

## 언어 모델을 이용한 상황 추론 기반 작업 계획 생성

## Context-Driven Task and Motion Planning with Language Models

이 진 휘<sup>†</sup>Jinhwi Lee<sup>†</sup>

**Abstract:** This paper presents a task planning framework for autonomous robots that combines Large Language Models (LLMs) and Vision-Language Models (VLMs) to enable context-driven reasoning and robust action generation. The framework infers preconditions by structuring environmental states and task goals into contextual representations, which are then refined through prompt engineering. These structured prompts are aligned with a predefined action library to generate task sequences that are both logically consistent and physically executable. By integrating symbolic task reasoning with motion planning, the method bridges the gap between high-level natural language understanding and low-level robotic control. Experiments conducted in simulated dynamic environments demonstrate that the approach produces reliable and adaptable task plans, even under incomplete or ambiguous natural language instructions. The results suggest that context-based inference and prompt optimization with LLM/VLMs provide a scalable pathway for developing autonomous robots capable of flexible task execution in real-world environments.

**Keywords:** Task and Motion Planning, Manipulation, Language-Driven Robotics

## 1. 서 론

현대 로봇 공학은 통제된 산업 현장을 넘어 가정이나 물류 센터와 같은 비정형적이고 동적인 실제 환경으로 그 응용 범위를 넓혀가고 있다. 이러한 환경에서 로봇이 자율적으로 작업을 수행하기 위해서는 사용자의 모호한 지시를 이해하고 상황에 맞는 고수준의 작업(Task) 순서를 결정함과 동시에, 이를 물리적인 충돌 없이 실행할 수 있는 저수준의 모션(Motion) 궤적을 통합적으로 생성해야 한다. 이를 해결하기 위한 Task and Motion Planning (TAMP)은 기호적 추론과 연속적 제어의 결합을 목표로 오랜 기간 연구되어 왔다. 그러나 전통적인 TAMP 방식은 사전에 엄격하게 정의된 규칙과 도메인 지식에 의존하기 때문에, 환경의 불확실성이나 예기치 못한 맥락 변화에 유연하게 대처하기 어렵다는 근본적인 한계를 지닌다. 그러나 이러한 기법들은 환경의 불확실성과 작업 맥락(context)의 불완전성을 충분히 다루는 데 한계를 지닌다.

최근 거대 언어 모델(Large Language Model, LLM)과 비전 언어 모델(Vision Language Model, VLM)의 비약적인 발전은 로봇이 자연어와 시각 정보를 바탕으로 상황을 인지하고 스스로 행동을 계획할 수 있는 새로운 패러다임을 제시하고 있다. 그러나 이를 물리적인 로봇 조작에 직접 적용하는 데에는 여전히 두 가지 주요한 문제가 존재한다. 첫째, 실제 사용자의 자연어 명령은 종종 불완전하며 작업 실행에 필수적인 선행조건이 생략되는 경우가 많아 언어 모델 단독으로는 완벽한 계획을 수립하기 어렵다. 둘째, 언어 모델이 텍스트 기반으로 생성한 논리적 계획은 로봇의 구체적인 기구학적 한계, 도달 가능성 등 저수준 제약을 온전히 반영하지 못해 물리적으로 실행 불가능한 지시를 내리는 경우가 빈번하다. 본 논문은 이러한 한계를 극복하고, 거대 언어 모델의 강력한 의미론적 추론 능력과 실제 로봇의 물리적 실행 가능성을 유기적으로 결합한 통합 TAMP 프레임워크를 제안한다.

## 2. 관련 연구

TAMP는 기호적 의사결정과 연속적인 궤적 계획을 결합하여 장기적 목표 달성과 물리적 실행 가능성을 동시에 확보하는 것을 목표로 한다. 기존 연구에서는 제약 기반(constraint-based),

Received : Sep. 5. 2025; Revised : Apr. 30. 2026; Accepted : May. 18. 2026

※ This work was supported by the Academic Promotion System Tech University of Korea.

† Assistant Professor, Corresponding author: Department of Mechatronics Engineering, Tech University of Korea, Siheung, Korea (jlee@tukorea.ac.kr)

Copyright©KROS

샘플링 기반(sampling-based), 최적화 기반(optimization-based) 접근 방법으로 발전해 왔으며 이러한 방법론은 물체 조작, 경로 계획, 다중 로봇 제어 등 다양한 문제에 적용되고 있다<sup>[1-4]</sup>. 일반적으로 로봇 물체 조작 영역에서 PDDL (Planning Domain Definition Language) 기반의 Task 계획과 기하학 기반의 Motion 계획을 연계한 TAMP 알고리즘이 제안되어 왔다<sup>[5,6]</sup>. 그러나 이러한 규칙 기반 기법들은 환경의 불확실성과 작업 맥락의 불완전성을 충분히 다루는 데 한계가 있기 때문에 객체 및 관계 정보를 포함하는 그래프 기반 상태 표현을 활용하고 이를 탐색 기반 알고리즘과 결합하여 효율적인 계획을 생성하는 방법도 연구되었다<sup>[7]</sup>. 이러한 방법론들은 수학적 타당성을 보장하지만 모든 환경 모델과 상태 전이 규칙을 사전에 엄격하게 수식화해야 하므로 모델링되지 않은 예외 상황이나 비정형 환경에 직접 적용하기에는 구조적 제약이 따른다.

