Dynamic Penetrative Trajectory Adaptation [⋆]

Zhifeng Han [*] **Claire Walton** [**]

[*] *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, Reinforcement 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 complementary AI paradigms. First, Penetrative AI enables contextual and semantic understanding by leveraging large language model-based sensor comprehension for high-level intent extraction and environmental interpretation. Second, the combination of Dynamic Movement Primitives (DMP) and Human–Robot Collaboration (HRC) contributes to adaptable trajectory generation through shared autonomy, interactive learning, and human demonstration. Third, Robust Reinforcement Learning (RL) performs policy optimization and closed-loop execution under uncertainty, ensuring real-time correction and robust action adaptation. Together, these three paradigms form the foundation of the DPTA framework, enabling semantic reasoning, dynamic motion modification, and reliable trajectory shaping in complex robotic environments.

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, see 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 LLMs.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, see 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, see 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. 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. 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), see 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

[⋆] This work was supported in part by the U.S. National Science Foundation (NSF) under Award No. 1826086.

to traditional methods. 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, see Xu et al. (2024). Susceptibility to Deployment Noise: Policies trained in simulation, even 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, see 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), see 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), see 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, see 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), see Bhat et al. (2024).

Prompt Design: Our prompt utilizes the procedural guidance and fuzzy logic methods demonstrated in the heart rate detection task, see 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, see 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:

2.3 Literature Review

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.

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, see Huang et al. (2022). Methods like those involving Semantic Translation improve executability by mapping LLM-generated phrases to the most semantically similar admissible action in a predefined set. SayCan grounds LLM output by weighting actions based on learned skill affordances (value functions), ensuring the proposed step is feasible in the current state, see Brohan et al. (2023).

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 and struggle with mid-level grounding, often missing necessary common-sense actions, see Huang et al. (2022). Critically, most systems rely on predetermined vision APIs (like object detectors) to describe the scene, see Vemprala et al. (2024). This focus on visual data ignores crucial non-visual physical feedback (e.g., force or torque) needed for fine-grained tasks, see Huang et al. (2023), leaving the LLM policies restricted by what the perception APIs can describe, see Brown et al. (2020).

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 [Liang et al. (2023)]. 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, see Liang et al. (2023).

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 [huang2023inner]. In the domain of reinforcement learning (RL), solving robotic tasks often faces the critical challenge of sparse and binary rewards, see Andrychowicz et al. (2017).

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, see Gal and Ghahramani (2016), Ebert et al. (2018). Moreover, traditional RL systems often fail in large state spaces when faced with sparse rewards, often necessitating tedious and domain-specific reward function engineering, see Andrychowicz et al. (2017).

3. ARCHITECTURE

The **Dynamic Penetrative Trajectory Adaptation (DPTA)** framework is a hierarchical robotic architecture that integrates large-scale language-based reasoning, phys-

