procedure, up to three retries are performed. Upon a task-planning retry, the plan skeleton search is re-executed with new initial conditions; upon a motion-planning retry, particles are re-initialized and optimization is performed again. If a valid solution is not found after three retries, the system halts the workflow and returns an error message containing the failure cause and the relevant procedure information, allowing the user to diagnose and revise the experimental protocol.

2) LLM-Guided Adaptive Tool Switching and Auxiliary Action Insertion

Given the decisions made by the Action Reasoner, including which tool to mount, whether rearrangement is needed, and where to place the obstacle, this section describes how these discrete outputs are translated into executable steps within the TAMP pipeline. Tool switching is handled as a pre-execution step decoupled from the symbolic domain, while auxiliary rearrangement is inserted as a preceding Pick-MoveHolding-Place sequence when required. The full procedure is summarized in Algorithm 1. Here, T_m denotes the required tool for the main task, T_a the auxiliary tool for rearrangement, b_r a boolean flag indicating whether rearrangement is necessary, and G^* the target grid for the relocated object.

Tool changes are treated as pre-execution steps independent of the TAMP symbolic domain, thereby avoiding domain-redefinition overhead whenever the hardware configuration changes. When required, cuTAMP generates a collision-free trajectory to the tool exchange point, after which the physical exchange is completed by the Tool Changing skill in the Skill Library Module.

When rearrangement is required, an auxiliary Pick-MoveHolding-Place sequence is inserted before the main procedure, with a tool change to the auxiliary tool performed first

if necessary. Rearrangement targets are specified only at the level of a target workspace grid by the Action Reasoner, while precise placement is determined by cuTAMP through collision checking. The target zone is selected to reflect the context of the subsequent procedure, thereby reducing unnecessary rearrangement in later steps and improving overall execution efficiency. The stage-wise world-state update before and after each procedure ensures that plans reflect all preceding changes, including object positions, tool configuration, and auxiliary rearrangement outcomes.

E. SKILL LIBRARY MODULE AND EXECUTION

The Skill Library Module is an execution-layer module that instantiates the nominal trajectories generated by the TAMP Module into sensor-feedback-based motion primitives at execution time. Since TAMP considers only geometric constraints and logical sequencing, physical variables that are not explicitly modeled at the planning stage arise during execution. To address this, the Skill Library invokes the appropriate skill mapped to each TAMP operator, thereby bridging the gap between planning and execution. The Skill Library Module is organized into two categories: manipulation skills and device operation skills.

1) Manipulation Skills

This category comprises two skills: Adaptive Pouring and Automatic Tool Changing. The Adaptive Pouring skill corrects the nominal trajectory of the Pour operator using real-time weight feedback provided by the Perception Module described in Section III-C3. Because the flow rate of liquids is difficult to predict, precise quantitative transfer is impossible with the nominal trajectory alone. Upon reaching the pouring configuration, the wrist angle is dynamically adjusted through a PD controller based on the error between the target and current weight, achieving precise quantitative transfer until the target amount is reached.

The Automatic Tool Changing skill completes the tool exchange as a concrete hardware operation. The TAMP Module plans the tool exchange solely as a geometric motion to the tool rack, without accounting for the electronic control elements required for physical coupling and decoupling. Upon reaching the exchange point, control signals are applied to the electromagnet and the mechanical latching sequence is executed to complete the physical exchange.

2) Device Operation Skills

Device operation skills translate device commands, which TAMP treats as abstract symbols, into actual communication protocols and sensor feedback loops. Unlike the immediate state transitions assumed by TAMP, physical laboratory devices require time to reach target conditions, and the execution layer must ensure that those target conditions are actually met. This category consists of a single skill, the Feedback-Based Device Control skill.

The Feedback-Based Device Control skill communicates with the corresponding device upon receiving a Heat or Stir command and controls execution until the target physical conditions are met, thereby preventing reaction failures caused by premature progression. Internal sensor values such as temperature or RPM, streamed from the Perception Module described in Section III-C3, are continuously monitored, and if the target conditions are not met, execution is suspended or local feedback control is performed. This ensures that experimental conditions are enforced at the execution layer rather than merely assumed at the planning layer.

IV. EXPERIMENTS AND EVALUATION

The framework proposed in this paper is designed to address three core limitations identified in Section II-D. The first is that fixed SDL pipelines require manual redesign when procedures or apparatus change, limiting operational flexibility. The second is unreliable and unpredictable constrained motion planning on low-redundancy manipulators under orientation constraints. The third is that LLM-based planning systems remain confined to protocol generation without determining the execution context required to carry out each procedure, namely end-effector selection and obstacle rearrangement decisions. To validate that the proposed framework addresses all three limitations, this section evaluates XDL generation accuracy, TAMP planning reliability and latency, LLM-based tool selection and rearrangement reasoning, and end-to-end system performance. The evaluation proceeds from module-level assessment of individual components to end-to-end evaluation on long-horizon chemical workflows. Additional implementation details, hyperparameters, and demonstration videos are provided on the project page at https://rcilab.github.io/auto_chem.

A. EXPERIMENT SETUP

This section describes the hardware platform and simulation environment used to ensure reproducibility, the chemical tasks selected to span a range of manipulation difficulty, and the fine-tuning procedure for the two LLM components.

1) Hardware and Simulation Environments

The proposed framework is evaluated on a 6-DoF collaborative manipulator, the *FAIRINO-FR5*, as shown in Fig. 7. The manipulator is equipped with three end-effectors to cover the range of object geometries encountered in chemistry laboratory workflows: the 2-finger gripper (*AG95*) for general-purpose manipulation of standardized test tubes and small lab assets, the 3-finger gripper (*DH3*) for cylindrical vessels such as beakers and wide-neck flasks, and the suction gripper (*VGC10*) for objects with large flat surface areas such as container lids and storage boxes. As illustrated in Fig. 8, no single gripper configuration can reliably cover all object geometries, motivating the LLM-based dynamic tool assignment described in Section III-B3.