최근에는 비정형적 명령과 환경 단서를 해석하기 위해 LLM과 VLM을 로봇에 적용하는 연구가 활발하다. 초기 연구들은 LLM을 제로샷(Zero-shot) 플래너로 활용하여 하위 목표를 분해하거나<sup>[8-10]</sup>, 시각 정보와 언어를 결합하여 로봇 행동을 직접 산출하는 VLA (Vision Language Action) 모델 구조를 제시하였다<sup>[11,12]</sup>. 또한 Affordance 기반 Prompt를 활용한 동작 계획 도출<sup>[13,14]</sup>, LLM과 PDDL을 결합한 문제 정의 및 계획 생성<sup>[15]</sup>, 인간-로봇 협업(Human Robot Collaboration, HRC) 환경에서의 시각 인식 및 모션 제어 연계<sup>[16,17]</sup> 등 구체적인 응용 연구들이 보고되고 있다. 더 나아가 검색 기반 증강(Retrieval-Augmented Generation, RAG)을 결합하여 환경 변화에 적응하는 방법론도 제안되었다<sup>[18]</sup>. 이러한 연구들은 거대 인공지능이 단순히 자연어 처리 도구를 넘어 조작 작업의 계획 및 실행을 지능적으로 지원할 수 있는 핵심 요소로 자리 잡아가고 있음을 보여준다. 그러나 기존의 LLM 기반 작업 계획 연구들은 주로 명확히 정제된 언어 명령이나 사전에 잘 정의된 초기 환경 상태를 전제로 작동한다는 한계를 지닌다. 즉, 실제 환경에서 사용자의 명령이 불완전하거나 작업 실행에 필수적인 선행조건이 누락되었을 때, 이를 시각 및 환경적 맥락으로부터 추론하여 보완하는 능력이 부족하다. 또한 LLM이 생성한 고수준의 작업 계획이 로봇의 기구학적 한계나 물리적 충돌 등 제약으로 인해 실행 불가능할 경우가 충분히 고려되지 않았다.

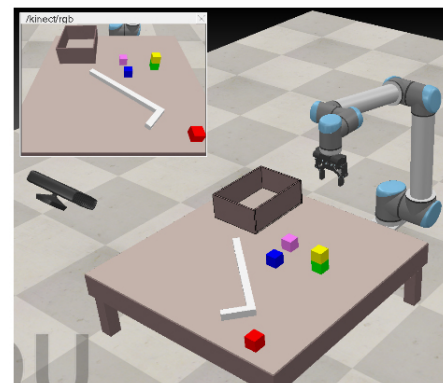
따라서 본 논문은 기존 접근법의 한계를 극복하기 위해 TAMP의 기호적 추론 능력과 거대 인공지능의 맥락 이해 능력을 융합하여 비정형 환경에서도 실행 가능한 작업 계획을 생성하는 프레임워크를 제안하고 실효성을 검증한다. 이에 다음과 같이 세부 목적을 설정한다: (i) 자연어 명령과 환경 상태를 기반으로 맥락적 선행조건(precondition)을 추론하여 불완전하거나 모호한 입력을 보완한다. (ii) 추론된 맥락 정보를 활용해 Prompt engineering을 수행하고, 이를 통해 액션 라이브러리와

연계된 실행 가능한 작업 시퀀스를 생성한다. (iii) High-level의 Task 계획과 Low-level의 Motion 계획을 통합하여 실제 물리적 환경에서 실행 가능한 계획을 산출한다. 본 연구의 핵심 기여는 거대 언어 모델의 기호적 추론과 로봇의 물리적 실행 한계를 유기적으로 연결하는 맥락 기반 선행조건 추론 및 피드백 기반 재탐색이 결합된 통합 TAMP 프레임워크를 제안하는데 있다. 구체적 내용은 다음과 같다: (i) 비정형 환경과 불완전한 자연어 명령으로 인한 정보 누락을 LLM/VLM을 활용한 맥락 기반 선행조건 자율 추론을 통해 해결함으로써 기존 TAMP의 초기 상태 정의의 한계를 극복한다. (ii) 언어 모델이 생성한 계획이 갖는 물리적 실행 불가능성의 한계를 Low-level 피드백 기반의 재탐색 구조로 보완하여 논리적 타당성과 물리적 실행 가능성을 동시에 보장한다. 이와 같은 핵심 기여를 실현하기 위해 구조화된 Prompt engineering과 Action library 매핑을 세부 방법론으로 적용하였으며, 동적 시뮬레이션 환경에서의 검증을 통해 본 프레임워크가 실제 산업 및 서비스 로봇의 유연한 작업 수행에 실효적으로 적용될 수 있음을 확인하였다.

### 3. 작업 계획 생성 프레임워크

#### 3.1 맥락 기반 선행조건 추론

로봇이 자연어 명령을 받아 작업 계획을 수립하기 위해서는 해당 작업이 실행 가능한지 여부를 판단할 수 있는 선행조건이 정의되어야 한다. 기존의 TAMP에서의 선행조건은 사전에 정의된 규칙 기반 도메인에 의존하였으나 이는 환경의 불확실성이나 사용자의 불완전한 명령에 효과적으로 대응하기는 어렵다는 한계가 있다. 본 논문에서는 LLM과 VLM을 활용하여 환경 맥락을 구조화하고 이를 기반으로 물체 조작을 하기 위해서 작업 실행에 필요한 선행 조건을 자동으로 추론하는 방법을 제안한다.



