2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Autonomous Behavior Planning For Humanoid Loco-manipulation Through Grounded Language Model

Jin Wang[12*], Arturo Laurenzi[1], Nikos Tsagarakis[1]

***Abstract*— Enabling humanoid robots to perform autonomously loco-manipulation in unstructured environments is crucial and highly challenging for achieving embodied intelligence. This involves robots being able to plan their actions and behaviors in long-horizon tasks while using multi-modality to perceive deviations between task execution and high-level planning. Recently, large language models (LLMs) have demonstrated powerful planning and reasoning capabilities for comprehension and processing of semantic information through robot control tasks, as well as the usability of analytical judgment and decision-making for multi-modal inputs. To leverage the power of LLMs towards humanoid loco-manipulation, we propose a novel language-model based framework that enables robots to autonomously plan behaviors and low-level execution under given textual instructions, while observing and correcting failures that may occur during task execution. To systematically evaluate this framework in grounding LLMs, we created the robot 'action' and 'sensing' behavior library for task planning, and conducted mobile manipulation tasks and experiments in both simulated and real environments using the CENTAURO robot, and verified the effectiveness and application of this approach in robotic tasks with autonomous behavioral planning. Video: `https://youtu.be/mmnaxthEX34`**

I. INTRODUCTION

Maintaining autonomy during the execution of a task in a real-world environment is both essential and challenging for robots, especially when performing tasks that require interaction with surroundings and manipulation of objects. This demands a high level of capability from robots that have to perceive and make decisions during the task execution and the ability to achieve autonomous planning based on these decisions. Furthermore, one of the main challenges lies in enabling robots to understand semantic instructions from humans and apply them within the context of different scenarios and their current state. This process involves encoding textual information into a hierarchical sequence of robot behaviors, as well as mapping high-level tasks to low-level robot control and generating reference trajectories that the robot can execute.

Integrating large language models (LLMs) has emerged as a promising avenue for enhancing the autonomy of robots. These models have demonstrated great flexibility in understanding and processing semantic information, along with remarkable reasoning and decision-making capabilities.

†This work was supported by the European Union's Horizon 2020 research and innovation programme, euROBIN EPUE034001, and Leonardo Joint Lab JL Leonardo ETCM058501.

[1]Humanoids and Human-Centered Mechatronics (HHCM), Istituto Italiano di Tecnologia, Via Morego 30, Genoa, 16163, Italy.
[2]DIBRIS, Universita di Genova, Italy, 16145.
*Corresponding author: `wang.jin@iit.it`

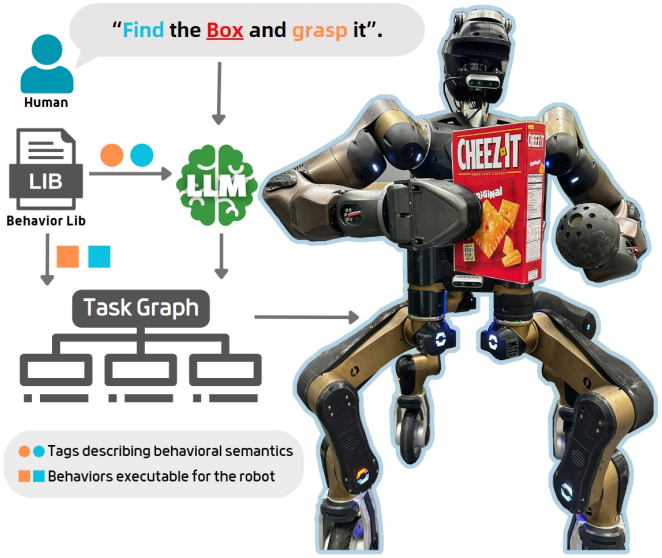

Fig. 1: Humanoid robot CENTAURO picks objects with the planning of the *task graphs* generated by the LLM. The '*behavior lib*' consists of various action and sensing behaviors with 'tags' describing the semantic content of different behaviors.

However, due to the complexity of whole-body control and perceptual decision-making in humanoid robots, it is challenging to directly apply LLMs for action generation and planning, especially when it involves understanding the current task based on environmental cues and interacting with objects. Previous work has shown that incorporating language models into robotic tasks [1] and enabling intelligence to better reason and evaluate textual information can interact with the environment to accomplish long-horizon tasks that require complex planning of robot action sequences. Leveraging this feature, we design a language model-based planner, which requires the pre-creation of a *behavior lib* for the robot, including multiple actions and perceptions for different modalities. This can be used to generate direct reference trajectories for execution. After acquiring the human instruction and the semantic indices from the *behavior lib*, the LLM generates hierarchical *task graphs*, which guide the robot to follow the logical sequence of tasks and make decisions according to different scenarios.