A high-fidelity simulation testbed was developed within *NVIDIA Isaac Sim*, generating 30 distinct initial configurations per task under identical constraint settings to support large-scale randomized trials and controlled ablation. The

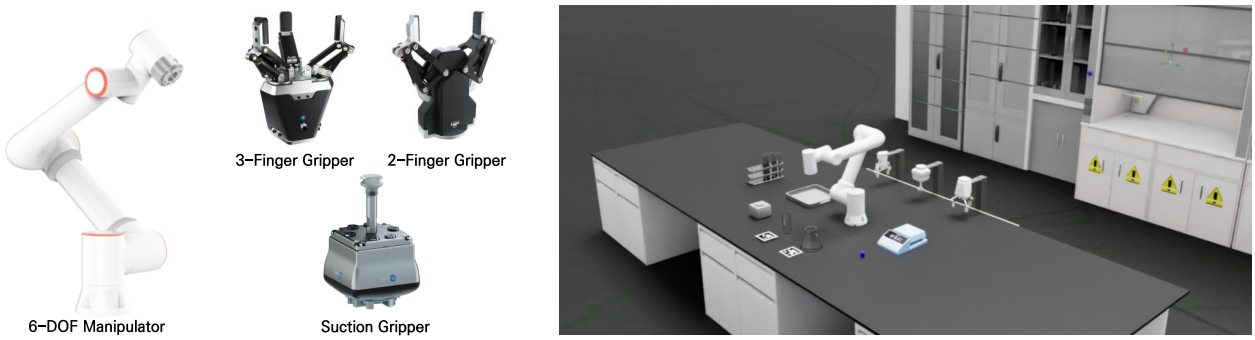


FIGURE 7. Hardware components and simulation environment for experimental evaluation. (a) The 6-DoF collaborative manipulator (*FAIRINO-FR5*) alongside three specialized end-effectors: the 2-finger gripper, 3-finger gripper, and suction gripper. (b) The simulation testbed is constructed using *NVIDIA Isaac Sim*.

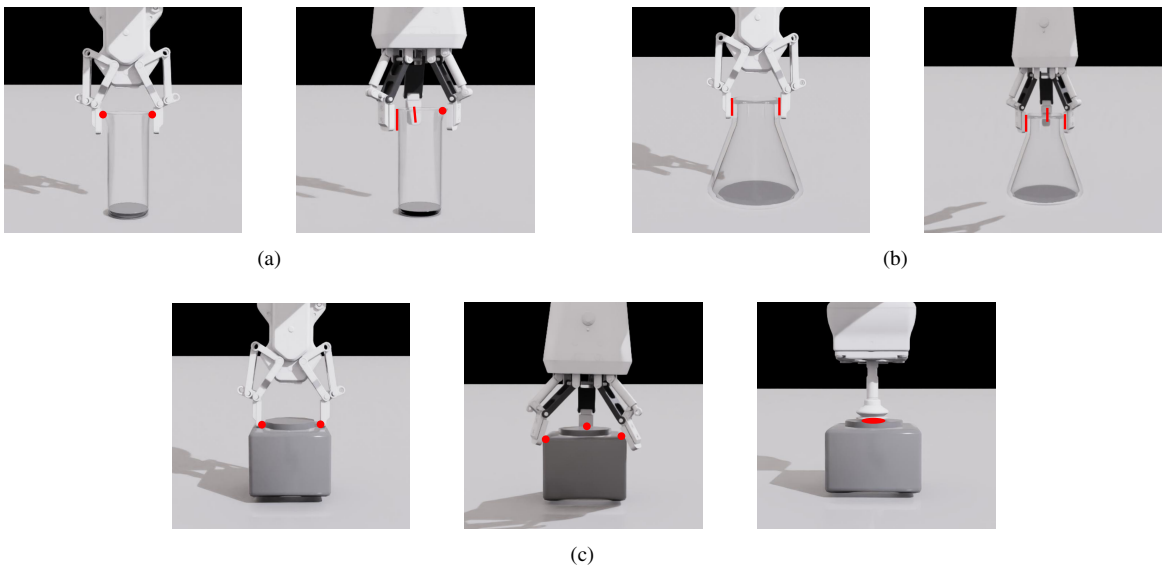


FIGURE 8. Motivation for multi-gripper tool selection. The 2-finger gripper provides insufficient contact area on the rim of cylindrical vessels, whereas the 3-finger gripper achieves stable top-down grasping. For large flat-surface objects exceeding the graspable width of finger grippers, the suction gripper is required. (a) Beaker, (b) Flask, (c) Large flat-surface object.

TAMP World State is synchronized with the simulation by extracting 6-DoF object poses and joint configurations at each stage, and the optimized trajectories are streamed back to the simulation for execution.

2) Chemical Experiments

Three core manipulation tasks are selected as representative evaluation scenarios, chosen to cover the essential procedures of real-world chemical experiments while spanning a range of motion planning complexity and manipulation difficulty. Move represents pick-and-place of laboratory objects and is evaluated using large-radius objects that require the suction gripper to assess the LLM's tool selection capability and manipulation success rate. Transfer represents liquid handling, a fundamental procedure in chemical synthesis, and tests the system's ability to maintain spill-free orientation constraints throughout the transit phase while tilting the source container and pouring its contents into a target vessel. Stir is chosen over simpler device operations such as heating because it demands a significantly higher level of spatial precision: the robot must not only relocate the vessel onto a magnetic stirrer but also precisely insert a magnetic stir bar into the narrow opening, making it an ideal scenario for evaluating complex, multi-step task and motion planning.

3) LLM Fine-Tuning and Training Details

The two LLM components introduced in Section III-B, the XDL Generator and the Action Reasoner, are each fine-tuned on a domain-specific dataset constructed to reflect the combinatorial structure of the target task. For the XDL Generator, the training dataset is constructed at a scale of 5,000 samples across four combinatorial axes: synonym variations for each

TABLE 1. Overall Performance of Natural Language to XDL Generation. Evaluated on 100 test instructions distributed as 30% single-step, 40% two-step, and 30% three-step composite workflows.

Metric	Count
Total Trials	100
Valid	97
Invalid	3
Success Rate (%)	**97.0**

of the six core XDL operators (Add, Stir, HeatChill, Transfer, CleanVessel, Move), continuous numerical parameter ranges (0–20 mL for volume, 0–20°C for temperature, 1–30 s for time), four instance identifiers per vessel type yielding 16 unique object identifiers, and command complexity ranging from single-operator instructions to three-step composite procedures with a weighted distribution of 30% single-step, 40% two-step, and 30% three-step. Since the task constitutes a structural mapping to a fixed XDL schema, the output space is inherently constrained, and this dataset sufficiently covers the principal combinatorial patterns, as later supported by the conversion results reported in Section IV-B1. Training is performed by applying LoRA (rank $r = 16$, $\alpha = 32$, dropout 0.05) to a 4-bit quantized *Llama 3.2 1B* model, using an 8-bit AdamW optimizer with a learning rate of 2×10^{-4}, an effective batch size of 16, and 5 epochs.