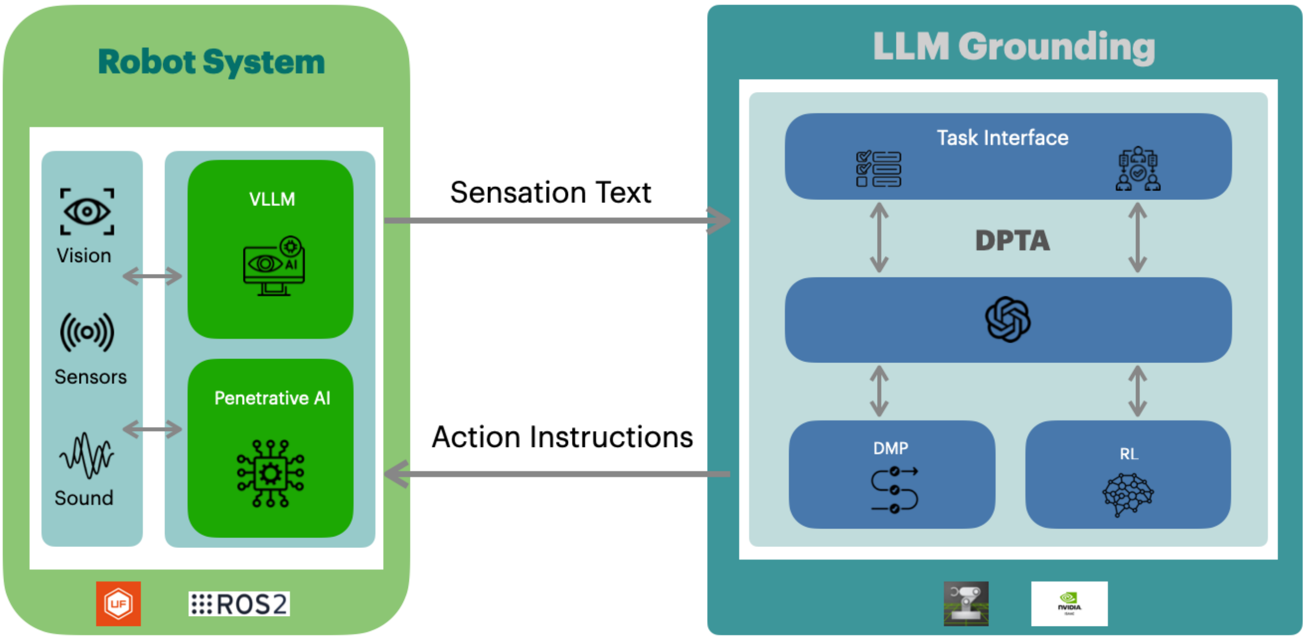

Fig. 1. System demonstration

ically grounded sensory interpretation, adaptive trajectory learning, and reinforcement learning. The system enables a robot to interpret natural language instructions, perceive through multimodal signals, and adapt its motion in real-world conditions (see Figure 1).

$$\mathcal{A}_{\text{DPTA}} = \{\mathcal{L}_{\text{plan}}, \mathcal{L}_{\text{perc}}, \mathcal{L}_{\text{exec}}, \mathcal{L}_{\text{learn}}\}.$$

3.1 High-Level Reasoning and Planning Layer

At the highest level, a Large Language Model (LLM) acts as the cognitive planner. Given a natural-language instruction (e.g., “open the cabinet”), the LLM performs long-horizon task decomposition and generates a structured sequence of sub-tasks. Each sub-task is mapped to motion primitives from a predefined *Basic Motion Library*, enabling generalized planning without task-specific programming.

3.2 Penetrative Perception Layer

Unlike vision-centric pipelines, DPTA leverages *Penetrative AI*, see Xu et al. (2024) to process raw sensor signals (e.g., joint torques, force/torque sensors). Using embedded knowledge and numerical reasoning, the LLM infers latent physical properties such as material stiffness, friction, compliance, or slippage. This enables detection of micro-level physical states that are difficult to capture visually.

3.3 Dynamic Execution Layer

When a task exceeds the capabilities of predefined motion primitives, DPTA employs *Dynamic Movement Primitives (DMP)*, see Wu et al. (2023) combined with *Human-Robot Collaboration (HRC)*, see Andrychowicz et al. (2017). Demonstrated trajectories are encoded as DMPs and stored in a library for future reuse, allowing continual expansion of manipulation skills.

3.4 Dynamic Execution Layer (Trajectory Adaptation via DMP)

This layer utilizes **Dynamic Movement Primitives (DMP)** to encode and reproduce complex behaviors learned from human demonstration. DMPs allow the system to generalize movements to new spatial targets while preserving the original trajectory shape.

The transformation system in DMPs can be expressed as a spring–damper system driven by a nonlinear forcing function:

$$\ddot{y} = \alpha\left(\beta(g-y) - \dot{y}\right) + f(x)(g-y) \tag{1}$$

where:

- y is the system state (e.g., end-effector coordinates or joint angle).
- g is the goal state.
- α, β are positive gain constants.
- $f(x)$ is the nonlinear forcing term that shapes the trajectory.

The forcing term is defined as a weighted sum of Gaussian basis functions:

$$f(x) = \frac{\sum_{i=1}^{N} \psi_i w_i x}{\sum_{i=1}^{N} \psi_i} \tag{2}$$

where:

- ψ_i are Gaussian basis functions,
- w_i are learnable weights obtained from human demonstration,
- x is the canonical phase variable, decaying monotonically over time,
- N is the number of basis functions (set to $N = 15$ in our implementation).

