Science, Engineering and Technology Vol. 6, No. 1, pp. 52–72
www.setjournal.com https://doi.org/10.54327/set2026/v6.i1.325

Neuro-Symbolic Control with Large Language Models for Language-Guided Spatial Tasks

Momina Liaqat Ali[1], Muhammad Abid[2], Muhammad Saqlain[3], and Jose M. Merigo[3]
[1]*Department of Computer Science, Middle Tennessee State University, Murfreesboro, TN, 37130, USA.*
[2]*Department of Mechanical and Aerospace Engineering, University of Tennessee, Knoxville, TN, 37996, USA.*
[3]*School of Computer Science, Faculty of Engineering and Information Technology, University of Technology Sydney, 81 Broadway, Ultimo, NSW, 2007, Australia.*

Abstract

Although large language models (LLMs) have recently become effective tools for language-conditioned control in embodied systems, instability, slow convergence, and hallucinated actions continue to limit their direct application to continuous control. A modular neuro-symbolic control framework that distinguishes between low-level motion execution and high-level semantic reasoning is proposed in this work. While a lightweight neural delta controller performs bounded, incremental actions in continuous space, a locally deployed LLM interprets symbolic tasks. We assess the suggested method in a planar manipulation setting with spatial relations between objects specified by language. Numerous tasks and local language models, such as Mistral, Phi, and LLaMA-3.2, are used in extensive experiments to compare LLM-only control, neural-only control, and the suggested LLM+Deep Learning (LLM+DL) framework. In comparison to LLM-only baselines, the results show that the neuro-symbolic integration consistently increases both success rate and efficiency, achieving average step reductions exceeding 70% and speedups of up to $8.83\times$ while remaining robust to language model quality. The suggested framework enhances interpretability, stability, and generalization without any need of reinforcement learning or costly rollouts by controlling the LLM to symbolic outputs and allocating uninterpreted execution to a neural controller trained on artificial geometric data. These outputs show empirically that neuro-symbolic decomposition offers a scalable and principled way to integrate language understanding with ongoing control, this approach promotes the creation of dependable and effective language-guided embodied systems.

Keywords: Neuro-Symbolic Control, LLMs, Language-Guided Robotics, Closed-Loop Control, Robotics, Deep Learning, Autonomous Systems

1. Introduction

Recent developments in large language models (LLMs) have shown their exceptional capacity for high-level decision-making, reasoning, and instruction following [1], [2]. These kinds of models are being studied extensively as cognitive engines for robotics and control systems, where they are employed to create the different tasks and plans, choose actions in complex environments, and interpret natural language instructions. However, when used in closed-loop control settings, LLMs show basic limitations despite their expressive power [3]. Particularly, LLM-only control policies typically suffer from hallucinated actions, poor sample efficiency, lack of convergence guarantees, and brittle behavior under feedback, which makes them unreliable for precise spatial manipulation and continuous control tasks [4], [5]. As illustrated in Figure 1, directly using large language models for continuous control leads

Corresponding author: Jose M. Merigo (Jose.Merigo@uts.edu.au)
Received: 29 December 2025; Revised: 12 February 2026; Accepted: 17 February 2026; Published: 4 April 2026

to unstable behavior, motivating the proposed neuro-symbolic decomposition.

In parallel, learning-based controllers such as deep neural networks have achieved strong performance in low-level control and perception tasks, benefiting from fast inference, smooth action outputs, and stability under feedback. However, these models typically lack symbolic reasoning and struggle to generalize across tasks without retraining [6]. Purely neural controllers are therefore limited in their ability to handle task-level abstraction, compositional instructions, or symbolic relational reasoning such as spatial constraints (e.g., *left of*, *right of*, *above*, *below*).

This dichotomy has motivated growing interest in neuro-symbolic control, which seeks to combine the complementary strengths of symbolic reasoning systems and neural function approximators. In this paradigm, symbolic components provide task-level reasoning and interpretability, while neural modules handle continuous control and execution. While prior work has explored hybrid approaches using classical symbolic planners or handcrafted logic modules [7], the emergence of LLMs offers a new opportunity: using language models as flexible, general-purpose symbolic planners that can reason over tasks, goals, and constraints expressed in natural language.

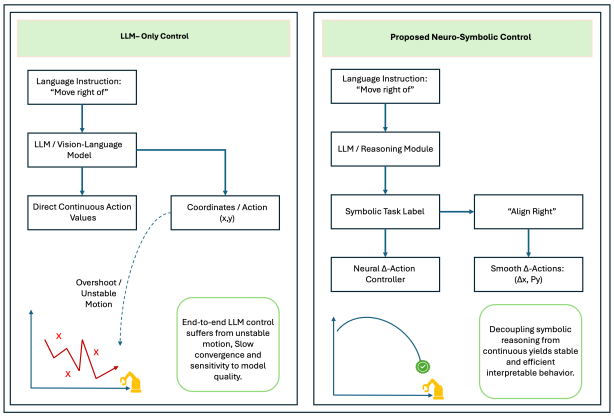

Figure 1. Motivation for neuro-symbolic control. End-to-end LLM-based control directly predicts continuous actions, leading to unstable motion and slow convergence. Our approach decouples symbolic reasoning from continuous execution, combining LLM-based semantic understanding with a neural delta controller for stable and efficient closed-loop control.

Nevertheless, directly deploying LLMs as action generators in control loops still remains problematic [8]. LLM outputs are fundamentally stochastic, sensitive to prompt phrasing, and also not optimized for numerical precision [9]. These type issues become particularly pronounced in spatial control tasks, where the small positional errors can lead to task failure or also towards slow convergence. In term of the result in this scenario, recent studies have highlighted the need for structured interfaces and also the auxiliary control mechanisms to ground LLM reasoning in stable, low-level execution [10], [11].

In the current research work, we propose a neuro-symbolic closed-loop control framework that precisely integrates local large language models with a lightweight type of neural delta controller. So, the LLM is responsible for high-level symbolic reasoning, task interpretation, and also for the goal selection, while the neural controller executes fine-grained corrective actions that ensure fast and stable convergence. By empowering the numerical control to the neural module and restricting the LLM to structured symbolic outputs, the proposed framework mitigates hallucinations, reduces action variance, and significantly improves the sample efficiency.

A fundamental aspect of our proposed approach is the use of locally hosted LLMs, including Mistral, Phi, and LLaMA-3.2. Unlike cloud-based proprietary models, local LLMs facilitate reproducibility, low-latency inference, privacy preservation, and deployment on edge or also resource-constrained systems. Local models, although they are often smaller and less robust than large proprietary counterparts, make them one of the best testbeds for evaluating whether neuro-symbolic integration can compensate for limited model capacity. Our results demonstrate that the proposed hybrid framework consistently improves performance across all evaluated local LLMs, also indicating that the improvements arise from architectural synergy rather than the model scale alone.