For the Action Reasoner, the training dataset consists of 3,000 samples generated combinatorially by placing up to four objects on 12 workspace grids. For each combination of the six XDL operators, rearrangement necessity is determined according to task-specific blocking criteria, and the ground-truth destination is assigned as either a context-aware target zone or a workspace edge depending on whether the obstacle is the target of the subsequent procedure. Candidate workspace grids are provided as input features after excluding fixed equipment and occupied grids, categorized solely by locational characteristics to prevent label leakage. The dataset is balanced across rearrangement-required and rearrangement-unnecessary cases, as well as context-aware and irrelevant obstacle cases, to prevent over-prediction of rearrangement. Training follows the same *Llama 3.2 1B* base model with LoRA configured at rank $r = 8$, $\alpha = 16$, dropout 0.05, a learning rate of 1×10^{-4}, batch size 16, and 3 epochs. The differing LoRA configurations used for the XDL Generator and Action Reasoner were selected empirically based on validation performance under their respective task structures, rather than through a dedicated hyperparameter ablation.

B. XDL GENERATION EVALUATION

This section evaluates the XDL generation pipeline across two aspects: the accuracy of the XDL Generator in converting natural language commands into valid protocols, and the ability of the XDL Validator to detect and intercept physically invalid protocols before execution.

1) Language Command to XDL Format

To assess whether the XDL Generator reliably converts natural language commands into valid protocols across varying task complexities, 100 test instructions spanning single-step to three-step composite workflows were evaluated. The test instructions were distributed across 30% single-step, 40% two-step, and 30% three-step procedures, and were constructed to include randomized synonym combinations and complex sentence structures beyond isolated commands. As summarized in Table 1, valid XDL protocols were generated in 97 out of 100 trials, with no syntax errors or semantic tag confusion observed across any test case. The three failures all occurred at the physical feasibility layer and originated from contradictions in the user instructions themselves, where a subsequent operator required a vessel state that had already been invalidated by a preceding operation. The average inference time per command was 1.57 seconds, confirming that the module introduces negligible latency relative to the overall pipeline execution time.

2) XDL Validator

To verify that the XDL Validator provides complete coverage of physically relevant failure modes, a benchmark test was conducted by injecting anomalous data across four error categories. While the three failures observed in the generation trials confirm that the validator correctly intercepts errors arising from user instruction contradictions, these cases alone are insufficient to verify coverage across all physically relevant failure modes. A benchmark test set was therefore constructed by injecting anomalous data across four error categories corresponding to the three-layer validation structure, as summarized in Table 2. The XDL Validator achieves 100% classification accuracy on the benchmark set, with zero false positives and zero false negatives, confirming that it provides complete coverage of the error space defined by the three-layer validation structure. In each case, the system returned a targeted error message such as ‘‘Cannot Stir empty vessel: beaker_A,’’ enabling the user to revise the instruction before any robot motion occurred. The resulting confusion matrix in Table 3 confirms that the validator reliably intercepts all injected errors regardless of error type, demonstrating that the three-layer structure provides complete and non-overlapping coverage of the physically relevant failure modes.

C. TAMP PLANNING RELIABILITY AND LATENCY EVALUATION

This section evaluates the ability of the TAMP Module to achieve reliable planning success rates and predictable termination time under orientation constraints on low-redundancy manipulators, addressing the structural limitation of sequential parameter-binding approaches identified in Section II-B. The proposed planning module is compared against PDDLStream as a representative sequential parameter-binding baseline across three chemical tasks, with 30 randomized initial configurations per task under identical starting poses, target conditions, and orientation constraints for spillage preven-

TABLE 2. Classification of Injected XDL Errors for Validator Benchmarking

Validation Layer	Error Category	Error Condition	Injected XDL Example	Remarks
Syntax	Missing required attribute	Intentional omission of mandatory attributes	`<Transfer to_vessel="flask_A" volume="10 mL" />`	from_vessel missing
Syntax	Undefined tag / attribute name	Use of disallowed tag or attribute name	`<Move object="flask_A" to_vessel="flask_B" />`	Using to_vessel instead of place in Move
Object existence	Non-existent object reference	Reference to object absent from workspace	`<Stir vessel="flask_C" time="30 s" />`	flask_C not registered in workspace
Physical feasibility	Physically infeasible sequence	Physical operation on an empty container	`<Stir vessel="beaker_A" time="1 min" />`	Stirring empty beaker_A

TABLE 3. XDL Validator Classification Accuracy. The benchmark set comprises 20 valid and 80 invalid samples, each evaluated once per error category across the three-layer validation structure.

	Validator: Pass (Positive)	Validator: Fail (Negative)
Valid	TP = 20	FN = 0
Invalid	FP = 0	TN = 80

tion, and a time budget of 60 seconds per trial. The proposed planning module was configured with 1,024 parallel particles and 1,000 optimization steps per planning attempt. The reported metrics reflect algorithmic planning latency exclusively, excluding physical execution and simulation rendering times.

The proposed planning module achieves higher planning success rates than the sequential parameter-binding baseline across all three tasks, with the most pronounced improvement on the Transfer task where orientation constraints are most restrictive, recording 100.0% versus 66.7% on the same 6-DoF manipulator. Beyond success rate, the proposed planning module exhibits structurally bounded and predictable runtime across all tasks, a property that sequential parameter-binding approaches cannot guarantee. These results are summarized in Table 4.

On the Transfer task, the proposed planning module achieves 100.0% success by jointly optimizing 1,024 candidate particles in parallel toward the constraint manifold, enabling effective exploration of the feasible region even when kinematic redundancy is limited. PDDLStream, by contrast, achieves only 66.7%: 10.0% of trials time out because its sequential sampling mechanism depends heavily on whether early random configurations happen to satisfy the orientation constraint, and the remaining 20.0% terminate early due to stream-evaluation failures in which the sampler exhausts valid parameter bindings before the time budget expires. On the Move task, the lower constraint complexity of top-grasp-based manipulation allows PDDLStream to reach 83.3%, whereas the proposed planning module eliminates the remaining failures and achieves 100.0%. On the Stir task, the requirement for precise stir-bar insertion into narrow vessel openings imposes tight spatial constraints that reduce PDDLStream to 60.0%, with 23.3% timeouts and 13.3% early terminations, while the proposed planning module maintains 100.0% success. Across all three tasks, these results show that batched joint differentiable optimization handles orientation and spatial constraints more reliably than sequential parameter binding on low-redundancy manipulators.

