Dynamic Penetrative Trajectory Adaptation ⋆

Zhifeng Han. Author * **Clarire Walton. Author, Jr.** **

* *University of Texas at San Antonio, San Antonio, TX 78249 USA (e-mail: zhifeng.han@utsa.edu).*
** *University of Texas at San Antonio, San Antonio, TX 78249 USA (e-mail: claire.walton@utsa.edu).*

Abstract: DPTA is a conceptual framework where the LLM's world knowledge is leveraged to analyze low-level sensor feedback, enabling the dynamic selection and refinement of complex, pre-learned robot trajectories to achieve high dexterity and resilience in autonomous manipulation.

Keywords: LLM, robotic, Reenforcement Learning.

1. INTRODUCTION

interactive robots mission is a complex mission it need to nagvigate in complex and dynamic environment also need to fully flexable to planing the sequentially sub tasks. In order to achieve the goal we demo a method by utlize LLMs(large language model) and Reenforcement Learning. Our idea focuses on using an LLM's capacity for precise, short-horizon physical reasoning to dynamically select or modify the parameters of complex, learned robotic trajectories, effectively allowing the robot to autonomously execute tasks that previously required initial human intervention. The conceptual framework of Dynamic Penetrative Trajectory Adaptation (DPTA) represents a synthesis of three distinct advanced AI paradigms found in the sources: Penetrative AI (LLM sensor comprehension), Dynamic Movement Primitives (DMP) and Human-Robot Collaboration (HRC) for trajectory learning, and Robust Reinforcement Learning (RL) for policy optimization and action excution.

2. BACKGROUND

The complex nature of interactive robot missions, particularly those involving high-level language models (LLMs) or sophisticated reinforcement learning (RL), presents numerous challenges across planning, learning, perception, and execution.

2.1 Main challenges

Limitations in High-Level Planning and Trajectory Execution (LLMs): When LLMs are used for autonomous manipulation, they encounter issues related to generating and executing physical motions: Inability to Handle Complex Trajectories(Liu et al. (2024)) (Feasibility Issues): The conventional approach of using LLMs to generate code for robot motion falters when dealing with complex trajectories. Tasks that require intricate trajectory planning and reasoning over environments, such as opening an oven door featuring a horizontal axis design or opening a cabinet with a press-pull structure, may be deemed infeasible when relying only on the basic motion library generated by the LLM(Liu et al. (2024)). Fragility of Prompt Design: The current design paradigm for using LLMs as controllers is fragile, meaning even minor alterations in the prompt can dramatically affect the performance.Wang et al. (2024) Designing a reliable prompt for robotic tasks is not yet well understood.

Executability Anomalies: Although generally high, the code generated by the LLM can occasionally generate sub-tasks without assigning corresponding motion functions, resulting in non-encodable and non-executable responses(Liu et al. (2024)).

Issues Related to Perception and Grounding:

Successfully linking high-level instructions to the physical world introduces multiple errors: Environmental Perception Errors and Error Accumulation: Real-world task success rates decrease due to error accumulation across sequential sub-tasks(Liu et al. (2024)). Errors in environmental perception stem from inaccuracies in object detection models (like YOLOv5), such as bounding box inaccuracies, leading to slightly variable coordinates for target objects.Liu et al. (2024) These discrepancies can cause the errors to exceed the necessary margins for precise manipulation (e.g., placing an apple into an oven with minimal clearance)(Liu et al. (2024)) Sensor Data Processing Limitations: LLMs, when used in a "penetrative" way to analyze digitized sensor signals (like sequences of ECG digits), exhibit lower efficiency in processing extensive sequences of digital data compared to traditional methods(Xu et al. (2024)). The hallucination rates and Mean Absolute Errors (MAEs) for some LLMs escalate with the increase in window size of the input data, suggesting an inherent limitation in processing extensive lengths of digitized sequences(Xu et al. (2024)). Susceptibility to Deployment Noise: Policies trained in simulation, even

⋆ Sponsor and financial support acknowledgment goes here. Paper titles should be written in uppercase and lowercase letters, not all uppercase.

when using modern techniques, may not be robust to real-world noise. For instance, a policy trained in simulation for pick-and-place was not robust to small errors in box position estimation (e.g., errors with a standard deviation of 1cm) when deployed on a physical robot(Andrychowicz et al. (2017)).