We evaluate our method on a suite of spatial reasoning and control tasks in a simulated 2D environment, including

relational goals such as *right of*, *left of*, *above*, and *below*. We compare three settings: LLM-only control, deep learning (DL)-only control, and the proposed LLM+DL hybrid. Extensive experiments show that the hybrid approach achieves substantially higher success rates, reduces the number of control steps by up to an order of magnitude, and yields significant speedups over LLM-only baselines. These improvements are consistent across many different types of LLM architectures and tasks, highlighting the robustness and generality of the proposed framework.

Here are the main and clear contributions of this paper as follows:

- We introduce a neuro-symbolic closed-loop control architecture that combines LLM-based symbolic reasoning along neural delta controller for stable execution.
- We provide a rigorous evaluation of the local LLMs (including Mistral, Phi, and LLaMA-3.2) in spatial control tasks, those highlighting their limitations in isolation and their strengths when integrated with neural control.
- We show the major improvements in success rate, convergence speed, and also efficiency using the proposed hybrid framework, supported by a comprehensive quantitative analysis and ablation studies.
- We release a reproducible experimental pipeline and evaluation framework that facilitates further research on neuro-symbolic control with local LLMs.

2. Related Work

This work intersects several active research areas, including large language models for control, neuro-symbolic reasoning, hybrid planning and control architectures, and learning-based spatial reasoning. We review the most relevant literature and highlight how our approach differs from prior work.

2.1. Large Language Models for Planning and Control

Large language models have recently been explored as high-level planners and controllers for robotic systems due to their strong reasoning and instruction-following capabilities [12], [13]. Early work demonstrated that LLMs can generate action sequences, code, or symbolic plans from natural-language task descriptions, enabling flexible task specification and zero-shot generalization [14]. Examples include language-driven planning frameworks that translate instructions into executable programs or symbolic action graphs.

Several landmark studies have applied LLMs directly to robotic control loops. SayCan [15] demonstrated that combining LLM-based task planning with learned affordance functions enables robots to execute complex, multi-step instructions in real kitchen environments. PaLM-E [16] extended this paradigm by incorporating multimodal sensor inputs directly into the language model, enabling end-to-end embodied reasoning across manipulation and navigation tasks. Code-as-Policies [17] showed that code-writing LLMs can generate executable robot policies that exhibit spatial-geometric reasoning and generalize to new instructions without additional training.

While promising, LLM-only approaches often suffer from instability, hallucinated actions, and slow convergence due to the lack of grounding in physical dynamics [18]. According to empirical investigations, LLM-only controllers perform poorly on tasks those requiring exact numerical reasoning or spatial accuracy [19], [20], show considerable variance between trials, and are sensitive to prompt phrasing. In most recent work, that has attempted to mitigate these issues through prompt engineering, structured outputs (e.g., JSON schemas), or iterative replanning [21]. However, such methodologies do not essentially address the mismatch among the discrete language reasoning and continuous control execution. But our work explicitly separates symbolic reasoning and numerical control, using the LLM only for high-level task with interpretation while delegating execution to a neural

controller trained for stability and precision.

2.2. Neuro-Symbolic and Hybrid Reasoning Systems

Neuro-symbolic AI aims to combine the comprehensibility and also the compositionality of symbolic reasoning with the scalability and robustness of neural networks [22], [23]. Classical methods especially integrate logic-based systems, rule engines, or symbolic planners with neural perception or control modules [24]. These type of methods have been applied in domains such as visual reasoning, program induction, and task planning [25], [26].

In the robotics field, hybrid symbolic-neural architectures often employ symbolic task planners (e.g., STRIPS, PDDL-based planners) coupled with some low-level controllers learned via reinforcement learning or also supervised learning [27]. While effective, these kinds of approaches typically depend on the handcrafted symbolic representations and domain-specific planners, limiting flexibility and scalability.

In the recent advances, they propose replacing classical symbolic planners with LLMs, leveraging their ability to reason over diverse tasks without explicit domain modeling [28]. However, many existing LLM-based neuro-symbolic systems specially focus on open-loop planning or single-shot decision-making rather than closed-loop control [29]. But our work extends this paradigm by embedding the LLM reasoning within a closed-loop control framework, where symbolic decisions are continuously corrected by a learned neural controller, which is enabling fast convergence and robustness to noise.

2.3. Learning-Based Controllers and Delta Control

Learning-based controllers, especially deep neural networks, have demonstrated a strong performance in continuous control tasks due to their ability to approximate the nonlinear dynamics and also to generate smooth action trajectories [30]. For stabilization and fine-grained adjustments, delta controllers, which forecast incremental state changes rather than absolute actions, are particularly useful [31], [32].

Prior work has shown that delta-based control significantly improves convergence speed, reduces overshooting, and also enhances robustness in manipulation and navigation tasks [33]. These types of controllers are typically trained using supervised learning or reinforcement learning and also operate efficiently at inference time [34]. Neural network-based trajectory tracking for parallel robots has demonstrated that adaptive learning can also compensate for modeling uncertainties and external disturbances [35], [36].

In our proposed technique, the neural delta controller has a crucial role in grounding symbolic decisions which are generated by the LLM. Rather than focusing on replacing the LLM, the controller complements it by making sure that each high-level decision is executed with minimal error. This strict integration especially enables our system to achieve both symbolic generalization and numerical precision. This combination is difficult to achieve with either component alone.

2.4. Spatial Reasoning and Relational Control Tasks

Spatial reasoning tasks involving relational concepts, including *left of*, *right of*, *above*, and *below,* have long been used as benchmarks for evaluating reasoning and control systems [37]. Classical approaches only rely on geometric constraints and optimization-based controllers, while learning-based methods often encode spatial relationships implicitly in the latent representations.

LLMs have shown favorable performance on symbolic spatial reasoning in purely textual domains, but also transferring this capability to physical or simulated control environments remains challenging [38]. The prior studies report that especially LLMs struggle with consistent spatial grounding, often producing actions that violate geometric constraints or converge slowly [39].

Our current work provides a systematic evaluation of spatial relational control using local LLMs, revealing

consistent performance gaps in LLM-only settings. By incorporating a special neural execution module, we demonstrate that spatial reasoning can be reliably converted into precise control behavior, even when using relatively small local language models.

2.5. Local and Resource-Constrained Large Language Models

Most of the previous work on LLM-based robotics depends on large proprietary models accessed via cloud APIs [40], [41]. While effective, such approaches raise concerns regarding reproducibility, latency, privacy, and deployment feasibility [42]. Recently, there has been growing interest in local LLMs that can run on consumer-grade hardware, driven by advances in model compression, quantization, and efficient architectures [43].

Local models such as Mistral [44], Phi [45], and LLaMA variants [46] provide a lower inference costs and also improved controllability but typically exhibit weaker reasoning performance compared to larger models [47]. These such models have been optimized for edge deployment scenarios, also including on-device translation, offline assistants, and autonomous robotics [48]. Few studies have also evaluated how this architectural integration with neural controllers can compensate for these limitations.