The two approaches also differ in temporal predictability. On the Transfer task, PDDLStream exhibits high variance among successful trials (std = 10.26 s), whereas the proposed planning module maintains a narrow time band (std = 1.44 s). This contrast reflects the outcome distribution of PDDLStream: trials in which early samples satisfy the constraints terminate quickly, whereas trials that do not either time out or terminate early due to stream-evaluation failures. By contrast, the proposed planning module succeeds on all 30 trials for the Move and Stir tasks; its nonzero standard deviation on these tasks reflects variation among successful solutions rather than a mixture of successes and unresolved failures. Overall, unlike PDDLStream, the proposed planning module consistently terminates within a bounded time envelope while returning a valid solution, a property that is critical for time-sensitive chemical workflows in which unpredictable planning delays may compromise reaction conditions.

D. LLM-BASED TOOL SELECTION AND REARRANGEMENT ACTION REASONING

To verify that the LLM Module reliably performs its two core functions, namely semantic tool assignment and context-aware obstacle rearrangement, the module is evaluated on unseen test scenarios not present in the fine-tuning dataset. The evaluation is structured as a two-stage ablation, because both functions are exercised within a single action sequence and removing the module entirely would make it difficult to distinguish their individual contributions. Section IV-D1 isolates tool assignment by evaluating inference accuracy on unseen scenarios independently of rearrangement. Section IV-D2, in contrast, isolates the rearrangement strategy by holding tool selection constant and comparing context-aware place-

TABLE 4. Comparison of planning success rate and latency between sequential parameter-binding (PDDLStream) and batched joint optimization-based (cuTAMP) TAMP solvers on the 6-DoF *FR5* manipulator. Planning time reports the mean ± std over successful trials only. Each condition is evaluated over 30 randomized trials with a 60 s timeout.

Task	Solver	Success (%)	Planning Time (s)	Timeout (%)	Early Term (%)
Transfer	PDDLStream	66.7	7.21 ± 10.26	10.0	20.0
	cuTAMP	100.0	31.18 ± 1.44	0.0	0.0
Move	PDDLStream	83.3	0.17 ± 0.12	0.0	16.7
	cuTAMP	100.0	17.60 ± 3.55	0.0	0.0
Stir	PDDLStream	60.0	0.79 ± 0.70	23.3	13.3
	cuTAMP	100.0	28.83 ± 1.00	0.0	0.0

TABLE 5. LLM-based tool selection and corresponding task performance. Each task–tool condition is evaluated on 30 unseen test scenarios not present in the fine-tuning dataset, for a total of 120 trials.

Task	Selected Tool	Target Object	Success Rate (%)
Move	3-finger gripper	Beaker, Flask, Tube	100.0
	suction gripper	Bottle, Box	96.67
Transfer	2-finger gripper	Beaker, Flask, Tube	100.0
Stir	3-finger gripper	Beaker, Flask, Tube	93.33

ment against no-rearrangement and random-rearrangement baselines. Section IV-E then verifies that the two functions combine effectively in the integrated pipeline through end-to-end evaluation.

1) Semantic Tool Assignment and Action Reasoner Evaluation

To assess whether the LLM Module assigns appropriate end-effectors across diverse task–object combinations, including object types not encountered during fine-tuning, tool selection accuracy is evaluated on 120 unseen test scenarios, comprising 30 trials per task–tool condition. As summarized in Table 5, the module achieves 100% accuracy for Move with the 3-finger gripper and Transfer with the 2-finger gripper, 96.67% for Move with the suction gripper, and 93.33% for Stir with the 3-finger gripper. The model consistently assigns the 3-finger gripper for cylindrical vessels across Move and Stir tasks, the 2-finger gripper for Transfer to enable side-grasp pouring, and the suction gripper for flat-surface objects that exceed the graspable width of finger-based end-effectors. The average inference time of 0.0665 s confirms that tool selection introduces negligible latency into the overall pipeline.

These results also reveal that no single fixed end-effector can cover the full set of evaluated task–object combinations. The 2-finger gripper cannot grasp large objects due to grip width constraints; the 3-finger gripper cannot execute the side-grasp-based pouring required for Transfer; and the suction gripper cannot achieve the stable insertion necessary for Stir. The tool selection accuracy reported above therefore validates not only the module's inference quality but also the architectural decision to support dynamic tool assignment as a prerequisite for operational coverage.

Beyond tool selection, the comprehensive reasoning capability of the Action Reasoner is evaluated on 600 unseen scenarios covering all four inference outputs: main tool, rearrangement necessity, auxiliary tool, and destination grid. As summarized in Table 6, main tool, rearrangement necessity, and auxiliary tool selection each achieve 100.0% accuracy. For the destination grid, category-level accuracy reaches 100.0%, while the exact grid match rate within a category is 76.5%. The Action Reasoner selects a single target workspace grid to specify the destination work zone, while precise placement coordinates within that zone are determined by the TAMP Module through collision checking in continuous space (Section III-B3). The category-level accuracy of 100.0% therefore confirms that the module fulfills this role in every scenario. The lower exact grid match rate reflects the fact that multiple grids within the same category can serve as valid placement candidates; since the final coordinate is resolved by the TAMP Module, this discrepancy does not affect downstream planning performance.

TABLE 6. Action Reasoner inference accuracy on unseen test scenarios. Evaluated on 600 unseen scenarios covering all four inference outputs across task–object combinations not present in the fine-tuning dataset.

Metric	Accuracy (%)
Main Tool	100.0
Need Rearrange	100.0
Rearrangement Tool	100.0
Target Grid (category)	100.0
Target Grid (exact)	76.5

2) Context-Aware Rearrangement Strategy Evaluation

To isolate the contribution of procedural context in rearrangement decisions, three conditions are compared. The No Rearrange condition executes the main task directly even when obstacles block the workspace. The Random Rearrange condition detects obstacles and relocates them to arbitrar-

TABLE 7. TAMP success rates and sequence completion rates by rearrangement strategy. Each sequence-condition pair is evaluated over 30 randomized trials.

Sequence	Metric	No Rearrange	Random Rearrange	Context-Aware
Transfer → Stir	Step 1 Success (%)	0.0	60.0	96.67
	Sequence Completion (%)	0.0	50.0	96.67
Stir → Transfer	Step 1 Success (%)	100.0	100.0	100.0
	Sequence Completion (%)	13.33	40.0	83.33

TABLE 8. End-to-end pipeline success rates for long-horizon chemical workflows. Each workflow is evaluated over 30 independent trials initialized with randomized World States and robot configurations.