Challenges in Low-Level Control:

Controllability and Security Risks: Since LLM responses are probabilistic, there is no guarantee that the swarm will behave as intended. This also introduces new security vulnerabilities, as it needs to be studied if users or even other robots can reprogram robots through prompt injection attacks or if a malicious agent could send misleading information (Byzantine robot detection)(Strobel et al.).

2.2 Overview: Dynamic Penetrative Trajectory Adaptation (DPTA)

In current LLM-based manipulation, environmental information is primarily derived visually (e.g., YOLOv5 for object position)(Liu et al. (2024)). However, fine-grained tasks often depend on non-visual physical feedback (e.g., force or torque required to open a tight hinge). In DPTA, the Penetrative AI(Xu et al. (2024)) paradigm is employed to process digitized sensor signals** from the robot's end effector (e.g., force/torque sensors, joint current feedback)

- LLM's task is not to determine a broad state (like "indoors/outdoors"), but to execute a real-time, micro-level physical classification of the object/environment state during the initial phase of interaction (e.g., the first 100 milliseconds of grasping a handle or pushing a button)Bhat et al. (2024).
- Prompt Design: Our prompt utilizes the procedural guidance and fuzzy logic methods demonstrated in the heart rate detection task.(Xu et al. (2024)) It contains a short sequence of raw numerical feedback (avoiding the token limit constraint associated with long digitized sequences) and instructs the LLM to classify the physical anomaly based on relative changes in the sequence.
- Example Output: Based on the input torque sequence, the LLM reasoning(Liu et al. (2024)) determines the precise physical condition, such as "Horizontal-Axis Oven Door, stiff hinge" or "Press-Pull Cabinet, high friction."
- Dynamic Controller Role: The LLM functions as a **dynamic feedback controller**, selecting between:

The proposed Dynamic Penetrative Trajectory Adaptation (DPTA) framework differentiates itself from existing research in LLM-controlled robotics by addressing limitations in physical grounding, low-level trajectory execution, and robust policy learning.

(1) **Existing Work: High-Level Planning with Imperfect Grounding** Existing works leverage large language models (LLMs) as **zero-shot planners** to decompose high-level instructions (e.g., "make breakfast") into a sequence of actionable steps [1-4]. Methods like those involving **Semantic Translation** improve executability by mapping LLM-generated phrases to the most semantically similar admissible action in a predefined set [5-7]. **SayCan** grounds LLM output by weighting actions based on learned **skill affordances** (value functions), ensuring the proposed step is feasible in the current state [1, 2, 8, 9].

→ **Limitation: Reliance on Narrow APIs and Lack of Deep Physical Context**

Despite these efforts, LLM-generated plans are still **frequently not executable** in interactive environments [10-12] and struggle with mid-level grounding, often missing necessary common-sense actions [3]. Critically, most systems rely on predetermined **vision APIs** (like object detectors) to describe the scene [13]. This focus on visual data ignores crucial non-visual **physical feedback** (e.g., force or torque) needed for fine-grained tasks [13, 14], leaving the LLM policies restricted by **what the perception APIs can describe** [15].

→ **Your Approach: Penetrative Awareness for Micro-Level Physical Grounding**

DPTA implements **Penetrative AI**, enabling the LLM to directly process **raw or digitized sensor signals** (numerical sequences) from the end effector [14, 16, 17]. This allows the system to achieve **micro-level physical classification** (e.g., determining hinge stiffness or surface friction) during the initial phase of interaction, providing crucial contextual grounding that traditional visual-language models (VLMs) lack [14].

(2) **Existing Work: Code Generation and Trajectory Management**

Recent research utilizes LLMs to directly generate **policy code (Code as Policies)**, allowing the model to compose perception-to-control logic and reference external libraries (like NumPy) for complex behaviors such as spatial-geometric reasoning [18, 19]. Furthermore, LLMs have been explored as **low-level feedback policies** for dynamic systems like robot walking by outputting target joint positions directly from historical input-output sequences [20, 21].