In our proposed work, we directly address this gap by conducting a comparative study across multiple local LLMs and demonstrating that the proposed neuro-symbolic architecture yields consistent gains regardless of model capacity. These findings suggest that architectural design is as significant as model scale in LLM-based control systems.

Previous research has explored LLMs for planning, neuro-symbolic reasoning, and learning-based control in isolation. However, current approaches either depend on open-loop reasoning, handcrafted symbolic planners, or also cloud-scale LLMs. Our work differs by the following points:

- Employing the local LLMs as symbolic planners,
- Embedding them within a closed-loop neuro-symbolic control architecture, and
- Demonstrating quantitative efficiency improvements in the spatial control tasks through tight integration with a neural delta controller.

These positions all show our current approach as a practical and scalable solution for grounded, interpretable, and also efficient LLM-based control.

3. Methodology

In addition to setting the problem and describing the design of each system component, this part provides the specific range of the suggested neuro-symbolic control that is our proposed framework. The methodology differentiates between symbolic reasoning and continuous control while focusing on the modularity, interpretability, and efficiency.

3.1. Problem Formulation

We consider a planar manipulation environment containing the following two entities:

- A *reference marker* (blue),
- A *target marker* (red).

The objective is to reposition the target marker such that it satisfies a specified spatial relation regarding the reference marker, as described by a natural language instruction.

Let $\mathcal{W} \subset \mathbb{R}^2$ denote the bounded workspace. At discrete time step $t \in \{0, 1, \dots, T\}$, the environment state is represented as:

$$\mathbf{s}_t = (x_r^t, y_r^t, x_b^t, y_b^t)^\top \in \mathcal{S} \subseteq \mathbb{R}^4, \tag{1}$$

where (x_r^t, y_r^t) denote the Cartesian coordinates of the target marker and (x_b^t, y_b^t) denote those of the reference marker.

The workspace is bounded by:

$$\mathcal{W} = \{(x, y) \in \mathbb{R}^2 : 0 \leq x \leq C,\ 0 \leq y \leq C\}, \quad (2)$$

where $C \in \mathbb{R}^+$ is the maximum coordinate value. The state space is thus defined as the Cartesian product $\mathcal{S} = \mathcal{W} \times \mathcal{W}$. Here, the system state explicitly represents the Cartesian positions of both the target and reference markers within a bounded planar workspace. This formulation provides a minimal yet sufficient representation for spatial relational tasks while avoiding unnecessary state complexity. The bounded workspace constraint ensures that all control actions remain physically valid throughout closed-loop execution.

Each episode is associated with a task $\mathcal{T}$ specified in natural language and mapped to one of four canonical spatial relations:

$$\mathcal{T} \in \Omega = \{\texttt{right\_of}, \texttt{left\_of}, \texttt{above}, \texttt{below}\}. \quad (3)$$

A margin parameter $m \in \mathbb{R}^+$ defines tolerance for task satisfaction. We formalize the task satisfaction conditions using the predicate $\mathcal{G} : \mathcal{S} \times \Omega \to \{0, 1\}$:

$$\mathcal{G}(\mathbf{s}_t, \mathcal{T}) = \begin{cases} \mathbf{1}[x_r^t \geq x_b^t + m] & \mathcal{T} = \texttt{right\_of}, \\ \mathbf{1}[x_r^t \leq x_b^t - m] & \mathcal{T} = \texttt{left\_of}, \\ \mathbf{1}[y_r^t \leq y_b^t - m] & \mathcal{T} = \texttt{above}, \\ \mathbf{1}[y_r^t \geq y_b^t + m] & \mathcal{T} = \texttt{below}, \end{cases} \quad (4)$$

where $\mathbf{1}[\cdot]$ denotes the indicator function. The task space is restricted to canonical spatial relations, allowing the symbolic reasoning module to operate over a discrete and interpretable output space. The goal predicate $\mathcal{G}(\cdot)$ uses margin-based geometric constraints, introducing tolerance against small positional errors and supporting stable convergence near task boundaries.

The action space consists of incremental displacements bounded by maximum step size δ_{max}:

$$\mathcal{A} = \{(\Delta x, \Delta y) \in \mathbb{R}^2 : |\Delta x| \leq \delta_{\text{max}},\ |\Delta y| \leq \delta_{\text{max}}\}. \quad (5)$$

Given an initial state $\mathbf{s}_0 \sim \mathcal{P}_0$ and task $\mathcal{T} \sim \mathcal{P}_{\mathcal{T}}$, the goal is to compute a control policy $\pi : \mathcal{S} \times \Omega \to \mathcal{A}$ satisfying three requirements. First, task satisfaction must be achieved within a finite horizon T:

$$\exists\, t^* \leq T : \mathcal{G}(\mathbf{s}_{t^*}, \mathcal{T}) = 1. \quad (6)$$

Second, the expected number of control steps is minimized:

$$\pi^* = \arg\min_{\pi}\ \mathbb{E}_{\mathbf{s}_0, \mathcal{T}} \left[\sum_{t=0}^{T-1} (1 - \mathcal{G}(\mathbf{s}_t, \mathcal{T})) \right]. \quad (7)$$

Third, the policy must be robust to variations in language models and initial configurations. The action space is defined in terms of bounded incremental displacements rather than absolute position commands. This choice prevents large, destabilizing movements and encourages smooth trajectories. The optimization objective prioritizes early task satisfaction, thereby directly minimizing control effort and convergence time in expectation.

3.2. Overall Neuro-Symbolic Architecture

The proposed framework decomposes decision-making into two interacting layers:

- **Symbolic reasoning layer** $\pi_{\text{sym}} : \mathcal{S} \times \Omega \to \mathcal{Z}$: responsible for task interpretation and semantic guidance.
- **Neural execution layer** $\pi_{\text{neu}} : \mathcal{S} \times \mathcal{Z} \to \mathcal{A}$: responsible for continuous control and motion refinement.

The composite policy is expressed as:

$$\pi(\mathbf{s}_t, \mathcal{T}) = \pi_{\text{neu}}\left(\mathbf{s}_t, \pi_{\text{sym}}(\mathbf{s}_t, \mathcal{T})\right), \quad (8)$$

where $\mathcal{Z}$ denotes the symbolic latent space encoding task semantics. This equation formalizes the neuro-symbolic decomposition central to the proposed framework. High-level symbolic reasoning is handled independently by π_{sym}, while continuous execution is delegated to the neural controller π_{neu}. This explicit separation allows semantic interpretation and numerical control to be managed by specialized modules, thereby improving control stability, interpretability, and overall robustness.

This separation follows the principle that symbolic abstractions and also the continuous dynamics should be managed by specialized modules rather than a monolithic policy.

3.3. Symbolic Reasoning Layer

The symbolic layer implements a locally deployed large language model (LLM) $\mathcal{M}_{\text{LLM}}$ to process the natural language instruction and also reason about the desired spatial relationship.