Experimental Workflow	XDL Gen. Success (%)	Planning Success (%)	Execution Success (%)	End-to-End Success (%)
Move → Transfer	100.0	100.0	100.0	100.0
Transfer → Stir → Move	100.0	96.67	96.67	96.67
Move → Transfer → Stir	100.0	96.67	96.67	96.67
Overall Average	**100.0**	**97.78**	**97.78**	**97.78**

ily selected unoccupied grids, serving as an ablation that preserves spatial clearance but removes context-awareness. The Context-Aware condition uses the Action Reasoner to analyze the subsequent procedure and preemptively relocate obstacles to the work zone required by the next step. Tool selection is held constant across all three conditions to isolate rearrangement strategy as the sole independent variable.

Table 7 reports the Step 1 TAMP success rate and sequence completion rate for each condition across two evaluation sequences: Transfer → Stir, chosen because the wide workspace required by side-grasp pouring makes rearrangement most frequent, and the reverse order Stir → Transfer as a control. Context-aware rearrangement achieves the highest sequence completion rate in both cases, reaching 96.67% and 83.33%, compared to 50.0% and 40.0% for random rearrangement and 0% and 13.33% for no rearrangement.

The failure patterns across conditions clarify how context-awareness contributes beyond spatial clearance. Without rearrangement, Step 1 success on Transfer → Stir drops to 0% due to workspace congestion. Random rearrangement recovers Step 1 to 60.0% by clearing the immediate workspace, but since obstacles are placed without regard to the subsequent procedure, Step 2 encounters new collisions or requires additional rearrangement, limiting sequence completion to 50.0%. Context-aware rearrangement simultaneously secures workspace for Step 1 and positions obstacles where they will be needed for Step 2, achieving 96.67% sequence completion while eliminating the redundant manipulations that random placement incurs. These results show that procedural context, rather than spatial clearance alone, is the decisive factor for maintaining reliable execution across multi-step chemical workflows. Fig. 9 illustrates the resulting execution flow for a representative Transfer procedure, showing how the auxiliary rearrangement sequence is inserted prior to the main task when rearrangement is required.

E. END-TO-END SYSTEM EVALUATION

While the preceding sections have verified planning reliability (Section IV-C) and operational flexibility (Section IV-D) independently, this section evaluates whether these capabilities are preserved when the modules operate as an integrated pipeline. A trial is counted as successful only when every procedural step completes in order, starting from a single natural language command; any failure at an intermediate step is treated as a complete trial failure. Each workflow is evaluated over 30 independent trials initialized with randomized World States and robot configurations.

To identify potential bottlenecks across the pipeline, end-to-end performance is decomposed into three sequential stages: XDL generation, in which the LLM Module translates the natural language command into a validated experimental protocol; planning, in which the TAMP Module produces a collision-free trajectory with LLM-guided tool selection and rearrangement; and execution, in which the Skill Library instantiates the planned trajectory into sensor-feedback-based motion primitives. As shown in Table 8, XDL generation succeeds in 100% of trials across all three workflows. Planning and execution each achieve 96.67% on the two three-step workflows, with the Move → Transfer sequence reaching 100% across all stages. The consistent alignment between planning and execution success rates indicates that the Skill Library introduces no additional failure modes beyond those already captured at the planning stage, meaning that planning-stage verification alone is sufficient to predict end-to-end outcomes.

Representative execution snapshots of the Move → Transfer → Stir workflow are illustrated in Fig. 10, showing the robot successfully completing all three procedural stages within a single trial. The overall average success rate of 97.78% shows that the planning reliability and operational flexibility demonstrated in the preceding sections are preserved under integrated multi-step execution. End-to-end ex-

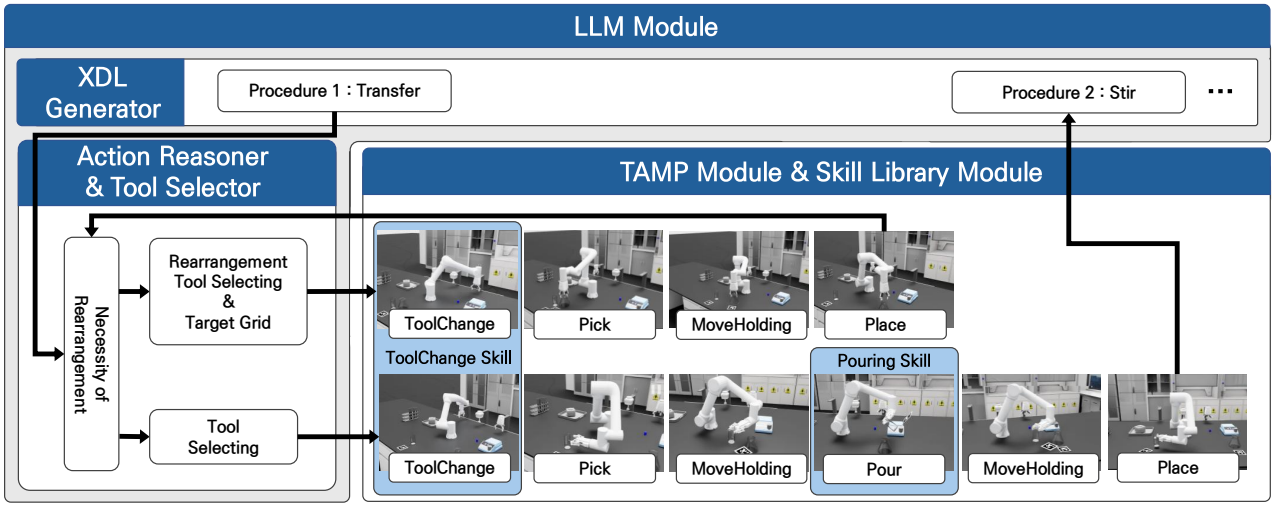
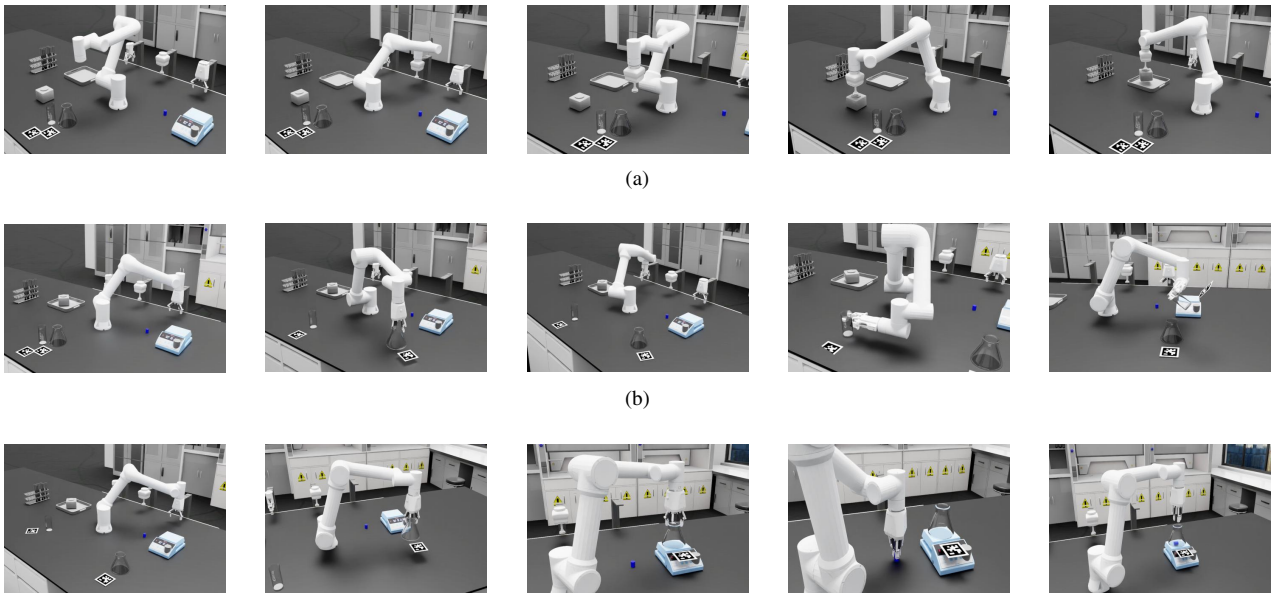