[Fig. 1] An example of an environment for object manipulation tasks. There are boxes (red, blue, yellow, green, and pink), an L-shaped tool, a bin and a table. The boxes, L-shaped tool, and bin are located on the table

## [Algorithm 1] Context-Based Precondition Inference

**Input:** Natural language command  $u$ , Sensor observation  $x$ **Output:** Validated precondition set  $P$ 

```

1: entity, intent, behavior  $\leftarrow$  LLM_PARSE_COMMAND( $u$ )
2:  $Z \leftarrow$  VLM_PERCEIVE( $x$ )
3:  $C \leftarrow$  SYMBOLIZE( $Z$ , entity, intent, behavior)
4:  $G \leftarrow$  LLM_PRECONDITION(entity, intent,  $C$ )
5:  $P \leftarrow \emptyset$ 
6: for  $g$  in  $G$  do
7:   if CONSISTENT( $g$ ,  $C$ ) and RELEVANT( $g$ , entity, intent)
8:     then
9:        $P \leftarrow P \cup \{g\}$ 
10:   end if
11: end for
12: return  $P$ 

```

물체 조작을 위해서는 먼저 입력된 언어 명령 형태에 대해 조작할 대상, 목표 위치 그리고 동작 유형이 구분되어야 한다. 예를 들어, 작업 공간에 존재하는 산재된 box를 bin에 정리하는 명령이 주어지면 조작 대상은 box, 목표 위치는 bin 그리고 동작 유형은 Pick과 Place로 정의할 수 있다. 즉, LLM 기반으로 구문을 분석하고 의미론적으로 분해하여 작업 목표(Goal state)를 정의하도록 사용자 입력에 대해 Natural Language Command Parsing을 수행한다.

환경 상태는 카메라 센서를 통해 획득한 데이터 기반으로 VLM으로 처리한다. VLM은 객체의 정의, 상태 그리고 상관관계를 추출하여 기호적 표현으로 변환한다. 예를 들어, [Fig. 1]의 경우 box, bin, L-shaped tool, table이 객체로 정의되고 각 객체의 상태와 객체간 상관관계가 표현된다.

작업 목표와 환경 상태가 정의되면 이를 기반으로 해당 작업이 실행되기 위해 필요한 선행조건을 추론한다. 먼저 LLM은 목표 상태에 도달하기 위한 선행조건 후보를 생성한다. [Fig. 1]에서 box를 bin에 정리한다는 목표가 주어졌을 때 선행조건으로는 작업 전의 box와 bin의 현재 위치, 다른 객체와의 상관관계, 상관관계에 따른 다른 객체의 상태가 필요하며 작업 후의 각 상태가 필요하다. 이를 통해 작업 전 초기 상태와 목표 도달 조건이 함께 정의된다.

맥락 기반의 선행조건 추론을 이용하면 사용자 입력에서 세부 정보가 부족한 경우 환경 맥락을 통해 누락된 조건을 보완할 수 있다. [Fig. 1]에서 green box를 bin에 위치시키는 명령이 주어졌을 때 yellow box가 green box위에 위치하므로 작업을 한번에 수행할 수 없다. 이 때 yellow box와 green box의 상관관계(stack과 같은 물체 간 접촉)를 정의함으로써 선행조건을 보완할 수 있다. 다음과 같이 구조화된 선행조건은 이후 Prompt engineering과 Action library 매핑에 직결된다.

[Algorithm 1]은 맥락 기반으로 추론된 선행조건을 출력한다. 먼저 자연어 명령( $u$ )이 입력되면 LLM command parsing을 수행하여 조작 대상(entity), 목표 위치(intent), 동작 유형(behavior)를 추출한다(line 1). 다음으로 환경 상태를 추론하기 위해 센서로 입력된 환경 정보( $x$ )를 기반으로 VLM을 통해 환경에 포함된 객체 및 상관관계 집합( $Z$ )을 정의한다(line 2). 그리고 LLM과 VLM으로 추출된 정보를 기반으로 구조화된 맥락( $C$ )을 추출한다(line 3). 맥락이 구조화 되면 조작 대상과 목표 위치와 함께 LLM을 통해 선행조건의 후보 집합( $G$ )을 추출한다(line 4). 추출된 선행조건의 신뢰성을 확보하기 위해 선행조건 집합  $G$ 의 각 원소( $g$ )를 기존의 맥락( $C$ )과의 일관성과 목표와의 상관관계를 다시 검증하여 최종적인 선행조건( $P$ )을 출력한다(line 5-11).

## 3.2 작업 계획을 위한 Prompt engineering

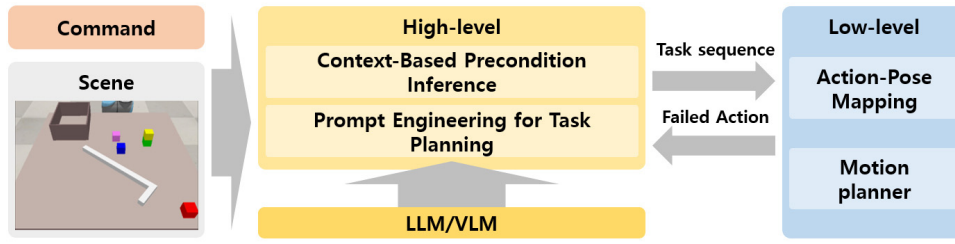
로봇이 자연어 명령을 기반으로 실행 가능한 작업 계획을 생성하기 위해서는 LLM과 VLM이 환경 맥락과 선행조건을 이해할 수 있도록 입력을 설계해야 한다. 이러한 과정을 Prompt engineering이라고 하며 본 논문에서는 구조화된 맥락 정보를 반영하여 로봇이 수행 가능한 작업 계획을 안정적으로 생성할 수 있는 방법을 제안한다. 본 연구에서 활용된 prompt는 Appendix에 제시한다.