The LLM receives a structured prompt at each time step t as follows:

$$\mathbf{q}_t = \langle \mathcal{T}_{\text{nl}}, \psi(\mathbf{s}_t) \rangle, \tag{9}$$

where the natural language task description is denoted as $\mathcal{T}_{\text{nl}}$ and $\psi : \mathcal{S} \to \Sigma^*$ is a serialization function mapping states to structured text over alphabet Σ.

To ensure the validity of the LLM output, it is constrained to strictly formatted JSON responses as follows:

$$\mathbf{z}_t = \text{Parse}\left(\mathcal{M}_{\text{LLM}}(\mathbf{q}_t)\right) \in \mathcal{Z}, \tag{10}$$

where $\mathcal{Z} = \{0, 1, 2, 3\}$ corresponds to the four canonical spatial relations. The LLM receives a structured prompt combining the natural language task description with a serialized representation of the current state. Importantly, the LLM output is constrained to a fixed symbolic space through structured parsing. This restriction prevents hallucinated continuous actions and ensures that the language model influences control only through interpretable, discrete decisions.

The LLM does not produce continuous actions directly. This constraint prevents instability, hallucinated coordinates, and non-physical behaviors commonly observed in LLM-only control.

The framework supports interchangeable local LLMs:

$$\mathcal{M}_{\text{LLM}} \in \{\text{Mistral}, \text{Phi}, \text{LLaMA-3.2}\}, \tag{11}$$

without retraining the execution policy. This design enables systematic evaluation of language reasoning quality while keeping the control layer fixed. By allowing interchangeable local language models without retraining the neural controller, the framework enables a controlled evaluation of symbolic reasoning quality. This design isolates the effect of the language model from low-level execution dynamics, making performance differences directly attributable to semantic inference rather than control capacity.

3.4. Neural Delta Controller

The execution layer is implemented as a feedforward neural network $f_\theta : \mathbb{R}^d \to \mathbb{R}^2$ with parameters θ, trained to predict incremental displacements toward task satisfaction.

3.4.1. Input Encoding

The controller input at time t is constructed by concatenating normalized state features with the task encoding:

$$\mathbf{u}_t = \left[\frac{x_r^t}{C}, \frac{y_r^t}{C}, \frac{x_b^t}{C}, \frac{y_b^t}{C}, \phi(\mathcal{T})\right]^\top \in \mathbb{R}^5, \tag{12}$$

where $\phi : \Omega \to [0, 1]$ is a scalar encoding of the task relation defined as:

$$\phi(\mathcal{T}) = \frac{\text{idx}(\mathcal{T})}{|\Omega| - 1}, \tag{13}$$

with $\text{idx}(\cdot)$ returning the index of the task in Ω.

Alternatively, a one-hot encoding $\phi_{\text{oh}} : \Omega \to \{0, 1\}^{|\Omega|}$ may be used:

$$\phi_{\text{oh}}(\mathcal{T}) = \mathbf{e}_{\text{idx}(\mathcal{T})} \in \{0, 1\}^4, \tag{14}$$

where $\mathbf{e}_i$ denotes the i-th standard basis vector, yielding input dimension $d = 8$. The neural controller input combines normalized state features with a compact encoding of the symbolic task label. Normalization ensures numerical stability during training and inference, while the task encoding provides explicit semantic conditioning. Both scalar and one-hot encodings are supported, allowing flexibility without altering the controller architecture.

3.4.2. Network Architecture

The neural controller consists of L fully-connected layers:

$$f_\theta(\mathbf{u}) = \mathbf{W}_L\,\sigma\left(\mathbf{W}_{L-1}\,\sigma\left(\cdots\sigma\left(\mathbf{W}_1\mathbf{u}+\mathbf{b}_1\right)\cdots\right)+\mathbf{b}_{L-1}\right)+\mathbf{b}_L, \tag{15}$$

where $\sigma(\cdot) = \max(0,\cdot)$ is the ReLU activation, $\mathbf{W}_l \in \mathbb{R}^{n_l \times n_{l-1}}$ are weight matrices, and $\mathbf{b}_l \in \mathbb{R}^{n_l}$ are bias vectors. The neural execution module is implemented as a feedforward network that maps state-task representations to incremental control actions. ReLU activations enable efficient learning of nonlinear control policies, while the fully connected structure ensures low inference latency suitable for closed-loop operation.

The output layer applies hyperbolic tangent activation to bound displacements:

$$(\Delta x_t, \Delta y_t) = \delta_{\max} \cdot \tanh\left(f_\theta(\mathbf{u}_t)\right) \in [-\delta_{\max}, \delta_{\max}]^2. \tag{16}$$

The hyperbolic tangent activation enforces strict bounds on the predicted displacements, preventing excessive control actions. This bounded delta formulation reduces overshooting and oscillations commonly observed in LLM-only control, leading to smoother trajectories and more stable convergence.

3.4.3. Output and State Transition

The predicted displacement updates the target marker position via the transition function $\mathcal{F} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$:

$$x_r^{t+1} = \text{clip}(x_r^t + \Delta x_t, 0, C), \tag{17}$$

$$y_r^{t+1} = \text{clip}(y_r^t + \Delta y_t, 0, C), \tag{18}$$

where the clipping function enforces workspace boundaries:

$$\text{clip}(z, a, b) = \min(\max(z, a), b). \tag{19}$$

The state transition updates only the target marker position while enforcing workspace constraints through clipping. This ensures that the system remains within valid spatial limits throughout execution. Combined with bounded actions, this transition model contributes to safe and predictable closed-loop behavior.

Delta-based control reduces overshooting and improves stability compared to absolute coordinate prediction. As shown in Figure 2, the proposed framework decouples symbolic reasoning from continuous control by delegating language understanding to a local LLM and motion execution to a neural delta controller.

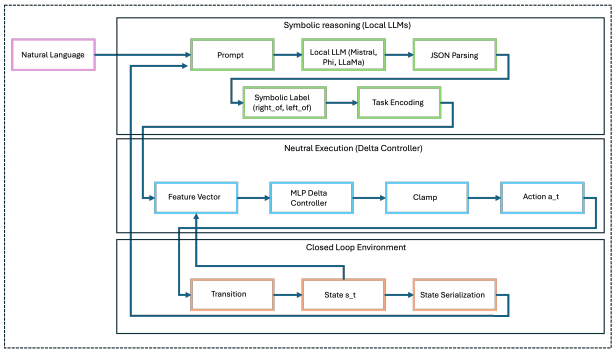

Figure 2. Overview of the proposed neuro-symbolic control framework. A local large language model performs symbolic reasoning over language instructions and environment state, producing a discrete task label. This symbolic output conditions a neural delta controller that executes bounded continuous actions in a closed-loop environment, enabling stable, efficient, and interpretable control.

3.5. Training Procedure

The neural controller is trained using supervised learning on synthetically generated trajectories.