Finally, the increasing autonomy of robots during the execution of complex tasks, as well as unexpected perturbations in long-term missions, make failures during the execution of

high-level behavioral sequences in real-world environments inevitable. Therefore, failure detection as well as correction procedures, are needed to ensure alignment between low-level execution and high-level semantic planning during the programming of complex loco-manipulation tasks that are composed of multiple subtasks. We attempt to fuse multimodal sensor data and add them as perceptual behaviors into the robot *behavior lib*, which enables the LLM to select the optimal combination of behaviors according to the task scenario. Among them, the visual language model (VLM) is used as one of the key metrics for determining whether the task is completed or not as well as for failure determination, due to its accurate and effective image information comprehension and reasoning capabilities. Once the VLM perceptual behavior detects a failure during a task, it triggers the behavioral planner to perform the corrective action. This process is pre-planned by the LLM and stored in the *task graphs*, and by combining the perceptual behavior of different modalities, a closed loop of high-level feedback is realized, which improves the robustness of the autonomous robotic system and increases the task success rate.

In this paper, we present a language model-based framework enabling the autonomous execution of loco-manipulation tasks. This framework leverages the semantic understanding capabilities of LLM in reaction to human instructions, which is based on the knowledge of the task scenario and the behavioral skills ("action" and "perception") possessed by the robot. As shown in Figure 1, when a human provides the instruction "find the bottle and pick it up", the LLM will generate a *task graphs* consisting of different behavioral nodes based on the instruction and the behavior lib as prompts, which will guide the CENTAURO robot [2] to perform the actions such as object detection, object grasping, and lifting, etc. While the VLM-based perceptual behavior node is used as a failure detector, it's triggered when the state of the current task is required to be detected and makes an inferential judgment based on the images returned from the robot onboard camera.

Our summary of the main contributions of this work includes:

- We exploit the appropriateness of LLM in loco-manipulation tasks and propose a novel framework for autonomous behavior planning, enabling rapid deployment without additional training, which can be applied to quadrupeds, humanoids, and mobile manipulators.
- We propose a new paradigm for robot behavior library, which enables to encoding of human instruction into optimal action sequences by combining behaviors in a modular way and linking high-level tasks to interpretable low-level control.
- We incorporate multimodal failure detection as higher-order feedback to facilitate the task graph in correcting the misalignment between the intended goal and the actual robot's actions during the execution of the task.

The proposed framework was experimentally validated to assess its feasibility in handling long-horizon tasks in simulation and real-world environments.

II. Related works

Endowing robots with autonomy in locomotion and manipulation tasks still represents a high-level challenge for robotics. In recent years, prior research has focused on motion planning and trajectory optimization [3], covering systems with various morphologies and levels of autonomy. The data-driven approach [4] enables the use of experience to make decisions online and generate appropriate multimodal reference trajectories for dexterous manipulation. Meanwhile, combining data and learning, the model-free reinforcement learning approach [5][6] has demonstrated impressive performance in several specific tasks and unstructured environments. For long-term tasks, [7][8] introduced a novel motion planning framework and task evaluation approach that allowed robots to maintain dexterity while navigating complex environments. Boston Dynamics [9] utilized an offline trajectory optimization and model predictive control strategy to codify the Atlas robot's movements into a time series, achieving consistency between simulation planning and motion execution. However, when faced with different task scenarios and long-horizon planning involving multiple subtasks, the ability to understand instructions and reason about tasks is often overlooked, making it challenging to achieve autonomy and adaptability in task-driven mobile manipulation.

With the emergence of large language models (LLMs), several transformer-based architectural planners [10][11][12] have played a pivotal role in predicting and generating robot actions and attempting to derive robot policy code [13] guided by natural language instructions. However, such end-to-end strategies often require substantial amounts of training data and expert demonstrations, complicating the model's training process, especially in unknown scenarios. In contrast, some approaches leverage LLMs' semantic comprehension capabilities as a higher-level task planner [1][14][15], transforming instructions into executable lower-level actions in a zero- or few-shot manner [16]. Nonetheless, these methods tend to assume the success of the task performed and overlook the potential discrepancy between planned expectations and real-world execution. They often fail to enable whole-body control in multi-joint, high degree-of-freedom (DoF) mobile manipulation tasks, such as those involving humanoid robots. Moreover, studies such as [16][17][18] attempt to interpret textual and visual inputs simultaneously, using them to address downstream robotic tasks. While visual question answering (VQA) [19] can be effectively achieved through understanding image descriptions and inferring the robot's state and current context to guide subsequent actions, relying solely on high-level visual feedback has been proved to be insufficient. It compromises the rapid response to dynamic environments and is less effective than other methods exploring proprioceptive perception, such as interaction force sensing when executing low-level tasks.

Our study, on the other hand, applies the LLM to robots by utilizing its ability to understand instructions and plan

with existing behavior libraries by generating task graphs that sequentially instruct the high DoF humanoid robot's whole-body actions. It detects and recovers from possible failures during tasks by using the VLM as a perceptual behavioral node and combining it with other perceptrons to enable multimodal feedback. This approach allows the robot to serialize discrete action nodes and select different perceptual combinations according to the task scenario. Ultimately, a robust autonomous mobile manipulation skill is realized and demonstrated.