[Table 1] Prompt output according to context-based precondition inference result in Fig. 1

|                      |                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Command              | Put the blue and green boxes into the bin                                                                                                                      |
| Objects              | red box, blue box, yellow box, green box, pink box, L-shaped tool, bin, table                                                                                  |
| Object relationships | on (red box, table), on (blue box, table), on (yellow box, green box), on (green box, table), on (pink box, table), on (L-shaped tool, table), on (bin, table) |
| Goal state           | <b>in (blue box, bin), in (green box, bin), on (yellow box, table), on (red box, table), on (pink box, table), on (L-shaped tool, table), on (bin, table)</b>  |

[Table 2] Primitives of the action library and prompt output according to the task sequence mapped to the results of Table 1 and the primitives

|                  |                                                                     |                           |
|------------------|---------------------------------------------------------------------|---------------------------|
| Action primitive | pick (object), place (object, object), pull (object, L-shaped tool) |                           |
| Task sequence    | 1                                                                   | pick (blue box)           |
|                  | 2                                                                   | place (blue box, bin)     |
|                  | 3                                                                   | pick (yellow box)         |
|                  | 4                                                                   | place (yellow box, table) |
|                  | 5                                                                   | pick (green box)          |
|                  | 6                                                                   | place (green box, bin)    |



[Fig. 2] Total framework for Task and Motion planning. Actions of High-level task sequence are mapped to motions and feasibility is checked with motion planner in Low-level. If an action fails, feedback is sent to the High-level to request an alternative sequence

2.1절에서 도출된 선행조건과 환경 상태에 대해 LLM이 일관성 있게 처리할 수 있는 텍스트 기반의 구조화된 prompt로 변환되어야 한다. 이때 Prompt engineering은 구조적 설계를 제공하게 된다. 이는 환경 제약 조건을 반영하여 계획 생성이 가능하도록 유도하는 제약 조건이 된다. [Table 1]은 [Fig. 1]에서 제시된 상황을 기반으로 Prompt engineering를 적용한 결과를 보여준다. 주어진 명령에 따라 객체 목록, 객체간 상관 관계 그리고 작업 목표가 구조적으로 표현된다. [Fig. 1]에서 green box는 yellow box 밑에 있는 stack된 구조이기 때문에 미리 추론된 객체간 상관 관계를 통해 보완된 작업 목표가 정의된다.

생성된 prompt 출력은 Action library와 직접 매핑될 수 있는 형태로 설계된다. 본 논문에서는 로봇이 수행 가능한 동작 집합을 기반으로 Action library를 정의하였으며, 기본적으로 {pick, place, pull} 세 가지 원시 동작(primitive actions)을 포함한다. 만약 추가적인 action이 요구될 경우 Action library와 prompt를 iterative refinement를 통해 점진적으로 업데이트함으로써 가능한 작업 범주를 확장할 수 있다<sup>[19]</sup>. [Table 2]는 로봇이 수행 가능한 action과 [Table 1]에서 구조화된 출력을 Action library와 매핑하여 LLM이 생성한 작업 시퀀스(Task sequence)를 예시로 보여준다. [Fig. 1]에서 blue box는 한번에 이동 가능하지만 green box는 yellow box가 이동되어야만 작업이 가능하다. 이와 같이 기존 정의된 객체간 상관관계와 작업 목표를 기반으로 작업 시퀀스가 도출된다. 이와 같이 생성된 작업 시퀀스는 TAMP 프레임워크 내에서 High-level의 Task 계획 역할을 하며 이후 단계에서 Low-level의 Motion 계획과 결합되어 실제 실행 가능한 계획으로 확장된다.

LLM 기반 출력은 종종 논리적 일관성이나 로봇의 물리적 수행 가능성을 충분히 고려하지 못하는 경우가 발생한다. 이를 보완하기 위해 본 논문에서는 Few-Shot Prompting을 적용한다. 즉, 유사한 작업의 예시를 Prompt에 포함시켜 LLM이 목표 형식의 작업 계획을 학습된 패턴에 맞추어 생성하도록 유도함으로써 출력의 일관성과 성능을 동시에 향상시킨다<sup>[20]</sup>.

이와 같이 LLM을 기반으로 한 Prompt engineering은 단순한 명령어 해석을 넘어 맥락 정보와 환경 상태를 반영한 작업 계획

을 생성할 수 있도록 한다. 따라서 Prompt engineering은 선행조건 추론 단계와 Action library 매핑 단계 사이를 연결하는 핵심적 매개체로 기능하며, 전체 프레임워크의 안정성과 신뢰성을 높이는 데 기여한다.