3.5.1. Dataset Generation

Each training sample consists of a state-task-displacement tuple:

$$\mathcal{D} = \left\{\left(\mathbf{s}^{(i)}, \mathcal{T}^{(i)}, \Delta\mathbf{p}^{*(i)}\right)\right\}_{i=1}^{N}, \tag{20}$$

where $\mathbf{s}^{(i)} \sim \text{Uniform}(\mathcal{S})$, $\mathcal{T}^{(i)} \sim \text{Uniform}(\Omega)$, and $\Delta\mathbf{p}^{*(i)} = (\Delta x^*, \Delta y^*)$ is the target displacement that moves the target marker closer to task satisfaction.

3.5.2. Target Displacement Computation

The optimal displacement is computed as the direction toward the goal region scaled by step size $\alpha \in (0, 1]$. For $\mathcal{T} = \texttt{right\_of}$:

$$\Delta x^* = \alpha \cdot \max(0, (x_b + m) - x_r), \quad \Delta y^* = 0. \quad (21)$$

Analogous expressions hold for the remaining spatial relations. For `left_of`: $\Delta x^* = -\alpha \cdot \max(0,\, x_r - (x_b - m))$, $\Delta y^* = 0$. For `above`: $\Delta x^* = 0$, $\Delta y^* = -\alpha \cdot \max(0,\, y_r - (y_b - m))$. For `below`: $\Delta x^* = 0$, $\Delta y^* = \alpha \cdot \max(0,\, (y_b + m) - y_r)$.

3.5.3. Loss Function

The mean squared error between the target and forecast displacements defines the loss function:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left\| f_\theta(\mathbf{u}^{(i)}) - \Delta \mathbf{p}^{*(i)} \right\|_2^2. \quad (22)$$

To encourage smooth trajectories, an optional regularization term penalizes large displacement magnitudes:

$$\mathcal{L}_{\text{reg}}(\theta) = \mathcal{L}(\theta) + \lambda \cdot \frac{1}{N} \sum_{i=1}^{N} \left\| f_\theta(\mathbf{u}^{(i)}) \right\|_2^2, \quad (23)$$

where $\lambda \geq 0$ is the regularization coefficient. This encourages smooth trajectories and reduces large-magnitude displacement predictions.

3.5.4. Optimization

Parameters θ are optimized using stochastic gradient descent with the Adam optimizer:

$$\theta_{k+1} = \theta_k - \eta \cdot \frac{\hat{\mathbf{m}}_k}{\sqrt{\hat{\mathbf{v}}_k} + \epsilon}, \quad (24)$$

where the learning rate is denoted by η, $\hat{\mathbf{m}}_k$ and $\hat{\mathbf{v}}_k$ are bias-corrected moment estimates, and ϵ is a small constant for numerical stability.

This formulation advocates a smooth, task-directed motion and more generalization across tasks.

3.6. Closed-Loop Control Strategy

The closed-loop operation of the system is governed by Algorithm 3:

Algorithm 1: Neuro-Symbolic Closed-Loop Control

Input: $\mathbf{s}_0$, $\mathcal{T}$, horizon T, LLM $\mathcal{M}_{\text{LLM}}$, controller f_θ
Output: $\mathbf{s}_{t^*}$, success flag, step count

$t \leftarrow 0$
while $t < T$ **and** $\mathcal{G}(\mathbf{s}_t, \mathcal{T}) = 0$ **do**
  $\mathbf{q}_t \leftarrow \langle \mathcal{T}_{\text{nl}},\, \psi(\mathbf{s}_t) \rangle$ *// Construct prompt*
  $\mathbf{z}_t \leftarrow \text{Parse}(\mathcal{M}_{\text{LLM}}(\mathbf{q}_t))$ *// Symbolic reasoning*
  $\mathbf{u}_t \leftarrow [\mathbf{s}_t / C,\ \phi(\mathbf{z}_t)]^\top$ *// Encode input*
  $(\Delta x_t, \Delta y_t) \leftarrow f_\theta(\mathbf{u}_t)$ *// Compute action*
  $\mathbf{s}_{t+1} \leftarrow \mathcal{F}(\mathbf{s}_t, (\Delta x_t, \Delta y_t))$ *// Update state*
  $t \leftarrow t + 1$
end while
return $\mathbf{s}_t,\ \mathcal{G}(\mathbf{s}_t, \mathcal{T}),\ t$

Figure 3. Neuro-Symbolic Closed-Loop Control algorithm.

This feedback loop enables online correction and robust convergence even under imperfect symbolic reasoning.

3.7. Distance-to-Goal Metric

To analyze convergence behavior, we define a distance-to-goal metric $d : \mathcal{S} \times \Omega \to \mathbb{R}_{\geq 0}$. For each spatial relation, d measures the remaining gap to the satisfaction boundary:

$$d(\mathbf{s}_t, \mathcal{T}) = \begin{cases} \max(0,\, (x_b^t + m) - x_r^t) & \mathcal{T} = \texttt{right\_of}, \\ \max(0,\, x_r^t - (x_b^t - m)) & \mathcal{T} = \texttt{left\_of}, \\ \max(0,\, y_r^t - (y_b^t - m)) & \mathcal{T} = \texttt{above}, \\ \max(0,\, (y_b^t + m) - y_r^t) & \mathcal{T} = \texttt{below}. \end{cases} \quad (25)$$

This metric satisfies non-negativity ($d \geq 0$), goal identification ($d = 0 \Leftrightarrow \mathcal{G} = 1$), and continuity in $\mathbf{s}_t$.

For cross-episode comparison, we define the normalized distance:

$$\bar{d}_t = \frac{d(\mathbf{s}_t, \mathcal{T})}{d(\mathbf{s}_0, \mathcal{T}) + \epsilon}, \quad (26)$$

where $\epsilon > 0$ ensures numerical stability when $d(\mathbf{s}_0, \mathcal{T}) \approx 0$. Normalization enables fair comparison across episodes

with different starting configurations and is used to analyze convergence speed and stability across control strategies.

4. Experimental Setup and Results

This section demonstrated the details of the experimental evaluation of the proposed neuro-symbolic control framework. We establish the baseline processes, evaluation metrics, experimental design, and quantitative results. We designed the experiments in such a way that they ensure reproducibility, controlled ablation, and statistically significant comparisons between control strategies and language models.

4.1. Environment Configuration

All experiments are conducted in a planar simulated environment containing two point markers: a movable target marker (red) and a static reference marker (blue). The workspace is defined as a bounded square region $\mathcal{W} = [0, C]^2 \subset \mathbb{R}^2$, with $C = 800$ pixels.

At the beginning of each episode, the positions of the target and reference markers are independently sampled from a uniform distribution over the workspace. Episodes are executed for a fixed control horizon T, and the episode terminates early if the task satisfaction condition is met. All experiments use identical workspace bounds and initialization procedures to ensure consistent evaluation across methods.