III. Autonomous Robot Behavior Planning

A. Problem Statement

The problem of performing autonomously loco-manipulation task based on higher-level instructions can be described as follows. We assume that the human's semantic description of the task, denoted as i articulates a specific task to be executed by the robot. Additionally, we consider that a *behavior lib* Π is provided to the robot, consisting of a set of action and perception behaviors $\pi \in \Pi$ that can be directly executed by the robot, such as the grasp action would enable the robot to control the gripper for grasping in Cartesian space, while the object detection behavior would utilize the camera to detect the pose of the target object. Each behavior is associated with a specific semantic description l_π (e.g., "open the griper and move to the target pose then close griper"). By invoking the large language model L, the task graph T corresponding to instruction i is generated:

$$T_i = L(i, l_{\pi_1}, l_{\pi_2}, ..., l_{\pi_n}) \tag{1}$$

The task graph T_i, encapsulates a sequential arrangement of behaviors necessary for the robot to execute in various states s to accomplish the designated task requested by the human. This graph is maintained in an XML file format and is operationalized through the Behavior Tree (BT) B. The robot state is the feedback by the execution results of different behavioral nodes, including (running, success, fail). Depending on the current state of the robot, the BT guides the robot to execute different behaviors. When a low-level execution deviation from the task plan is detected, the BT fixes this failure by resuming the behavior in an attempt to correct the error.

The above process is described in Algorithm 1. In this way, the robot is enabled to encode human instructions into various sequences of behaviors and execute them according to the task demands, as well as to recover possible misalignment between the instructions and the robot's execution.

B. Overview of the framework

We propose a robot behavior planning system for performing autonomously loco-manipulation tasks that makes use of language models. Figure 2 shows an overview of the system. We first create a library of behaviors for the CENTAURO robot, divided into action behaviors and perceptual behaviors, each containing a corresponding behavior tag and a behavior code. The behavior tag records the name and type of the behavior, as well as a detailed semantic description of the behavior. In addition, prompts are defined in advance. Prompts include conditional information for input to the large language model, a description of the robot's features, the skills that the robot possesses (*behavior lib*), and a description of the expected outputs. When a robot is called upon to perform a task, the human first gives task instructions as input, which are fed to the large language model together with the prompts, along with the behavior tags from the *behavior lib*. Upon obtaining this information, the LLM generates a sequence comprising the robot's behaviors based on the given task and stores it in an XML-formatted file, which is used to generate a Behavior Tree (BT) [20] that controls the robot's task execution. The behavior code in *behavior lib* and the BT generated by the LLM together form a *task graph*, which instructs the robot to perform different behaviors according to the current robot state and conditions, and ultimately completes the task. The task graph also plans a correction policy for failures during the task. If the perceptual behavior detects an inconsistency in the execution of the task, the task graph will execute specific behaviors to try to fix the failure.

Algorithm 1 Language model based robot behavior planner

Given: Language model L, a human instruction i, and a behavior lib Π with the language description l_π.

T_i = $L(i, l_\pi)$
$B \leftarrow BehaviorTree(T_i)$
Initialize the state s_π, number of steps n
while $B(s_{\pi_n}) \neq$ "$done$" **do**
  for $\pi_n \in \Pi_B$ **do**
    executeBehaviorπ_n
    $s_{\pi_n} \leftarrow updateStatus$
  end for
  if allBehavior Completed **then**
    return Done
  end if
end while

While in the behavior execution phase, the CENTAURO depends on Xbot[21] and Cartesian I/O[22] to execute the action commands issued by the task graph, the current state of the robot, as well as the sensory information, are fed back to the task graph to guide the next action. During this process, the host PC is responsible for behavioral planning, including receiving human instructions and behavior tags, and generating the task graph through the large language model. Meanwhile, the robot pilot is tasked with receiving and executing the low-level action commands.

C. Language Model for Behavior Planning and Modification

1) Behavior Lib: Controlling a robot to accomplish complex actions in a long-horizon task is difficult and challenging. To link semantic behaviors with the actual execution of actions by the robot, we designed a behavior library for the CENTAURO robot such that each skill can directly control