[Algorithm 2]는 Prompt engineering을 통해 작업 계획을 출력한다. 먼저 [Algorithm 1]에서 정의된 자연어 명령( $u$ ), 구조화된 맥락( $C$ ), 선행조건( $P$ )을 prompt에 입력한다(line 1). 그리고 출력의 일관성을 유지하기 위해 Action library( $A_{lib}$ )에 포함된 primitives 기반으로 Few-Shot prompting를 수행한다(line 2). 생성된 prompt는 LLM에 입력하여 최초의 작업 계획( $\pi_H = \{s_1, s_2, \dots, s_k\}$ )을 생성한다(line 3). 그리고 생성된 작업 계획에 포함된 subtask( $s$ )를 Action library에 기반하여 수행할 액션( $a$ )으로 매핑하고(line 6), 매핑된 액션이 맥락과 선행조건에 만족하지 않는다면 prompt를 업데이트 하고 출력하여 최초 작업 계획( $\pi_H$ )이 다시 생성할 수 있도록 한다(line 7-10). 반대로 액션이 수행가능하다면 맥락과 파라미터 바인딩을 수행하여 최종 작업 계획 집합( $\pi_A$ )에 저장한다(line 11-12).

#### [Algorithm 2] Prompt Engineering for Task Planning

**Input:** Natural language command  $u$ , Structured context  $C$ , Precondition set  $P$ , Action library  $A_{lib}$

**Output:** Executable task plan  $\pi_A$

```

1: prompt.task  $\leftarrow$  SUMMARIZE_TASK( $u, C, P$ )
2: prompt.examples  $\leftarrow$  FEW_SHOT( $A_{lib}$ )
3:  $\pi_H \leftarrow$  LLM_PLAN(prompt)
4:  $\pi_A \leftarrow \emptyset$ 
5: for  $s$  in  $\pi_H$  do
6:    $a \leftarrow$  MAP_TO_ACTION( $s, A_{lib}$ )
7:   if not FEASIBILITY( $a, C, P$ ) then
8:     prompt  $\leftarrow$ 
       ADD_CONSTRAINT_FEEDBACK(prompt,  $a$ )
9:   return PLAN_WITH_PROMPTS( $u, C, P, A_{lib}$ )
10: end if
11:  $a \leftarrow$  PARAM_BIND( $a, C$ )
12:  $\pi_A.append(a)$ 
13: end for
14: return  $\pi_A$ 

```

### 3.3 통합 프레임워크

제안된 LLM/VLM 기반의 High-level Task 계획은 작업 시퀀스(Task sequence)의 논리적 일관성을 보장하지만 실제 로봇의 물리적 구속 조건(kinematics, dynamics, collision, reachability 등)을 직접 반영하지 않으므로 실제 환경에서의 작업 성공 여부를 보장하지 못한다. 따라서 본 논문에서는 논리적 타당성과 물리적 실행 가능성을 동시에 확보하기 위해 High-level Task 계획과 Low-level Motion 계획을 통합하는 프레임워크를 제안한다.

[Fig. 2]는 본 논문에서 제안하는 통합 프레임워크를 나타낸다. 2.1와 2.2절에서 도출된 Task sequence가 Low-level로 전달되면 먼저 sequence의 각 action에 대해 로봇 End-effector의 6-DoF pose를 이를 action과 매핑한다. Task sequence를  $\pi = \{a_1, a_2, \dots, a_n\}$ ,  $a_i \in A_{lib}$  라 할 때, Action-Pose 매핑 함수는 다음과 같이 정의된다.

$$f: a_i \rightarrow \tau_i \quad (1)$$

여기서  $\tau_i$ 는  $a_i$ 에 대응하는 End-effector의 6-DoF pose를 의미한다.

다음으로 매핑된 pose는 Motion planner를 통해 실행 가능성이 검증된다. 이 과정에서 로봇의 기구학적 한계, 물체의 위치, 충돌 여부 등을 고려한다. 만약 특정 pose가 실행 불가능하다면 해당 action과 실패原因的 구속 조건을 High-level로 피드백하여 주어진 Task sequence에서 실패한 action  $a_i$ 를 포함한 이후 actions를 삭제하고 실패 지점부터 Task sequence를 수정하는 재탐색(Replanning)을 수행한다. 이는 이미 실행 가능성이 검증된 actions는 저장하고 실패한 상황부터 재탐색을 수행하기 때문에 가능한 Task sequence에 빠르게 수렴할 수 있도록 한다.

최종적으로 검증된 모든 motion을 병합하여 실제 환경에서 실행 가능한 최종 로봇 계획을 생성한다. 즉, 최종 계획  $\pi_E$ 는 하위 모든 pose에 대한 motion이 실행 가능하도록 다음을 만족해야 한다.

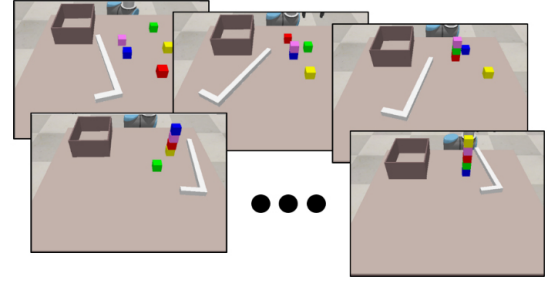
$$\pi_E = \{\tau_1, \tau_2, \dots, \tau_n\}, \forall i, \text{Feasibility}(\tau_i) = \text{True} \quad (2)$$

이와 같은 절차를 통해, 논리적으로 타당하면서도 실제 환경에서 실행 가능한 통합 작업 계획을 산출할 수 있다.