4.1.1. Task Specification

We evaluate four canonical spatial relations: $\{\texttt{right\_of}, \texttt{left\_of}, \texttt{above}, \texttt{below}\}$. A task is considered successful if the corresponding spatial constraint is satisfied within a tolerance margin of $m = 50$ pixels at any time step $t \leq T$.

Each task is provided to the system in natural language form. For example, the instruction "move the red object to the right of the blue object" corresponds to the `right_of` relation. The same task set and margin values are used across all compared control strategies.

4.1.2. Compared Methods

We compare three control strategies to isolate the contribution of symbolic reasoning and neural execution.

LLM-Only Control.. In this baseline, a large language model directly predicts absolute target coordinates at each control step based on the current state and the natural language instruction. The resulting displacement is computed as the difference between the predicted and current target positions. No learned motion prior or bounded control mechanism is applied.

DL-Only Control.. This baseline uses the neural delta controller exclusively, conditioned on a fixed oracle task encoding corresponding to the ground-truth spatial relation. No language model inference is performed during execution. This configuration serves as an upper bound on execution performance when task semantics are perfectly specified.

LLM+DL (Proposed).. The proposed method combines symbolic reasoning and neural execution. At each control step, a local language model infers a symbolic task label from the natural language instruction and current state. This symbolic output conditions the neural delta controller, which generates bounded incremental displacements in a closed-loop manner.

All methods share the same environment, control horizon, and evaluation protocol.

4.1.3. Large Language Models

Symbolic reasoning is performed using locally deployed large language models to ensure reproducibility, low-latency inference, and independence from external APIs. The following models are evaluated:

- **Mistral-7B**, a 7B-parameter instruction-tuned model,
- **Phi-2**, a 2.7B-parameter compact model optimized for

efficiency,

- **LLaMA-3.2**, an open-weight model with strong general reasoning capability.

The neural delta controller f_θ is kept fixed across all experiments. This design isolates the effect of symbolic reasoning quality and ensures that performance differences arise from language model inference rather than changes in low-level control dynamics.

4.1.4. Evaluation Metrics

For each combination of our proposed method, language model, and also the task, we conduct a comprehensive $N = 20$ independent episodes with the randomized initial configurations to show the strong results. We report the mean performance with standard deviation to quantify variability of the proposed framework.

We assess performance using the following metrics [49], [50]:

Success Rate (SR). The fraction of episodes achieving task satisfaction within the time horizon:

$$\mathrm{SR} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\left[\exists\, t \le T : \mathcal{G}(\mathbf{s}_t^{(i)}, \mathcal{T}) = 1\right]. \tag{27}$$

Average Steps (AS). The mean number of control steps required for successful episodes:

$$\mathrm{AS} = \frac{1}{|\mathcal{S}_{\text{succ}}|}\sum_{i \in \mathcal{S}_{\text{succ}}} t_i^*, \tag{28}$$

where $\mathcal{S}_{\text{succ}}$ denotes the set of successful episodes and t_i^* is the termination step.

Normalized Distance-to-Goal. The task-specific geometric distance to the satisfaction boundary, normalized by initial distance:

$$\bar{d}_t = \frac{d(\mathbf{s}_t, \mathcal{T})}{d(\mathbf{s}_0, \mathcal{T}) + \epsilon}. \tag{29}$$

Relative Improvement Metrics. To quantify the benefit of neuro-symbolic integration, we compute:

- **Success Rate Improvement**: $\Delta\mathrm{SR} = \mathrm{SR}_{\text{LLM+DL}} - \mathrm{SR}_{\text{LLM-only}}$
- **Step Reduction**: $\rho = \dfrac{\mathrm{AS}_{\text{LLM-only}} - \mathrm{AS}_{\text{LLM+DL}}}{\mathrm{AS}_{\text{LLM-only}}} \times 100\%$
- **Speedup Factor**: $\sigma = \dfrac{\mathrm{AS}_{\text{LLM-only}}}{\mathrm{AS}_{\text{LLM+DL}}}$

4.2. Results

In this section, we discussed the results obtained on different LLM models and what are the key improvements and observations emerged from these experiments. In Figure 4, we can see that the proposed LLM+DL framework consistently achieves higher success rates across all the spatial tasks when it is compared to both LLM-only and DL-only baselines frameworks.

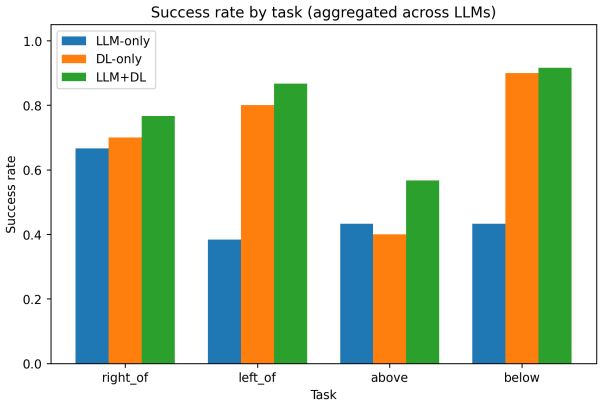

Figure 4. Success rate aggregated across all language models for each spatial task. The proposed LLM+DL framework consistently outperforms LLM-only and DL-only baselines, demonstrating the effectiveness of neuro-symbolic integration.

The average number of steps and success rate for each technique, task, and language model for the `right_of` task are summarized in Table 1. The supplemental material contains the full results for each task.

Several key observations emerged from the results. First, for consistent improvement, across all language models, the suggested LLM+DL framework produces success rates that are either greater or equivalent to those of LLM-only control, with significant drops in average steps. Second, for efficiency gains, depending on the underlying language model, the neural delta controller allows conver-

Table 1. Performance comparison across control methods and LLMs.

LLM Model	Method	Task	Success Rate	Avg. Steps
–	DL-only	right_of	0.70	1.35
Mistral	LLM-only	right_of	0.70	3.60
Mistral	LLM+DL	right_of	0.70	1.15
Phi	LLM-only	right_of	0.60	4.95
Phi	LLM+DL	right_of	0.85	0.85
LLaMA-3.2	LLM-only	right_of	0.70	3.40
LLaMA-3.2	LLM+DL	right_of	0.75	1.35

gence in many fewer steps, with speedups ranging from $2.52\times$ to $5.82\times$. The third most significant observation was compensation for weak reasoning, for which the Phi model shows the best LLM+DL success rate (0.85) but the lowest LLM-only success rate (0.60), indicating that the neural controller successfully compensates for weaker symbolic reasoning.