FIGURE 9. Execution flow of the TAMP pipeline for a Transfer procedure. The Action Reasoner determines tool selection and rearrangement necessity. When rearrangement is required, an auxiliary rearrangement sequence (upper path) is inserted prior to the main experimental procedure (lower path), which includes sensor-feedback-based skills such as adaptive pouring.



(a) (b) (c)

FIGURE 10. Sequential execution snapshots of the Move → Transfer → Stir workflow evaluated in the end-to-end experiments. Each row shows five representative keyframes captured during a single trial. (a) Move: the robot relocates the container to the target zone. (b) Transfer: liquid reagent is poured into the destination vessel. (c) Stir: the mixture is stirred on the hotplate to complete the reaction sequence.

ecution videos for all evaluated workflows are available on the project website at https://rcilab.github.io/auto_chem.

V. DISCUSSION AND LIMITATIONS

The experimental results in Section IV validate the proposed framework within a high-fidelity simulation testbed. This section discusses the assumptions and limitations that bound the interpretation of those results, focusing on four aspects: sim-to-real transfer of the execution layer, perception robustness under realistic laboratory conditions, generalization of the LLM components beyond the training distribution, and system-level error propagation.

A. SIM-TO-REAL TRANSFER AND PHYSICAL EXECUTION UNCERTAINTY

All quantitative results reported in this paper were obtained in *NVIDIA Isaac Sim*, which provides high-fidelity rigid-body kinematics and contact simulation but abstracts away the fluid-dynamic phenomena that dominate real-world liquid handling, including sloshing during transport, viscosity-dependent flow rates, and meniscus effects near vessel rims. The reported success rates for the Transfer task should therefore be interpreted as evidence of geometric and kinematic feasibility, namely collision-free, orientation-constrained tra-

jectory generation and valid pouring configurations, rather than as a claim of validated quantitative liquid-transfer accuracy.

The design of the Adaptive Pouring skill reflects this distinction. Precisely because analytical flow prediction is unreliable across reagents of varying viscosity, the skill does not depend on a fluid model: it closes the control loop on the measured weight from the electronic scale and adjusts the wrist angular velocity through a PD controller until the target mass is reached. This measurement-driven structure is consistent with sensor-feedback pouring systems that have been demonstrated on physical robots [12], [13]. Nevertheless, physical deployment introduces effects that the simulated feedback loop does not exercise: scale settling time and streaming latency, quantization and electrical noise in the weight signal, the delayed fluid response to wrist rotation, and gripper compliance under varying vessel weights. The filtering stage of the Perception Module (Section III-C3) and conservative tilt-rate limits mitigate, but cannot substitute for, controller gain tuning against real fluids; asymptotic slowdown near the target mass and explicit compensation for feedback latency will additionally be required to prevent overshoot with low-viscosity reagents. Quantifying the residual transfer error across viscosities and pour volumes on the physical FR5 platform is the first priority of our future work; the modular structure of the framework confines the required adaptation to the skill layer without affecting the LLM and TAMP Modules.

B. PERCEPTION ROBUSTNESS UNDER REALISTIC LABORATORY CONDITIONS

The Perception Module assumes that fiducial markers remain detectable throughout operation. In a working chemistry laboratory this assumption can be violated: markers may degrade through chemical spills, condensation, or thermal exposure near hotplates, and may be transiently occluded by the manipulator or by relocated vessels. The current design provides two layers of tolerance. First, the dual-camera configuration preserves tracking continuity under single-view occlusion, since the estimate of the remaining valid view is used directly when fusion is impossible (Section III-C1). Second, because the World State is grounded at procedure boundaries rather than continuously during motion (Section III-C2), transient detection losses between stages do not corrupt planning inputs: the most recent validated pose remains in effect, and objects are stationary between manipulations by construction of the workflow.

These mechanisms do not cover permanent marker loss. If a marker becomes unreadable in both views at a stage boundary, the affected object can no longer be localized, and the system halts and reports the failure rather than planning on stale geometry. Operationally, the framework assumes laminated markers mounted away from heated surfaces and periodic recalibration of the fixed camera extrinsics to bound calibration drift; the inverse-distance weighting additionally suppresses the influence of the more distant, and typically less accurate, view. The fixed tag-to-object offset transform assumes rigid vessels with known marker placement, which holds for standard labware but requires re-registration whenever markers are re-affixed. Likewise, the simplified geometric primitives used for collision modeling trade fidelity for tractability; because non-target obstacles are enlarged conservatively, approximation errors bias the planner toward rejecting marginal plans rather than toward collision, at the cost of occasional infeasibility in tightly cluttered scenes. Replacing fiducial tracking with marker-free 6-DoF pose estimation, and cross-checking geometric state against instrument-level feedback in the spirit of closed-loop quality control [15], are natural extensions that would relax these operational constraints.