## 4. 실험

### 4.1 실험 환경 세팅

실험은 3D 동적 시뮬레이터 CoppeliaSim에서 수행되었으며, 로봇 플랫폼으로는 Universal Robots UR10을 사용하였다. LLM/VLM은 GPT-4.x를 기반으로 하였으며, 모션 계획에는 Open



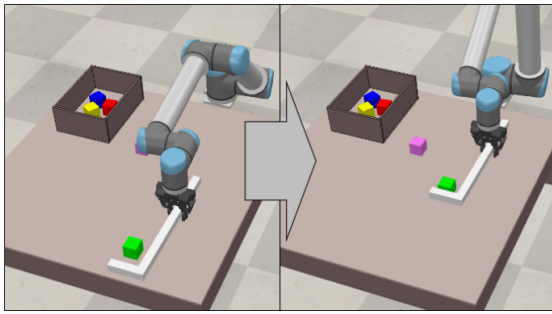
[Fig. 3] Examples of simulation environments

Motion Planning Library (OMPL)에서 제공되는 RRT-Connect 알고리즘을 적용하였다<sup>[21]</sup>. VLM 입력으로는 테이블 전체가 보이는 시점에서 획득한 카메라 뷰를 사용하였다. 작업 환경은 [Fig. 3]과 같이 5개의 box, L-shaped tool, bin, 그리고 table으로 scene을 구성하였다. 테이블 내 boxes를 무작위로 배치하였으며, 2개에서 5개까지의 박스가 쌓인(stack) 상태를 포함하여 총 100개의 scene을 생성하였다. 실험에서 로봇에게 주어진 명령은 다음과 같다: “Put all boxes into the bin in the following order: red, blue, yellow, green, and pink.”

### 4.2 High-level planner 성능 실험

제안된 프레임워크의 성능을 평가하기 위해 앞서 GPT-4, GPT-4o, GPT-4.5를 활용하여 High-level에서의 작업 계획 성능을 비교하였다. 구체적으로 작업 생성 성공률(Success rate), 생성된 계획의 길이(Plan length), 작업 생성 소요 시간(Response time)을 측정하였다. [Table 3]은 각 방법에 대한 평균을 비교한 결과이다.

[Table 3]에 나타난 바와 같이, 모든 지표에서 GPT-4.5가 가장 우수한 성능을 보였다. Success rate 측면에서 실패 사례는 주로 box가 stack된 상황에서 발생했는데 이는 VLM이 box 간의 stack 관계를 정확히 인식하지 못하여 아래에 위치한 box를 먼저 pick하는 오류가 생성된 경우였다. 특히 GPT-4의 경우 VLM을 통해 stack 관계가 프롬프트 상에 명확히 정의되었음에도 불구하고, 하위 물체를 먼저 조작하려는 논리적 오류가 반복적으로 발생하였다. 반면, GPT-4.5는 본 논문에서 제안한 맥락 기반 선행조건 추론([Algorithm 1])을 통해 구조화된 환경의 기호적 제약조건(예: on (yellow box, green box))을 정확히 해석하여 물리적 간섭이 없는 최적의 작업 순서를 도출하였다. 이는 본 프레임워크가 제공하는 구조화된 맥락 정보가 추론 능력이 뛰어난 언어 모델과 결합될 때, 복잡한 비정형 환경(Stacking 등)에서도 강력하고 안전한 작업 계획을 도출하는 핵심 기반이 됨을 보여준다. Plan length는 stack이 없는 경우에는 모든 모델이 동일하게 10의 길이를 기록하였다. 그러나 stack이 존재하는 경우 GPT-4에서 불필요한 동작이 추가되어 Task sequence



[Fig. 4] An example of a pulling action performed to move a box located outside the robot's workspace into the workspace. When a failure occurs at the low level, the cause is re-reasoned at the high level to regenerate the task plan

의 길이가 증가하는 현상이 나타났으나, 모델 성능이 향상될수록 프레임워크의 제약 조건을 최적화하여 가장 짧은 Task sequence에 수렴하는 결과를 확인하였다.

#### 4.3 통합 프레임워크 성능 실험

제안된 프레임워크의 실제 로봇 적용 가능성을 검증하기 위해 최종 생성된 작업 계획에 대해 작업 생성 성공률(Success rate), 작업 수행 성공률(Execution S/R) 생성된 계획의 길이(Plan length), 재탐색 횟수를 측정하였다. 작업 수행 실패에는 작업 생성 실패도 포함되며, 무한 탐색을 방지하기 위해 재탐색 횟수는 최대 5회로 제한하였다. [Table 4]는 각 방법에 대한 비교 결과이다.

전체 프레임워크에서의 작업 생성 성공률은 High-level 결과보다 낮게 나타났으며 특히 GPT-4의 경우 차이가 두드러졌다. 주요 원인은 High-level에서 작업 생성이 실패할 경우 재탐색을 수행하게 되는데 제한된 재탐색 횟수를 초과하면 실패로 간주되기 때문이다. [Table 3]과 같이 GPT-4는 High-level에서 도출한 계획이 실제 물리적 제약(Reachability, Collision 등)을 위반하는 경우가 많아 Low-level 피드백에 의한 재탐색 횟수가 평균 3.7회로 급격히 증가하였다. 이러한 재탐색 과정은 본 논문이 제안하는 통합 프레임워크의 필수적인 안정성 보장 기제로 작용한다. LLM은 로봇의 정밀한 기구학적 한계를 온전히 이해할 수 없으나 본 프레임워크는 Low-level의 실패 원인을 High-level로 피드백함으로써 언어 모델의 실행 불가능한 동작 생성을 효과적으로 억제하고 물리적으로 실행 가능한 대안을 재탐색하도록 유도한다.