According to Figure 5, LLM+DL typically reduces the number of necessary control steps by more than 70%. Each method's unique qualities are shown by the convergence behavior:

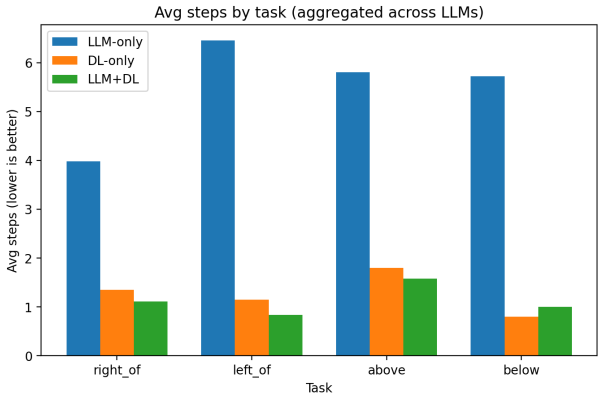

Figure 5. Total average number of control steps for all language models. Compared to LLM-only control, the LLM+DL framework converges far more quickly, resulting in a step reduction of more than 70% for all tasks.

- For the LLM-only, it shows oscillatory behavior as a result of inaccurate coordinate predictions and sluggish convergence with large variance. The distance-to-goal curve exhibits numerous reversals and non-monotonic descent.
- For the DL-only, when given oracle task encoding, it converges quickly and monotonically. Nevertheless, new task descriptions or variations in natural language cannot be accommodated by this setup.
- For the LLM+DL, it retains the capacity to understand normal language instructions while achieving quick, consistent convergence on par with DL-only. While symbolic reasoning offers high-level semantic guidance, the neural delta controller guarantees smooth paths.

Table 2. Relative improvement of LLM+DL over LLM-only (mean $\pm$ std).

Task	Δ Success	Step Reduction (%)	Speedup ($\times$)
right_of	0.10 ± 0.13	70.4 ± 11.4	3.82 ± 1.76
left_of	0.48 ± 0.23	85.9 ± 7.7	8.83 ± 4.93
above	0.13 ± 0.25	72.4 ± 9.2	3.87 ± 1.09
below	0.48 ± 0.10	82.4 ± 3.3	5.81 ± 1.07

The normalized distance-to-goal over time is shown in Figure 6, which highlights the LLM+DL framework's faster and more stable convergence.

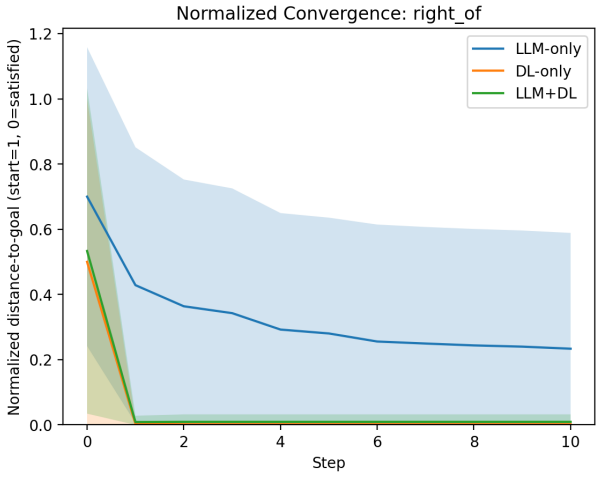

Figure 6. Normalized distance-to-goal over time for the `right_of` task. LLM+DL exhibits fast, monotonic convergence with low variance, whereas the LLM-only control shows slower and less stable behavior.

To quantify the benefit of neuro-symbolic integration across all tasks, Table 2 reports aggregated statistics comparing LLM+DL to LLM-only control. Results are averaged across all three language models.

For the significant speedup, for all tasks, the suggested method delivers an average speedup of $5.58\times$, with a maximum speedup of $8.83\times$ for the `left_of` relation. For the task-dependent gains, compared to orthogonal relations (`right_of`, `above`), lateral relations (`left_of`,

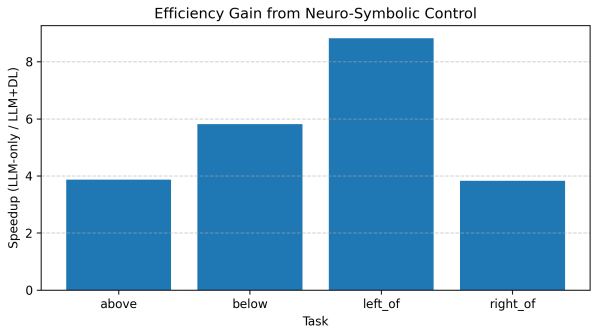

Figure 7. Speedup of LLM+DL relative to LLM-only control aggregated across language models. The proposed framework achieves substantial efficiency gains, with speedups of up to $8.83\times$ depending on the spatial relation.

`below`) show greater increases in speedup and success rate ($\Delta\text{SR} = 0.48$). This imbalance implies that some spatial interactions are more difficult for LLM-only management because of incorrect or hallucinated coordinate predictions. Regardless of the kind of work, step reduction consistently surpasses 70%, indicating that the neural delta controller significantly increases control efficiency.

As summarized in Figure 7, the proposed method achieves speedups of up to $8.83\times$ over LLM-only control.

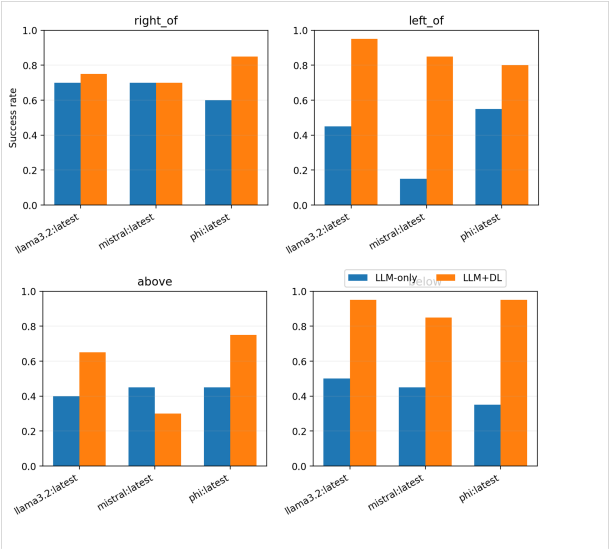

Figure 8. Success rate by language model and task. The suggested LLM+DL framework consistently improves performance on Mistral, Phi, and LLaMA-3.2, demonstrating resilience to scale and language model selection.

As the performance gains are consistent across configurations, several notable observations emerge:

- For the model-agnostic improvement, the LLM+DL framework enhances all three models' performance, suggesting that architectural decomposition, rather than language model capacity or scale, is the source of the advantages.
- For the compensation effect, even in situations where LLM-only control operates poorly, robust performance is made possible by the neural controller's compensation for inferior symbolic reasoning. This is especially true for Phi, where LLM+DL outperforms the model even with fewer parameters.
- Due to the decreased variance, the LLM+DL performs more consistently across episodes and beginning settings, as seen by a significantly lower standard deviation of success rates than LLM-only.

The performance improvements of the neuro-symbolic framework hold true for all assessed local LLMs, as shown in Figure 8.