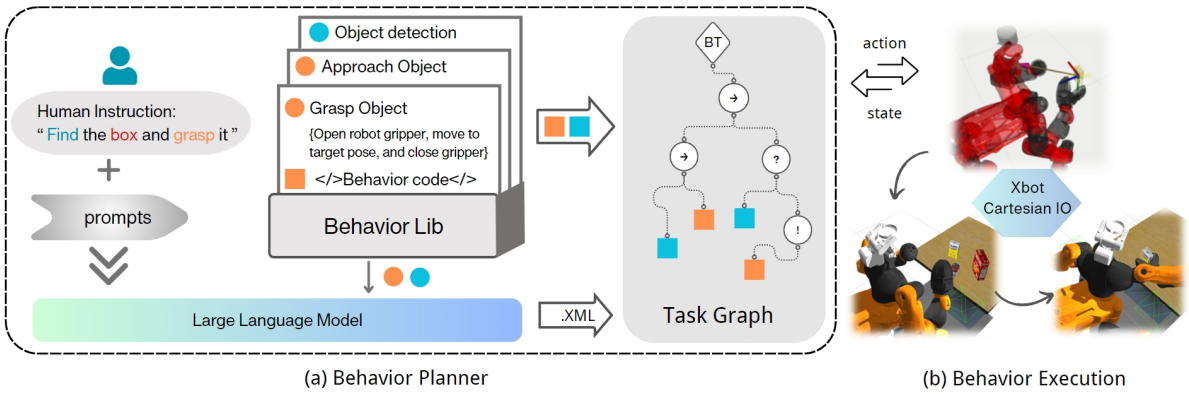

Fig. 2: Overview of the Framework. (a) Behavior Planner takes the human instruction as input, given the behavior lib and prompts, LLM generates a hierarchical structure behavior tree, which forms the task graph along with the behavior code. (b) The CENTAURO robot executes the lower action command and feeds back its current state. The entire process does not require any additional training.

the robot to complete basic actions. This approach improves the interpretability of each step of the task process and reduces the deviation between high-level tasks and low-level execution by partitioning the complex task into a sequence of actions consisting of several behaviors skills.

We classify the *behavior lib* into action behaviors and perceptual behaviors. Action behaviors control the CENTAURO robot to complete whole-body motions such as moving and manipulating. Inspired by object manipulation in daily office environments, we have designed several individual robot actions to compose the action lib using Cartesian I/O[22], including `Approach`, `Grasp`, `Lift`, etc., which are used to implement subtasks such as navigating to various locations, grasping a target object, and lifting a target object, etc. The action lib is a library of robot actions that can be added and combined depending on the demands of a loco-manipulation task and the interaction environment. Meanwhile, perceptual behaviors rely on the robot's internal sensors (torque sensing, RGBD camera) to detect the position of the object, evaluate the robot's state, and reason whether the current task is complete or if there are failures. Various algorithms [23][24] are integrated into different perceptual behaviors, and similar to action behaviors, perceptual behaviors can be designed independently and added to the perceptual lib. Multiple sensors can be invoked in a single perceptual behavior and fused with data from different modalities according to the requirements.

2) LLM Generated Task Planner: While large language models can utilize their extensive knowledge of semantic data as well as their text comprehension reasoning capabilities to provide answers to human instructions, the answers can be diverse. To obtain the desired output, it is necessary to impose constraints on the instructions given as input. One approach is to use prompt words, a linguistic construct designed to qualify a language model to give a specific output. In our framework, prompts are used as input to the large language model along with human instructions and behavior tags, which consist of several components. First, there is information about the current state of the CENTAURO robot and its hardware configuration. Then, the concept of a behavior library, its components, and sample applications are introduced. Lastly, there is the expected output in a format that includes the concept of a behavior tree, the definition of the nodes, and examples of applications.

TABLE I: Behavior Lib Definition

Behavior	Type	Tag
`Homing`	*action*	'bringing all of the joints of robot to homing configuration'
`Approach`	*action*	'moving robot torso closer to target by certain distance'
`Grasp`	*action*	'moving gripper to a given pose and close it'
`Lift`	*action*	'raising gripper to the chest and adjusting pose'
`Place`	*action*	'moving gripper to the given position and open it'
`Distance`	*perception*	'measuring distance between object and robot'
`Grip force`	*perception*	'obtaining the actual torque of gripper'
`Detection`	*perception*	'detecting and estimating 6D poses of objects'
`Visual Q&A`	*perception*	'reasoning task state using visual language model'

To convert high-level instructions into a sequence of implementable low-level skills, we leverage Behavior Trees as both an intermediate bridge and an output of the LLM. The use of Behavior trees provides a hierarchical, tree-structured framework for controlling the robot's actions and decision-making processes [25]. This framework consists of nodes with different functions, including those controlling the execution process and conditional judgments, as well as nodes that actually execute the robot's actions. Having previously defined the *behavior lib*, the LLM generates the behavior tree framework based on the behavioral skills and task instructions. This framework is stored in an XML file.

