BEYOND LANGUAGE: INTEGRATING LLM COMMONSENSE AND TASK PLANNING FOR ROOM RE-ARRANGEMENT

Zhanxin Wu [zhanxinwu@u.nus.edu], **Nguyen Hoang Bao Dai** [baodai@u.nus.edu]
National University of Singapore (NUS)

Demo time!

Motivation

Do you want a personal robot housekeeper?

In this project, we seek to endow robots with the capability of tidying up a room, given only partial textual description of the layout from humans. This task embodies three significant challenges:

- Insufficient information for navigation due to **partial map description**
- Notion of tidiness requires **commonsense** understanding not available in existing systems
- Re-arranging the room requires a long action sequence with **combinatorial complexity**

To this end, we propose a novel framework that integrates classical task planning and modern Large Language Models (LLMs) to achieve robust room re-arrangement. To the best of our knowledge, we are the first to demonstrate such capabilities!

Approach Overview

Our strategy is to exploit the knowledge LLMs learn from language data at three levels:

- **Spatial layout understanding**: Construct complete map representations from human partial text description of the environment
- **Commonsense reasoning**: Generate high-level task plans for re-organizing objects
- **Programming and control skills**: Refine high-level task plans to obtain executable action plans

Proposed framework:
Given human partial description of the map ***T***, language description of the scene ***S***, and out-of-place objects ***O***, we perform the following:

- ***Stage 1: Recover full graphical map representation G with partial language description T***
  $$G = LLM(T, P_1)$$
  where P_1 is a stage-specific prompt.
- ***Stage 2: Generate high-level task plan TP***
  $$\mathrm{TP} = LLM(G, T, P_2)$$
  where P_2 is a stage-specific prompt.
- ***Stage 3: Generate executable action plan AP***
  $$\mathrm{AP} = LLM(TP, P_3)$$
  where P_3 is a stage-specific prompt.

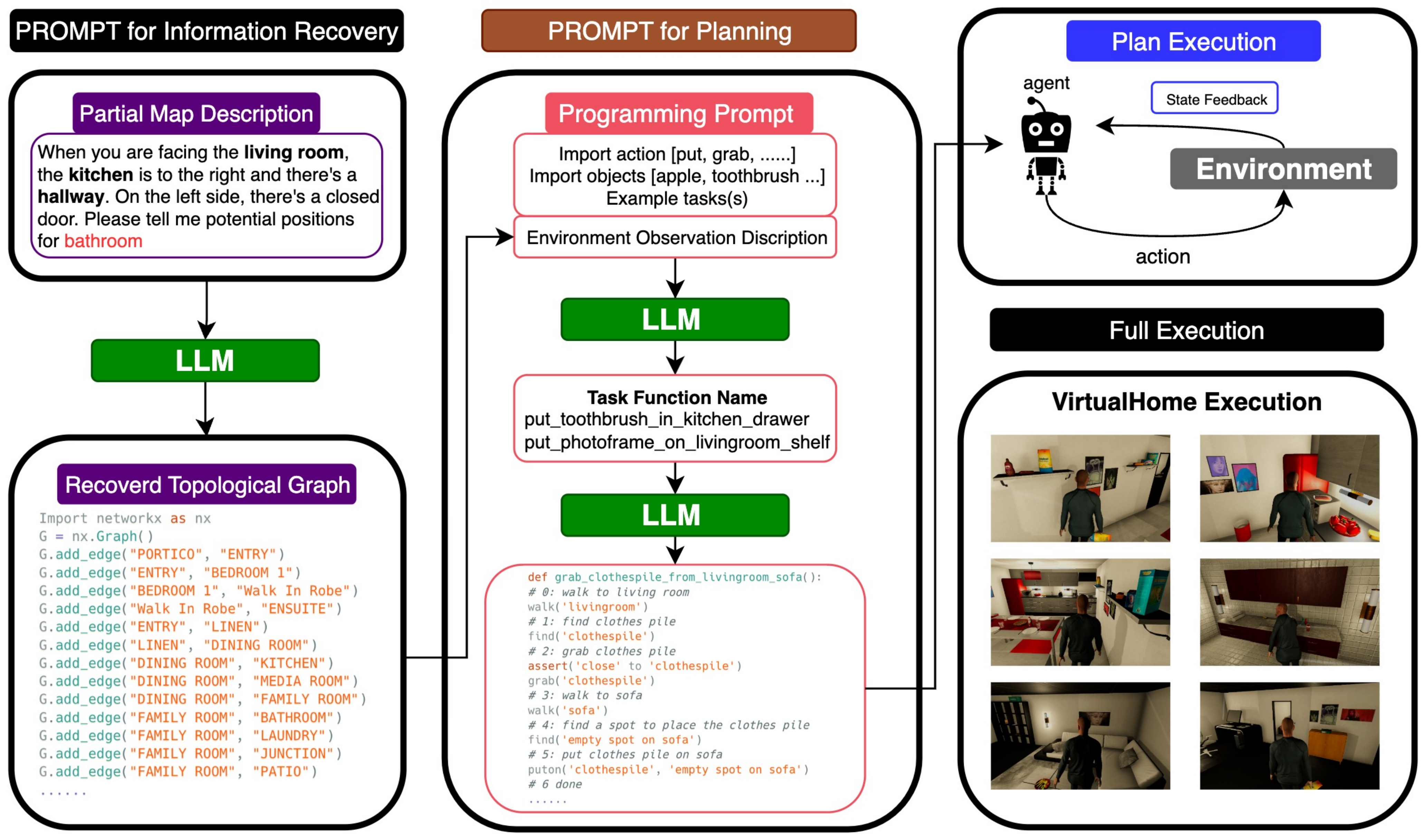
Finally, AP is executed by the robot.