또 다른 실패 요인은 로봇의 물리적 구속 조건에 의해 발생한다. 일부 scene의 경우 특정 box가 로봇의 워크스페이스 밖에 위치하였는데 이러한 경우는 High-level 계획만으로는 대응이 어렵다. Low-level 단계에서 해당 실패 원인을 인식하고 재탐색을 수행하지만 [Fig. 4]에서 보이는 것처럼 L-shaped tool을 사용하여 워크스페이스 밖에 있는 box를 끌어오는 pull action

[Table 3] Results of High-level output evaluation

|               | GPT-4 | GPT-4o | GPT-4.5 |
|---------------|-------|--------|---------|
| Success rate  | 84%   | 87%    | 93%     |
| Plan length   | 16.4  | 15.4   | 14.2    |
| Response time | 16.2s | 3.4s   | 2.8s    |

[Table 4] Results of simulation experiment

|               | GPT-4 | GPT-4o | GPT-4.5 |
|---------------|-------|--------|---------|
| Success rate  | 71%   | 79%    | 89%     |
| Execution S/R | 52%   | 70%    | 82%     |
| Plan length   | 21.1  | 19.3   | 17.7    |
| Replanning    | 3.7   | 2.8    | 2.7     |

(pull (box, L-shaped tool))을 추가로 추론하지 못하면 작업 생성은 실패하게 된다. 본 연구에서는 LLM이 이러한 동작을 추론할 수 있도록 Few-Shot Prompting을 통해 관련 예시를 Prompt에 포함시켜 작업을 생성할 수 있도록 하였다. 이는 제안된 프레임워크의 Action-Pose 매핑이 단순한 Pick, Place를 넘어 도구 사용과 같은 복잡한 물리적 맥락까지 확장될 수 있음을 시사한다. 워크스페이스 이탈이라는 Low-level의 실패가 피드백되면 Few-Shot 예시를 바탕으로 액션을 동적으로 계획에 삽입하여 문제를 해결하는 것은 기호적 추론과 물리적 실행이 결합된 본 TAMP 프레임워크만의 주요한 이점이다.

작업 수행 성공률은 계획 길이(Plan length)에 큰 영향을 받았다. 계획이 길수록 실행 과정에서 발생하는 동적 오차(dynamic error)가 누적되어 작업 도중 실패로 이어질 가능성이 높았다. 특히, 특정 길이를 초과하는 시점에서 성공률이 급격히 떨어지는 현상이 관찰되었다. 예를 들어, GPT-4 대비 GPT-4.5는 작업 길이가 평균 16% 짧은 반면 작업 생성 성공률은 18% 높은데 작업 수행 성공률은 무려 30% 향상된 결과를 보였다. 이러한 결과는 작업 수행 성공률이 High-level 단계의 작업 생성 성공률 뿐만 아니라 작업 계획의 길이에도 크게 영향을 받는다는 사실을 실험적으로 확인시켜 준다. 따라서 향후 연구에서는 논리적 타당성을 유지하면서 불필요한 동작을 최소화하는 단축 계획 생성 전략을 적용함으로써 제안된 프레임워크의 신뢰성과 효율성을 한층 더 향상시킬 수 있을 것으로 기대된다.

## 5. 결 론

본 논문에서는 거대 언어 모델과 비전 언어 모델을 활용하여 불완전한 명령과 환경 불확실성을 극복하는 통합 TAMP 프레임워크를 제안하였다. 제안된 방법은 시각적 맥락 기반의 선행 조건 자율 추론을 통해 로봇의 고수준 작업 계획을 생성하고 이를 저수준 모션 계획의 기구학적 피드백과 결합하여 언어 모델의 물리적 실행 한계를 성공적으로 보완하였다. 시뮬레이션 검증을

통해 동적 스택 구조나 워크스페이스 이탈과 같은 비정형 상황에서도 불필요한 동작을 최소화하며 높은 성공률을 달성함을 확인하였다. 향후 연구에서는 실제 로봇 구동 환경에서 발생하는 센서 노이즈 및 동적 장애물에 대한 실시간 대응 전략을 고도화하고 본 프레임워크의 범용적인 산업적 확장성을 검토할 예정이다.

## Appendix

본 논문에서 사용된 Prompt는 다음과 같다.

### Prompt to analyze context-based precondition inference

You are a robot task planner.

Analyze the following command and environment, and output in the required format.

1. Objects: list all relevant objects.
2. Object relationships: describe spatial relations using predicates on(obj1, obj2).
3. Goal state: describe desired state after execution using predicates.

Always output in the following format:

Objects: [...]

Object relationships: [...]

Goal state: [...]

// for Few-Shot prompting

Example 1:

Command: Put the box into the bin.

Objects: box, bin, table

Object relationships: on(box, table), on(bin, table)

Goal state: in(box, bin), on(bin, table)

Example 2

...

### Prompt to build task plan

Convert the following analyzed environment into a primitive action sequence.

Only use the following primitives:

- pick(object)
- place(object, object)
- pull(object, L-shaped tool)

Output Format: Output actions only, one per line, no explanations.

// for Few-Shot prompting

Example 1

Command: Put the box into the bin.

Objects: box, bin, table

Object relationships: on(box, table), on(bin, table)

Answer:

pick(box)

place(box, bin)

Example 2

...