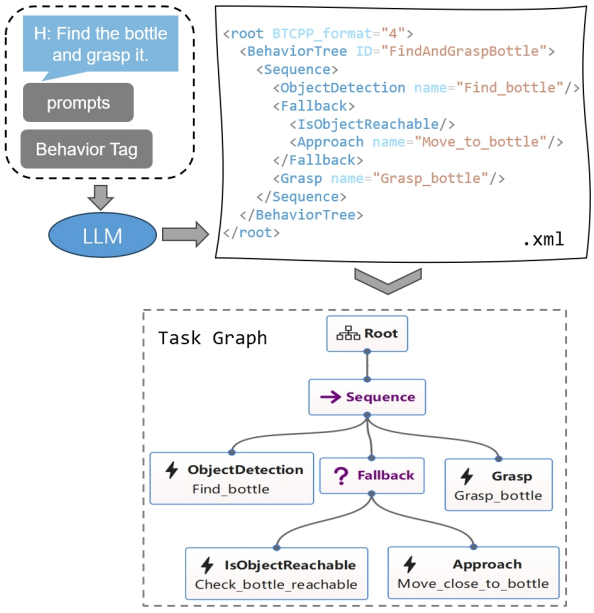

Fig. 3: Behavior Planner Grounding LLM

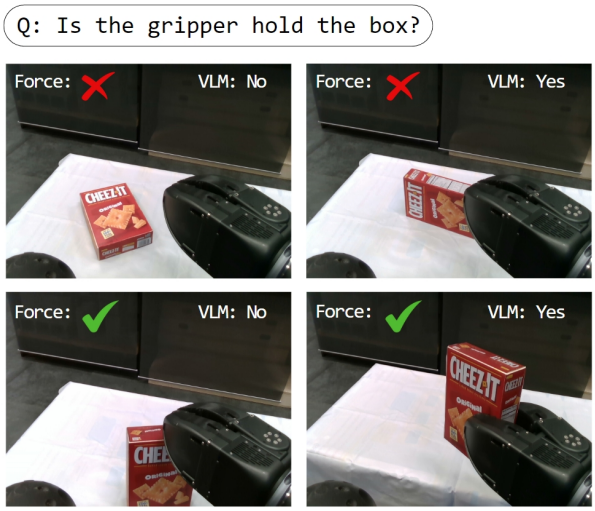

Fig. 4: Failure detection using a combination of perception behaviors. By asking VLM, the `visual Q&A` behavior can reason the state of the task, while using the torque sensor, the `Grip force` behavior will return the torque on the gripper.

The *task graph* is responsible for loading the behavior tree and invoking behaviors from the *behavior lib* according to the node guidance. This setup realizes decision reasoning through the LLM. The *task graph* is used for behavior planning, and the robot ultimately executes the tasks.

We access the `gpt-4` model as the LLM through the OpenAI API, which directly outputs the XML file used for generating the behavior tree, as shown in Figure 3.

3) Failure Detection and Recovery: In order to determine whether a task is successfully completed or deviates during execution, we try to incorporate a failure detection and recovery mechanism into the task graph. In our work, to take advantage of the visual language model's capability of understanding and reasoning about images, we utilize visual questions and answers (VQA) as perceptual behaviors to determine the current state of the robot performing the task, such as in the task of 'picking the box' by giving the robot's camera image and asking "Is the box being held?", the VLM will respond to the query by answering "`Yes` " or "`No`". Proprioceptive sensing like torque and distance has also been developed as behaviors to detect the potential failures in specific tasks, like during the tasks requiring grasping, detecting the torque on the gripper can be a reference of whether the object is being held. The perceptual behaviors we define in the behavior lib give multiple alternatives and combinations for the failure detection nodes, allowing the LLM to design the behavior tree based on the reasoning of different tasks. Some simple tasks such as "find and approach to object" only require the initiation of `Detection` behavior to determine if the object is available, while tasks that require multiple robotic actions often demand a combination of different perceptual behaviors for failure detection.

During the execution of the behavior tree, the node will return three signals: $success$, $failure$, and $running$, and the behavior tree will guide the execution of the behavior according to the returned signals. After the action node is executed, the condition node can be added to decide whether the current task is successful or returns the current state of the target object. For instance, in the process of grasping and lifting an object, after the completion of grasping, a condition node `IsObjectHeld` can be added to decide whether the object has been successfully grasped or not. In this scenario, the node will activate the `Grip force` and `Visual Q&A` perception behavior, which will obtain the torque of the gripper and ask the VLM "Is the (Target Object) held by the gripper". Only if there is a torque on the gripper and the VLM answers "Yes", then the node will return a success signal. The behavior tree will continue to execute the subsequent nodes. If it returns a failure, the recovery node is activated and the robot will try to grasp the object again.

IV. Experiment and Evaluation

We experimentally verify the capability of LLM as a behavior planner by implementing it and assessing its performance on CENTAURO robot executing long-horizon tasks under semantic commands. Few studies have employed LLM to plan the behavior of humanoid type of robots like CENTAURO and conducted real-world experiments. It is challenging to use different robots as a control group due to the variations in their functionalities and configurations. Therefore, we compare this method with our previous study in terms of functional aspects as shown in Table 2, and conduct preliminary experiments on applying LLM on the CENTAURO robot.

A. Experiment Setup

We conducted the experiment using objects from the YCB dataset [26] that are commonly found in an office kitchen.