C. GENERALIZATION OF THE LLM COMPONENTS BEYOND THE TRAINING DISTRIBUTION

Both LLM components are fine-tuned on combinatorially constructed datasets, and the evaluations in Section IV probe generalization only partially: test instructions include synonym and sentence-structure variations unseen during training, and tool selection is evaluated on object types absent from fine-tuning, but all evaluated commands remain within the six-operator XDL schema and the 12-grid workspace abstraction. Three out-of-distribution regimes therefore remain uncharacterized. Chemically ambiguous instructions, such as underspecified volumes or vessels, currently yield a generated protocol whose violated preconditions are intercepted by the XDL Validator, which halts execution and reports the offending step; this converts silent misinterpretation into an explicit rejection, but a systematic benchmark across ambiguity classes is left to future work. Adversarial or out-of-schema prompts are structurally contained by the schema-restricted output space, as the generator cannot emit operators outside the XDL vocabulary without failing syntactic validation, yet this containment has not been stress-tested against deliberately crafted inputs. Vessel geometries outside the training axes are partially covered by the unseen-object results in Table 5, where semantic affordance grounding transferred to unseen vessel types; substantially novel apparatus such as separatory funnels or condensers would, however, require extending both the operator schema and the end-effector inventory. Finally, the choice of a 1B-parameter backbone was driven by inference latency rather than by a capability comparison across model scales; establishing the minimum model capacity required for reliable protocol generation and action reasoning remains an open question.

D. ERROR PROPAGATION AND SYSTEM-LEVEL ROBUSTNESS

The hierarchical pipeline raises the concern that errors in early modules propagate downstream. The framework contains such propagation through three mechanisms, each exercised in Section IV. First, the XDL Validator rejects invalid protocols before any physical planning begins, intercepting 100% of injected errors (Table 3), so language-level failures terminate at the language layer. Second, the stage-wise World

State refresh before and after every procedure (Algorithm 1) re-grounds planning in observed geometry, preventing pose errors from accumulating across stages. Third, bounded retry with re-initialization at both the task- and motion-planning levels absorbs transient planning failures, and unrecoverable failures halt the workflow with a diagnostic message rather than proceeding on an invalid plan. The end-to-end decomposition in Table 8 provides empirical evidence that these mechanisms are effective: planning and execution success rates coincide across all workflows, indicating that no additional failure modes emerge at the execution layer, and the observed incompletions correspond to isolated planning failures rather than cascaded module errors.

Two design choices deserve a precise reading in this context. The 12-grid workspace discretization reduces the output space of the Action Reasoner to a reliably learnable discrete set without limiting final placement precision, because exact coordinates are resolved by cuTAMP in continuous space; the category-level destination accuracy of 100% (Table 6) supports this division of labor, and a finer grid would enlarge the symbolic output space and dilute per-cell training coverage without changing the continuous placement mechanism. Similarly, the bounded-latency property of the TAMP Module should be read as a per-attempt bound: the batched solver executes a fixed number of optimization steps over a fixed particle budget, so each planning attempt terminates within a bounded budget by construction, as reflected in the low variance of Table 4; end-to-end wall-clock time additionally includes up to three re-planning attempts in the failure path.

VI. CONCLUSION

This paper identified three coupled limitations that prevent existing approaches from achieving flexible and reliable chemistry laboratory automation: existing automation pipelines require manual redesign whenever new apparatus or procedures are introduced; constrained motion planning on low-redundancy manipulators suffers from degraded reliability and unpredictable termination time under orientation constraints; and LLM-based robot planning systems remain confined to protocol generation without determining the execution context required at each procedural stage. To address all three simultaneously, we proposed the LLM-Guided Tool-Aware TAMP framework, which unifies LLM-based context-aware decision-making with batched joint differentiable optimization-based planning within a single end-to-end pipeline.

The LLM Module jointly addresses the first and third limitations. By converting natural-language commands into validated XDL protocols, it eliminates the need for manual protocol redesign. By further inferring the appropriate end-effector and obstacle rearrangement strategy from the current World State and procedural context, it extends the role of the LLM beyond protocol generation to execution-context determination, achieving 100% tool selection accuracy on standard task–object conditions and 96.67% sequence completion through context-aware obstacle positioning, even for object types absent from the training distribution. The TAMP Module addresses the second limitation by replacing sequential parameter-binding with batched joint differentiable optimization, improving the planning success rate from 66.7% to 100.0% on the Transfer task where orientation constraints are most restrictive, while maintaining a structurally bounded runtime appropriate for time-sensitive chemical workflows. Together, these capabilities enable the integrated framework to achieve a 97.78% end-to-end success rate on long-horizon chemical workflows initiated from a single natural language command.

Future work will prioritize physical deployment on the FR5 platform to validate and quantify the sim-to-real gap discussed in Section V, with particular focus on adaptive pouring under real fluid dynamics and sensor noise, alongside expanding the supported XDL operators and skill library to cover more complex chemical procedures such as titration, filtration, and centrifugation, and benchmarking the LLM components across model scales and out-of-distribution instruction regimes.

REFERENCES

[1] M. Seifrid, R. Pollice, A. Aguilar-Granda, Z. Morgan Chan, K. Hotta, C. T. Ser, J. Vestfrid, T. C. Wu, and A. Aspuru-Guzik, "Autonomous chemical experiments: Challenges and perspectives on establishing a self-driving lab," *Accounts of Chemical Research*, vol. 55, no. 17, pp. 2454–2466, 2022.

[2] M. Abolhasani and E. Kumacheva, "The rise of self-driving labs in chemical and materials sciences," *Nature Synthesis*, vol. 2, no. 6, pp. 483–492, 2023.

[3] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *Annual review of control, robotics, and autonomous systems*, vol. 4, no. 1, pp. 265–293, 2021.

[4] 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*, vol. 40, no. 6-7, pp. 866–894, 2021.