Importantly, we use **Python** as the language to represent the map G, high-level task plans TP, and action plan AP, because of four key reasons:

- Programming languages inherent have **procedural structures** well suited to describe the solution
- Programming languages are **directly executable** by the agent, which sidesteps the difficulty of converting natural language solutions to machine-understandable language (i.e., code)
- Programming languages are inherently combinatorial, enabling **combinatorial generalization**
- LLMs have relative **strengths** in generating code representations

System Architecture

Below is the system architecture that implements the abovementioned 3 stages.



Results and Discussions

Stage 1: Map recovery from partial descriptions

Map Representation	Natural Language Description	Pythonic Map with Coordinates	Pythonic Topological Map
Success Rate	80%	20%	100%
Remarks	The output contains complex expressions that make it difficult to extract keywords.	It consistently outputs a specific room and lacks the ability to identify the spatial layout.	Structured programming prompts aid LLM in comprehending the spatial layout.

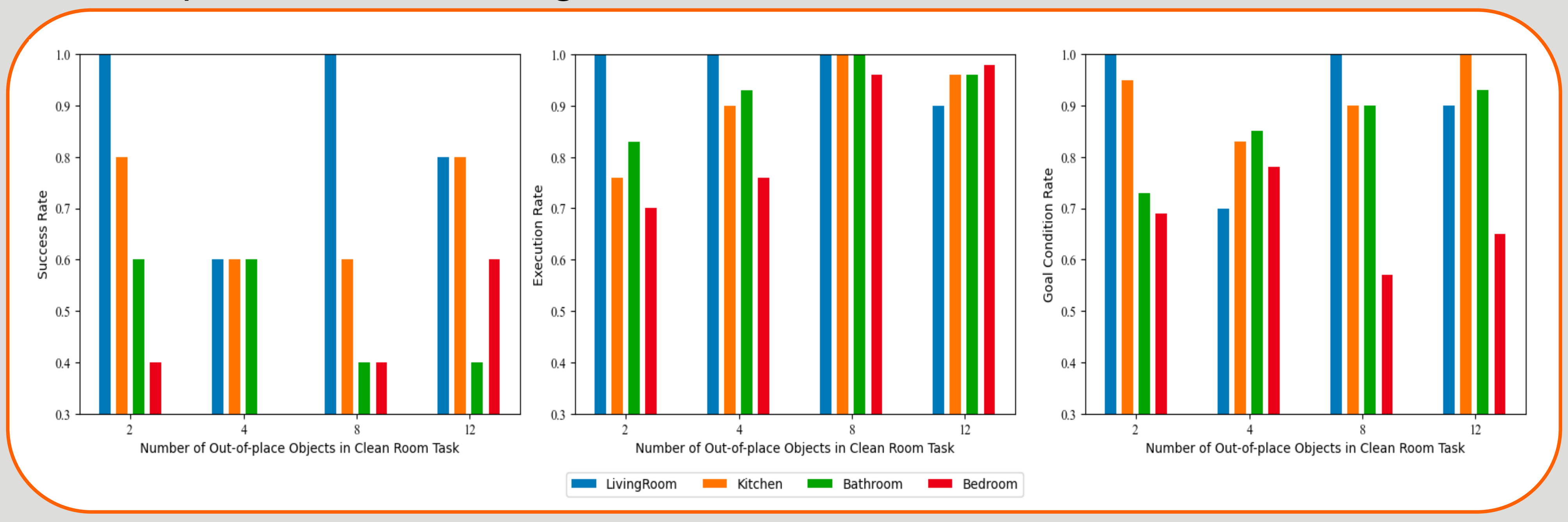
Findings:

- LLMs embody commonsense understanding of spatial layouts of human environments
- Graph-based map representations sidestep the difficulty of predicting accurate metric locations on the map
- Programming languages enable LLMs to better describe the world

Stage 2 & 3: Generate high-level task plans and low-level action plans

Metrics:

- Success Rate: the fraction of executions that achieved goal-conditions
- Execution Rate: the fraction of actions in the plan that are executable in the environment, even if they are not relevant for the task.
- Goal Condition Rate: the fraction of the desired outcome that the plan accomplishes in the actual goal states.



Findings:

- Programming prompts combine LLM's strengths in reasoning and code understanding
- Using programming prompts with examples helps to ground the actions into the agent's capabilities
- However, LLM performance decreases as item size increases, resulting in mistakes such as repeating meaningless actions. This suggests the difficulty of scaling up.
- LLMs demonstrate unstable performance with reasoning about the task plans.