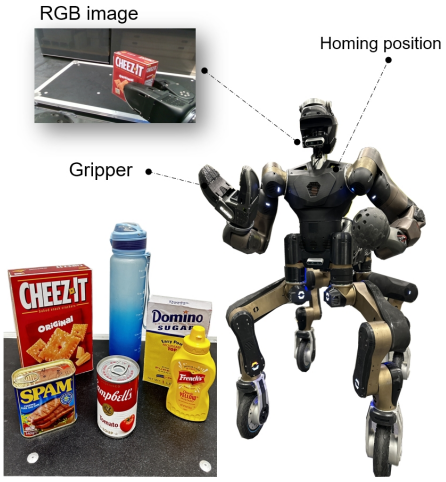

Fig. 5: Experiment setup

TABLE II: Comparison of different methods

Abilities	Method	
	Whole-body MPC [3]	LLM Behavior Planner (ours)
Autonomy	low	high
Whole-body motion	✔	✔
Long-horizon task	✗	✔
Failure detection	✗	✔
Real-time replanning	✔	✗

The test environment was an open area inside the lab, with objects randomly placed on a desk as shown in Figure 5. The CENTAURO robot, a hybrid wheels and legs quadruped robot with a humanoid upper body, features 37 degrees of freedom and a two-fingered claw gripper, enabling it to perform a wide range of loco-manipulation tasks. Equipped with an RGBD camera on its head and torque sensors in the joints throughout its body, the robot possesses extensive perceptual capabilities to measure joint efforts and interaction forces.

The action behaviors in the behavior lib were designed based on Cartesian I/O, requiring no extra training. The Xbot functions as middleware, providing real-time communication between the robot's various underlying actuators and the task commands through an API interface. For message transmission between the LLM, behavior lib, Behavior Tree, and the robot, we utilized the Robot Operating System (ROS) and conducted simulation experiments in the Gazebo simulator. Experiments for both simulation and the real world are demonstrated in the accompanying video.

B. Autonomous Humanoid Loco-manipulation Task

1) Behavior Planning with LLM: We first tested the LLM's behavior planning capabilities for robot tasks of varying complexity. The experiment was conducted for eight different tasks, including tasks with failure detection and recovery (FR), and the behaviors were planned using the method shown in Figure. 3. We provided standard instructions only for the given objects in Figure. 5, and the behaviors created in the behavior lib. These instructions are simple descriptions of the task content, e.g., "Find the soup can and pick it up." If failure detection and recovery are required during the task, this will need to be stated in the instructions such as "Pick the cracker, place it aside. Detect and recover the failure during the task.". We then used the Behavior Tree to load the XML files generated by the LLM and verified their feasibility. Finally, the appropriateness of the behavioral planning and the successful completion of the task were manually verified. Experiments that were executed and followed the requirements of the task instructions for planning were judged as successful. Each task was planned a total of 50 times, all using the same behavior lib and prompt. The time for each task graph generation was recorded, as well as the executable and success rate of the behavioral planning, as shown in Table 3.

TABLE III: Behavior planning results for different tasks including tasks with failure detection and recovery (FR).

Task	Executable	Success	Time(s)
Find object	100%	94%	14.93
Approach object	98%	90%	16.15
Grasp object	96%	92%	16.27
Pick object	96%	84%	17.11
Pick object (FR)	90%	82%	17.91
Pick and place object	92%	84%	18.23
Pick and place object (FR)	84%	80%	19.07
Find and pick object (FR)	86%	82%	17.86

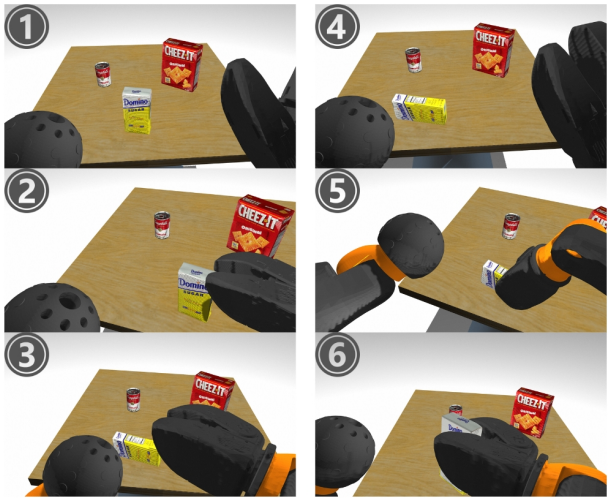

Fig. 6: Task execution with failure detection and recovery in simulation. Images 1, 2, and 3 show the robot's first attempt to pick up an object. After the perception behaviors detected that the gripper did not successfully grasp the object in image 3, then the robot tried again and successfully picked the object as shown in image 4, 5, 6.