By modifying only the goal state g or start state y_0, the DPTA framework can generate new adaptive trajectories while maintaining the learned motion structure. This enables smooth re-targeting and robust trajectory reshaping without retraining.

3.5 Robust Learning Layer

To ensure continuous improvement, DPTA incorporates *Hindsight Experience Replay (HER)*, see Andrychowicz et al. (2017). Sparse rewards in manipulation tasks are relabeled, allowing failed trajectories to contribute to learning alternative goals. This approach improves sample efficiency and policy robustness, enabling the system to autonomously adapt over time.

4. PROGRAM

Table 1. System Components and Roles in the UFactory-Based DPTA Experiment

Component	**Role**
UFactory Lite 6	Physical execution & HRC skill acquisition
UF Studio	Manual teaching with built-in recorder (joint position, velocity, gripper force)
UFactory Python SDK (Modbus/xArm API)	Low-level execution, feedback readout, torque and velocity streaming
IsaacSim + IsaacGym	Physics variability and large-scale parallel reinforcement learning
LLM Planner (GPT-4/5)	High-level command interpretation and motion adaptation decisions
DMP Encoding Module	Gaussian basis encoding of human-demonstrated motion primitives

5. CONCLUSION

This work introduces the **Dynamic Penetrative Trajectory Adaptation (DPTA)** framework, a novel architecture bridging high-level semantic planning and low-level dynamic control in embodied agents. While LLMs provide broad world knowledge capable of decomposing abstract tasks, they lack real-world experience and physical grounding, often generating plans that are plausible but not executable. DPTA addresses these limitations by synthesizing *Penetrative AI* Xu et al. (2024), *Dynamic Movement Primitives (DMP)* Huang et al. (2023), nd *Hindsight Experience Replay (HER)*, seeAndrychowicz et al. (2017) into a cohesive system.

First, DPTA enhances perception by leveraging Penetrative AI to interpret digitized sensor signals such as force and torque, enabling inference of micro-level physical states (e.g., hinge stiffness, surface friction). Second, it addresses the limitations of rigid code generation by parameterizing DMPs through Human-Robot Collaboration (HRC), ensuring kinematically sound and human-like motions. Finally, robust learning through HER allows the system to learn from failure, improving policy performance in high-dimensional state spaces.

In summary, DPTA moves beyond the “Say” and “Can” dichotomy by introducing “Feel” and “Adapt” capabilities. By grounding semantic intent in penetrative sensory data and leveraging both stored motion primitives and learned behaviors, DPTA enables embodied agents to plan logically and interact physically with nuance and adaptability.

DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS

During the preparation of this work the author(s) used NotbookLM and ChatGPT in order to help English as secondary language speaker. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

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.*

Brohan, A., Chebotar, Y., Finn, C., Hausman, K., Herzog, A., Ho, D., Ibarz, J., Irpan, A., Jang, E., Julian, R., et al. (2023). Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, 287–318. PMLR.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33, 1877–1901.

Ebert, D.D., Van Daele, T., Nordgreen, T., Karekla, M., Compare, A., Zarbo, C., Brugnera, A., Øverland, S., Trebbi, G., Jensen, K.L., et al. (2018). Internet-and mobile-based psychological interventions: applications, efficacy, and potential for improving mental health. *European Psychologist.*

Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, 1050–1059. PMLR.

Huang, W., Abbeel, P., Pathak, D., and Mordatch, I. (2022). Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning.*

Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., et al. (2023). Inner monologue: Embodied reasoning through planning with language models. In *Conference on Robot Learning*, 1769–1782. PMLR.

Liang, J., Huang, W., Xia, F., Xu, P., Hausman, K., Ichter, B., Florence, P., and Zeng, A. (2023). Code as policies:

Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

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*.

Vemprala, S.H., Bonatti, R., Bucker, A., and Kapoor, A. (2024). Chatgpt for robotics: Design principles and model abilities. *Ieee Access*, 12, 55682–55696.

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.

Wu, P., Escontrela, A., Hafner, D., Abbeel, P., and Goldberg, K. (2023). Daydreamer: World models for physical robot learning. In *Conference on robot learning*, 2226–2240. PMLR.

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.