→ **Limitation: Failure on Complex Trajectories and Prompt Fragility**

LLM-generated code often **falters when dealing with complex trajectories** [13]. While the code approach allows the robot to execute sequences of predefined motion primitives (e.g., pick, move, place), it assumes the existence of complex skills (e.g., 'robot.place_a_bit_right()') [22]. Furthermore, using LLMs for low-level control is highly susceptible to **prompt fragility**, where minor alterations can dramatically affect performance [23]. Generating code for intricate or long, specialized trajectories remains impractical without extensive domain knowledge injection.

→ **Your Approach: Dynamic Movement Primitives (DMP) via HRC**

DPTA mitigates the code generation failure for complex motions by incorporating **Dynamic Movement Primitives (DMP)** [24]. Complex, specialized trajectories are acquired through **manual teleoperation** via **Human-Robot Collaboration (HRC)** and stored in a specialized **DMP library** [24, 25]. The LLM's task is then reduced to selecting the correct, highly specialized, and robust DMP from the library to execute, promoting **task-specific autonomy** without forcing the LLM to synthesize brittle code for movement kinematics [24].

(3) **Existing Work: Closed-Loop Adaptation and Policy Learning**

Systems such as **Inner Monologue** and **Socratic Models** integrate closed-loop feedback by feeding natural language observations (e.g., success detection or scene description updates) back into the LLM prompt, enabling the LLM planner to reason over outcomes and dynamically re-plan actions [2, 26, 27]. In the domain of reinforcement learning (RL), solving robotic tasks often faces the critical challenge of **sparse and binary rewards** [28].

→ **Limitation: Feedback Limitations and Learning Inefficiency**

The effectiveness of closed-loop reasoning is **bottlenecked** by the capabilities of the low-level control policies and the fidelity of the language description provided by the perception system [15, 29]. Moreover, traditional RL systems often fail in large state spaces when faced with sparse rewards, often necessitating tedious and domain-specific **reward function engineering** [28].

→ **Your Approach: Robust Reinforcement Learning Integration**

DPTA incorporates **Hindsight Experience Replay (HER)**, an advanced RL principle, to efficiently train universal policies that overcome the challenge of sparse rewards [28]. HER allows the system to **re-examine failed trajectories** with the retrospectively achieved state treated as a successful goal, thus harvesting information even from episodes that originally received a reward of -1 [28]. This enhances the policy learning component, ensuring the executed policies remain resilient and adaptable over time [28].

REFERENCES

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., and Zaremba, W. (2017). Hindsight experience replay. *Advances in neural information processing systems*, 30.

Bhat, V., Kaypak, A.U., Krishnamurthy, P., Karri, R., and Khorrami, F. (2024). Grounding llms for robot task planning using closed-loop state feedback. *CoRR*.

Liu, H., Zhu, Y., Kato, K., Tsukahara, A., Kondo, I., Aoyama, T., and Hasegawa, Y. (2024). Enhancing the llm-based robot manipulation through human-robot collaboration. *IEEE Robotics and Automation Letters*, 9(8), 6904–6911.

Strobel, V., Dorigo, M., and Fritz, M. (????). Llm2swarm: Robot swarms that responsively reason, plan, and collaborate through llms. In *NeurIPS 2024 Workshop on Open-World Agents*.

Wang, Y.J., Zhang, B., Chen, J., and Sreenath, K. (2024). Prompt a robot to walk with large language models. In *2024 IEEE 63rd Conference on Decision and Control (CDC)*, 1531–1538. IEEE.

Xu, H., Han, L., Yang, Q., Li, M., and Srivastava, M. (2024). Penetrative ai: Making llms comprehend the physical world. In *Proceedings of the 25th International Workshop on Mobile Computing Systems and Applications*, 1–7.

Appendix A. A SUMMARY OF LATIN GRAMMAR

Appendix B. SOME LATIN VOCABULARY