2) Long-horizon Task Execution: After verifying that the behavioral plans generated by the LLM can be converted

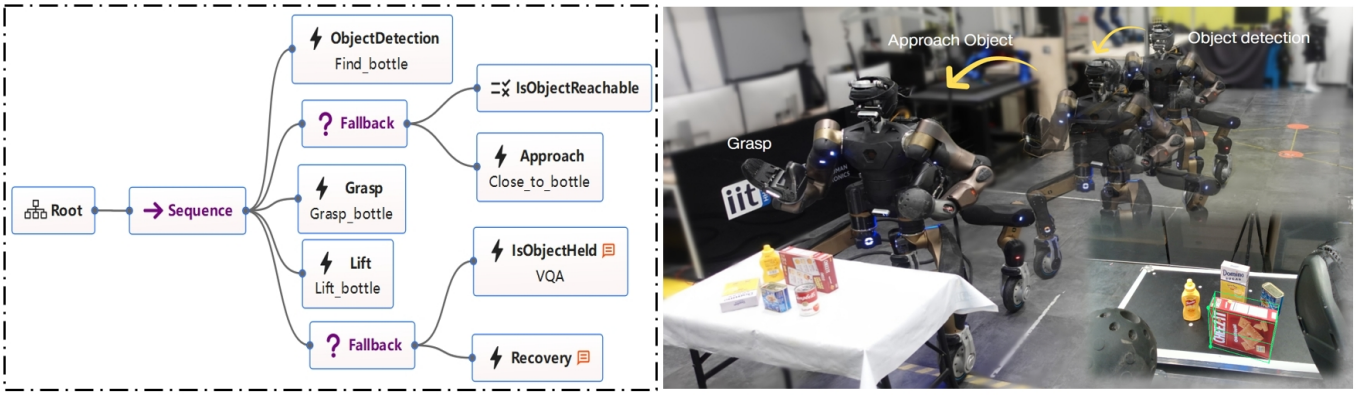

Fig. 7: Long-horizon task planning with CENTAURO robot. The left shows the task graph generated using LLM, the right shows the robot executing human instruction according to the behavioral plan.

into an executable Behavior Tree, we conducted experiments using the CENTAURO robot in both simulation and real-world environments. We selected the last six tasks from Table 3 to test the actual performance of the robot executing LLM-planned behaviors. Starting with object grasping and picking tasks, the initial positions of the robot were set at different distances from the target objects, which were randomly placed on the table. In the last task, the CENTAURO was placed relatively far to test the performance of the planner in tasks where "approach" behavior was necessary. For the same task, different descriptions of instruction and different target objects were used to verify the LLM's ability to reason about the simple task and plan the robot's behavior. For tasks that require in-process failure detection and recovery, the LLM incorporates perceptual behaviors in the behavioral planning phase and attempts to recover if it detects that the planned action fails to complete the task. This is shown in Figure. 6, where the recovery behaviors are selected based on the task requirements and the current state of the robot. Finally, the experiments focus on verifying the LLM's behavioral planning for long-horizon tasks and the robot's ability to perform autonomous loco-manipulation, with multiple perceptual and action behaviors being selected and combined to achieve the task goal, as shown in Figure. 7. We conducted 25 experiments for each task separately and recorded the success rate and execution time of the robot to complete the task in both simulation and the real environment, as shown in Table 4.

C. Results analysis

In the experiments, we evaluated the behavioral planning capabilities of the LLM for tasks with varying complexity levels, applying it to the CENTAURO robot. With a defined behavior lib and appropriate prompts, the LLM can generate corresponding behavior plans based on different task instructions, achieving a high planning success rate and task execution rate. These rates vary with the task's complexity and the number of behaviors needed to complete it. By comparing the original tasks with the tasks including FR, incorporating failure detection and recovery into the task process increases the difficulty of behavior planning, affecting the success rate of the generated task graphs. By adding FR, it increased the time for task execution with the robot as shown in Table 3, but not significantly increase the time for LLM planning. Finally, the behavioral planning time depends on the feedback speed of the language model used and the hardware device response time. The task complexity primarily affects the time taken to load the Behavior Tree, leading to minor differences in planning time across different tasks.

TABLE IV: Experiment results of simulation and real-world environment

Task	Simulation		Real World	
	Success	Time(s)	Success	Time(s)
Grasp object	92%	85.7	96%	98.4
Pick object	84%	104.9	80%	121.3
Pick object (FR)	88%	116.2	88%	167.1
Pick and place object	76%	132.7	72%	160.6
Pick and place object (FR)	84%	189.2	80%	203.2
Find and pick object (FR)	80%	174.5	76%	197.8

The results from robot task execution in both simulation and real environments demonstrate that LLM can effectively plan humanoid robot loco-manipulation tasks to a considerable degree. By integrating perception and action behavior in the behavior lib through LLM, the CENTAURO robot reaches a satisfactory level of success rate ($\geq$ 72%) in task execution. In long-horizon tasks, the incorporation of failure detection and recovery significantly boosts the robot's execution success rate in both simulation and real-world settings, and the success rate can be increased by up to 8% in specific tasks. Additionally, increasing task complexity and the addition of more robot behavioral nodes result in extended task implementation times as shown in Table 4.