## References

- [1] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," *IEEE International Conference on Robotics and Automation*, Shanghai, China, pp. 1470-1477, 2011, DOI: 10.1109/ICRA.2011.5980391.
- [2] 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, DOI: 10.1146/annurev-control-091420-084139.
- [3] M. Toussaint, "Logic-Geometric Programming: An Optimization-Based Approach to Combined Task and Motion Planning," *International Joint Conferences on Artificial Intelligence*, Buenos Aires, Argentina, pp. 1930-1936, 2015, DOI: 10.5555/2832415.2832517.
- [4] W. Vega-Brown and N. Roy, "Asymptotically optimal planning under piecewise-analytic constraints," In *Algorithmic Foundations of Robotics XII: Proceedings of the Twelfth Workshop on the Algorithmic Foundations of Robotics*, Cham, Switzerland: Springer International Publishing, 2020, pp. 528-543, DOI: 10.1007/978-3-030-43089-4\_34.
- [5] J. Lee, U. Rakhman, C. Nam, S. Kang, J. Park, and C. Kim, "High dimensional object rearrangement for a robot manipulation in highly dense configurations," *Intelligent Service Robotics*, vol. 15, no. 5, pp. 649-660, Nov. 2022, DOI: 10.1007/s11370-022-00444-4.
- [6] S. Lee and I. Kim, "Task and motion planning for grasping obstructed object in cluttered environment," *The Journal of Korea Robotics Society*, vol. 14, no. 2, pp. 104-113, June 2019, DOI: 10.7746/jkros.2019.14.2.104.
- [7] S. Byeon and Y. Oh, "Task Planning Algorithm with Graph-based State Representation," *The Journal of Korea Robotics Society*, vol. 19 no. 2, pp. 196-202, 2024, DOI: 10.7746/jkros.2024.19.2.196.
- [8] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," In *International conference on machine learning*, Baltimore, USA, pp. 9118-9147, 2022, DOI: 10.48550/arXiv.2201.07207.
- [9] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "ProgPrompt: Program generation for situated robot task planning using large language models," *Autonomous Robots*, vol. 47, no. 8, pp. 999-1012, Aug. 2023, DOI: 10.1007/s10514-023-10135-3.
- [10] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, and A. Zeng, "Do as I can, not as I say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022, DOI: 10.48550/arXiv.2204.01691.
- [11] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, and K. Han, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," In *Conference on Robot Learning*, Atlanta, USA, pp. 2165-2183, 2023, DOI: 10.48550/arXiv.2307.15818.

- [12] P. Sermanet, T. Ding, J. Zhao, F. Xia, D. Dwibedi, K. Gopalakrishnan, and Y. Cao, "RoboVQA: Multimodal long-horizon reasoning for robotics," *International Conference on Robotics and Automation*, Yokohama, Japan, pp. 645-652, 2024, DOI: 10.48550/arXiv.2311.00899.
- [13] G. Cheng, C. Zhang, W. Cai, L. Zhao, C. Sun, and J. Bian, "Empowering large language models on robotic manipulation with affordance prompting," *arXiv preprint arXiv:2404.11027*, 2024, DOI: 10.48550/arXiv.2404.11027.
- [14] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg, "Text2motion: From natural language instructions to feasible plans," *Autonomous Robots*, vol. 47, no. 8, pp. 1345-1365, Nov. 2023, DOI: 10.1007/s10514-023-10131-7.
- [15] M. Kwon and Y. J. Kim, "Neuro-symbolic task replanning using large language models," *The Journal of Korea Robotics Society*, vol. 20, no. 1, pp. 52-60, Feb. 2025, DOI: 10.7746/jkros.2025.20.1.052.
- [16] Y. Yin, K. Wan, C. Li, and P. Zheng, "An LLM-enabled human demonstration-assisted hybrid robot skill synthesis approach for human-robot collaborative assembly," *CIRP Annals*, vol. 74, no. 1, pp. 1-5, 2025, DOI: 10.1016/j.cirp.2025.04.015.
- [17] A. Cangelosi, "Grounding language in action and perception: From cognitive agents to humanoid robots," *Physics of life reviews*, vol. 7, no. 2, pp. 139-151, June 2010, DOI: 10.1016/j.plrev.2010.02.001.
- [18] R. Mon-Williams, G. Li, R. Long, W. Du, and C. G. Lucas, "Embodied large language models enable robots to complete complex tasks in unpredictable environments," *Nature Machine Intelligence*, vol. 1, no. 10, 2025, DOI: 10.1038/s42256-025-01005-x.
- [19] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegrefe, and P. Clark, "Self-refine: Iterative refinement with self-feedback," *Advances in Neural Information Processing Systems*, vol. 36, New Orleans, USA, pp. 46534-46594, 2023, DOI: 10.5555/3666122.3668141.
- [20] L. Reynolds and K. McDonell, "Prompt programming for large language models: Beyond the few-shot paradigm," In *Extended abstracts of the 2021 CHI conference on human factors in computing systems*, Yokohama, Japan, pp. 1-7, 2021, DOI: 10.1145/3411763.345176.
- [21] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics & Automation Magazine*, vol. 19 no. 4, pp. 72-82, 2012, DOI: 10.1109/MRA.2012.2205651.



## 이진휘

2016 한양대학교 기계공학부(학사)

2023 한양대학교 기계공학과(박사)

2023~2024 한국과학기술연구원 박사후연구원

2024~현재 한국공학대학교 메카트로닉스  
공학부 조교수

관심분야: Autonomous system, Robotic Manipulation, Service robotics