[5] N. Yoshikawa, A. Z. Li, K. Darvish, Y. Zhao, H. Xu, A. Kuramshin, A. Aspuru-Guzik, A. Garg, and F. Shkurti, "Chemistry lab automation via constrained task and motion planning," *arXiv preprint arXiv:2212.09672*, 2022.

[6] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, "Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning," in *Proceedings of the international conference on automated planning and scheduling*, vol. 30, 2020, pp. 440–448.

[7] Z. Kingston, M. Moll, and L. E. Kavraki, "Exploring implicit spaces for constrained sampling-based planning," *The International Journal of Robotics Research*, vol. 38, no. 10-11, pp. 1151–1178, 2019.

[8] W. Shen, C. Garrett, N. Kumar, A. Goyal, T. Hermans, T. Lozano-Perez, and F. Ramos, "Differentiable gpu-parallelized task and motion planning," in *Robotics science and systems*, 2025.

[9] B. Burger, P. M. Maffettone, V. V. Gusev, C. M. Aitchison, Y. Bai, X. Wang, X. Li, B. M. Alston, B. Li, R. Clowes *et al.*, "A mobile robotic chemist," *Nature*, vol. 583, no. 7815, pp. 237–241, 2020.

[10] H. Fakhruldeen, G. Pizzuto, J. Glowacki, and A. I. Cooper, "Archemist: Autonomous robotic chemistry system architecture," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6013–6019.

[11] A. M. Lunt, H. Fakhruldeen, G. Pizzuto, L. Longley, A. White, N. Rankin, R. Clowes, B. Alston, L. Gigli, G. M. Day *et al.*, "Modular, multi-robot integration of laboratories: an autonomous workflow for solid-state chemistry," *Chemical Science*, vol. 15, no. 7, pp. 2456–2463, 2024.

[12] M. Kennedy, K. Schmeckpeper, D. Thakur, C. Jiang, V. Kumar, and K. Daniilidis, "Autonomous precision pouring from unknown containers," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2317–2324, 2019.

[13] Y. Huang, J. Wilches, and Y. Sun, "Robot gaining accurate pouring skills through self-supervised learning and generalization," *Robotics and Autonomous Systems*, vol. 136, p. 103692, 2021.
[14] N. Yoshikawa, G. D. Akkoc, S. Pablo-García, Y. Cao, H. Hao, and A. Aspuru-Guzik, "Does one need to polish electrodes in an eight pattern? automation provides the answer," *Digital Discovery*, vol. 4, no. 2, pp. 326–330, 2025.
[15] S. U. Khan, V. K. Møller, R. J. N. Frandsen, and M. Mansourvar, "Real-time AI-driven quality control for laboratory automation: a novel computer vision solution for the opentrons OT-2 liquid handling robot," *Applied Intelligence*, vol. 55, no. 7, p. 524, 2025.
[16] R. I. C. Muchacho, R. Laha, L. F. Figueredo, and S. Haddadin, "A solution to slosh-free robot trajectory optimization," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 223–230.
[17] Z. Kingston, M. Moll, and L. E. Kavraki, "Sampling-based methods for motion planning with constraints," *Annual review of control, robotics, and autonomous systems*, vol. 1, no. 1, pp. 159–185, 2018.
[18] D. Coleman, I. Sucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: A moveit! case study," *arXiv preprint arXiv:1404.3785*, 2014.
[19] Y. Li, L. Ke, and C. Zhang, "Dynamic obstacle avoidance and grasping planning for mobile robotic arm in complex environment based on improved TD3," *Applied Intelligence*, vol. 55, no. 11, p. 776, 2025.
[20] S. H. M. Mehr, M. Craven, A. I. Leonov, G. Keenan, and L. Cronin, "A universal system for digitization and automatic execution of the chemical synthesis literature," *Science*, vol. 370, no. 6512, pp. 101–108, 2020.
[21] N. Yoshikawa, M. Skreta, K. Darvish, S. Arellano-Rubach, Z. Ji, L. Bjørn Kristensen, A. Z. Li, Y. Zhao, H. Xu, A. Kuramshin *et al.*, "Large language models for chemistry robotics," *Autonomous Robots*, vol. 47, no. 8, pp. 1057–1086, 2023.
[22] K. Darvish, M. Skreta, Y. Zhao, N. Yoshikawa, S. Som, M. Bogdanovic, Y. Cao, H. Hao, H. Xu, A. Aspuru-Guzik *et al.*, "Organa: A robotic assistant for automated chemistry experimentation and characterization," *Matter*, vol. 8, no. 2, 2025.
[23] A. M. Bran, S. Cox, O. Schilter, C. Baldassari, A. D. White, and P. Schwaller, "Chemcrow: Augmenting large-language models with chemistry tools," *arXiv preprint arXiv:2304.05376*, 2023.
[24] D. A. Boiko, R. MacKnight, B. Kline, and G. Gomes, "Autonomous chemical research with large language models," *Nature*, vol. 624, no. 7992, pp. 570–578, 2023.
[25] Y. Qu, X. Hu, F. Chen, and Z. Wei, "TRTP: a three-stage robust task planning framework for open worlds via visual-language models and digital twin simulation," *Applied Intelligence*, vol. 56, no. 4, p. 107, 2026.
[26] J. Wang and E. Olson, "Apriltag 2: Efficient and robust fiducial detection," in *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2016, pp. 4193–4198.
[27] K. M. Lynch and F. C. Park, *Modern robotics*. Cambridge: Cambridge University Press, 2017.

BOHYEONG PARK received the B.S. degree from the Department of Mechanical Engineering, Hanbat National University, in 2025. He is currently pursuing the M.S. degree with the Department of Mechanical Engineering, Kyung Hee University. His current research interests include collision avoidance, manipulation, and vision language model planning.

HYUNHO CHO received the B.S. degree from the Department of Mechanical Engineering, Hanbat National University, in 2025. He is currently pursuing the M.S. degree with the Department of Mechanical Engineering, Kyung Hee University. His current research interests include optimal control and reinforcement learning.

SANGHYUN KIM (Member, IEEE) received the B.S. degree in mechanical engineering and the Ph.D. degree in intelligent systems from Seoul National University, South Korea, in 2012 and 2020, respectively. He was a Postdoctoral Researcher with The University of Edinburgh, U.K., in 2020, and a Senior Researcher with the Korea Institute of Machinery and Materials, South Korea, from 2020 to 2023. He is currently an Assistant Professor in mechanical engineering with Kyung Hee University. His main research interest includes the optimal control of mobile manipulators.