V. Conclusion

In this work, we introduce an autonomous online behavioral planning framework utilizing a large language model

(LLM) for performing robot loco-manipulation tasks, requiring only human language instructions. Within this framework, we propose the concept of a behavior library and design action and perception behaviors, which are both interpretable and pragmatically efficient, with corresponding behavioral tags provided for semantic interpretations. The LLM organizes these behaviors into a task graph with a hierarchical structure, derived from the understanding of given instructions. The robot then follows the nodes in this task graph to sequentially complete the task. Additionally, it detects and attempts to correct possible failures by integrating the visual language model with intrinsic perceptions throughout the task process, thus successfully planning and executing long-horizon tasks. Experiments with the CENTAURO robot validate the achieved performance and practicality of this framework in robotic task planning.

Future work will focus on enriching the robot's behavior lib, as well as improving the prompts system, so that the LLM can better plan and optimize behavioral sequences automatically based on the robot's intrinsic mobility, manipulation, and perceptual strengths, thus enabling to perform more complex mobile manipulation tasks. Another direction is to improve the dynamic planning and multiconditional reasoning capability of the framework. This includes behavioral replanning in response to external perturbations or the introduction of artificial subtasks during a task.

REFERENCES

[1] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[2] N. Kashiri, L. Baccelliere, L. Muratore, A. Laurenzi, Z. Ren, E. M. Hoffman, M. Kamedula, G. F. Rigano, J. Malzahn, S. Cordasco, P. Guria, A. Margan, and N. G. Tsagarakis, "Centauro: A hybrid locomotion and high power resilient manipulation platform," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1595–1602, 2019.

[3] I. Dadiotis, A. Laurenzi, and N. Tsagarakis, "Whole-body mpc for highly redundant legged manipulators: Experimental evaluation with a 37 dof dual-arm quadruped," in *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, 2023, pp. 1–8.

[4] D. Kappler, P. Pastor, M. Kalakrishnan, M. Wüthrich, and S. Schaal, "Data-driven online decision making for autonomous manipulation." in *Robotics: science and systems*, vol. 11, 2015.

[5] Y. Ma, F. Farshidian, T. Miki, J. Lee, and M. Hutter, "Combining learning-based locomotion policy with model-based manipulation for legged mobile manipulators," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2377–2384, 2022.

[6] J.-P. Sleiman, F. Farshidian, and M. Hutter, "Versatile multicontact planning and control for legged loco-manipulation," *Science Robotics*, vol. 8, no. 81, p. eadg5014, 2023.

[7] A. De Luca, L. Muratore, and N. G. Tsagarakis, "Autonomous navigation with online replanning and recovery behaviors for wheeled-legged robots using behavior trees," *IEEE Robotics and Automation Letters*, vol. 8, no. 10, pp. 6803–6810, 2023.

[8] M. Murooka, I. Kumagai, M. Morisawa, F. Kanehiro, and A. Kheddar, "Humanoid loco-manipulation planning based on graph search and reachability maps," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1840–1847, 2021.

[9] BostonDynamics, "Inside the lab: Taking atlas from sim to scaffold."

[10] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[11] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[12] A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, A. Brohan *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models," *arXiv preprint arXiv:2310.08864*, 2023.

[13] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9493–9500.

[14] 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*. PMLR, 2022, pp. 9118–9147.

[15] Y. Ding, X. Zhang, C. Paxton, and S. Zhang, "Task and motion planning with large language models for object rearrangement," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 2086–2092.

[16] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "Voxposer: Composable 3d value maps for robotic manipulation with language models," *arXiv preprint arXiv:2307.05973*, 2023.

[17] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu *et al.*, "Palm-e: An embodied multimodal language model," *arXiv preprint arXiv:2303.03378*, 2023.

[18] A. Yang, A. Nagrani, P. H. Seo, A. Miech, J. Pont-Tuset, I. Laptev, J. Sivic, and C. Schmid, "Vid2seq: Large-scale pretraining of a visual language model for dense video captioning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 10 714–10 726.

[19] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "Vqa: Visual question answering," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2425–2433.

[20] M. Colledanchise and P. Ögren, *Behavior trees in robotics and AI: An introduction*. CRC Press, 2018.

[21] L. Muratore, A. Laurenzi, E. M. Hoffman, and N. G. Tsagarakis, "The xbot real-time software framework for robotics: From the developer to the user perspective," *IEEE Robotics & Automation Magazine*, vol. 27, no. 3, pp. 133–143, 2020.

[22] A. Laurenzi, E. M. Hoffman, L. Muratore, and N. G. Tsagarakis, "Cartesi/o: A ros based real-time capable cartesian control framework," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 591–596.

[23] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," *arXiv preprint arXiv:1809.10790*, 2018.

[24] R. OpenAI, "Gpt-4v (ision) system card," *Citekey: gptvision*, 2023.

[25] M. Iovino, E. Scukins, J. Styrud, P. Ögren, and C. Smith, "A survey of behavior trees in robotics and ai," *Robotics and Autonomous Systems*, vol. 154, p. 104096, 2022.

[26] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols," *arXiv preprint arXiv:1502.03143*, 2015.