4.2.1. Ablation: Component Contribution

We calculate the performance difference between approaches in order to further isolate each component's contribution:

$$\Delta_{\text{symbolic}} = \text{SR}_{\text{LLM+DL}} - \text{SR}_{\text{DL-only}}, \tag{30}$$

$$\Delta_{\text{neural}} = \text{SR}_{\text{LLM+DL}} - \text{SR}_{\text{LLM-only}}. \tag{31}$$

Averaged across all configurations, we observe $\Delta_{\text{symbolic}} = 0.12$ and $\Delta_{\text{neural}} = 0.30$, indicating that both components contribute meaningfully to overall performance, with the neural controller providing the larger marginal improvement. We can infer the following conclusions from the experimental results:

- Control using LLM alone is ineffective: Particularly for spatial manipulation tasks requiring numerical precision, direct coordinate prediction by LLMs results in sluggish convergence, significant variance, and inconsistent behavior.

- DL-only control lacks semantic flexibility: While the neural controller converges rapidly with oracle task encoding, it cannot interpret natural language instructions or generalize to novel task formulations.
- Neuro-symbolic integration works well: By combining the complementing advantages of neural control and symbolic thinking, the suggested LLM+DL paradigm achieves the following:
  - Up to $8.83\times$ speedup over LLM-only control
  - Average success rate improvement of 0.30
  - Consistent gains across all language models and tasks
  - Reduced variance and improved robustness
- Architectural design matters: The neuro-symbolic decomposition offers advantages independent of underlying model capability, as evidenced by the performance gains that are consistent across language models of different scales.

These results support the main hypothesis that more effective, reliable, and comprehensible behavior is produced in language-conditioned manipulation tasks when symbolic thinking and continuous control are separated.

5. Discussion

5.1. Deterministic Stability vs Motion Fluidity

The symbolic discretization mainly prioritizes deterministic and stable execution by preventing hallucinated actions, but it may produce less fluid trajectories in some cases. This trade-off is intentional for the purpose of safety and convergence, and the framework can further be extended to finer symbolic resolutions (e.g., headings or intensities) while preserving bounded control.

5.2. Semantic Limits of Local Language Models

Locally deployed language models may introduce a lower semantic ceiling than cloud-scale options, specifically for abstract or compositional instructions. Since the neural controller cannot correct incorrect symbolic intent, but it does stabilize execution once a symbolic decision is made, and richer symbolic reasoning can be incorporated as per the need.

5.3. Robustness to Sensor Noise and State Uncertainty

The evaluation presented in the paper assumes ground-truth state information to isolate control behavior. In the presence of moderate sensor noise, the closed-loop delta controller can modify and correct transient symbolic errors as state estimates are updated, while persistent uncertainty would require additional perceptual filtering.

5.4. Generalization Beyond 2D Environments

Although evaluated in a planar 2D setting, the neuro-symbolic decomposition presented in this paper is not inherently restricted to only two dimensions. Extension to 3D environments would involve increasing the state representation and symbolic predicate set. Irregular object shapes can be accommodated through geometry-aware symbolic abstractions. This modular design facilitates such extensions without changing the core execution controller.

6. Conclusion

In this paper a neuro-symbolic control framework for language-guided spatial manipulation that explicitly separates high-level symbolic reasoning from low-level continuous execution is presented. By constraining the language model to symbolic task outputs and delegating numerical control to a neural delta controller, the proposed architecture addresses key limitations of monolithic LLM-based control, including unstable trajectories, slow convergence, and hallucinated actions. The experimental results demonstrate that this decomposition leads to more stable and efficient closed-loop behavior.

Across all evaluated spatial tasks and locally deployed language models, the proposed LLM+DL framework con-

sistently outperforms LLM-only baselines in convergence efficiency while preserving or improving task success rates. Average step reductions exceed 70%, with speedups of up to $8.83\times$, and these gains remain consistent even for smaller language models with weaker symbolic reasoning capabilities. These findings indicate that the observed performance improvements arise primarily from architectural design rather than language model scale, highlighting the effectiveness of neuro-symbolic integration for reliable and interpretable control.

Despite these advantages, the current study is limited to planar environments with perfect state observability and a fixed set of primitive spatial relations. The expressiveness of the symbolic layer is constrained by the predefined task vocabulary, and stochastic language model outputs and inference latency remain practical considerations. Future work will extend this framework to real-world robotic platforms, richer symbolic representations, hierarchical and compositional task structures, and vision-based environments, with the goal of enabling robust language-guided control in more complex and realistic settings.

Competing Interest Statement

The authors declare no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

Data Availability Statement

Additional experimental data and analysis that corroborate the conclusions are provided in supplementary section.

References

[1] Y. Zhang et al., "Can LLM graph reasoning generalize beyond pattern memorization?" *arXiv preprint arXiv:2406.15992*, 2024.

[2] D. Bandyopadhyay, S. Bhattacharjee, and A. Ekbal, "Thinking machines: A survey of LLM-based reasoning strategies," *arXiv preprint arXiv:2503.10814*, 2025.

[3] L. Liu et al., "Optimizing task planning efficiency in LLMs: Beyond closed-loop systems," *Authorea Preprints*, 2024.

[4] S. Banerjee, A. Agarwal, and S. Singla, "LLMs will always hallucinate, and we need to live with this," in *Proceedings of the Intelligent Systems Conference*, Springer, 2025, pp. 624–648.

[5] L. Huang et al., "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Transactions on Information Systems*, vol. 43, no. 2, pp. 1–55, 2025.

[6] C. Tang, B. Abbatematteo, J. Hu, R. Chandra, R. Martín-Martín, and P. Stone, "Deep reinforcement learning for robotics: A survey of real-world successes," *Annu. Rev. Control Robot. Auton. Syst.*, vol. 8, pp. 153–178, 2025.

[7] Q. Du et al., "Fast task planning with neuro-symbolic relaxation," *arXiv preprint arXiv:2507.15975*, 2025.

[8] V. Bhat, A. U. Kaypak, P. Krishnamurthy, R. Karri, and F. Khorrami, "Grounding LLMs for robot task planning using closed-loop state feedback," *arXiv preprint arXiv:2402.08546*, 2024.

[9] W. Su, "Do large language models (really) need statistical foundations?" *arXiv preprint arXiv:2505.19145*, 2025.

[10] V. Enoasmo, C. Featherstonehaugh, X. Konstantinopoulos, and Z. Huntington, "Structural embedding projection for contextual large language model inference," *arXiv preprint arXiv:2501.18826*, 2025.

[11] S. Ullah, M. Liaqat, A. Asif, A. Khan, U. Aslam, and H. Asif, "Deep auto encoder based chatbot for discrete math course," in *Int. Conf. Recent Advances in Electrical Engineering & Computer Sciences (RAEE&CS)*, IEEE, Oct. 2022, pp. 1–7.

[12] Y. Kim, J. Choi, and S. Lee, "A survey on integration of large language models with intelligent robots," *Intelligent Service Robotics*, 2024.

[13] F. Zeng, W. Gan, Y. Wang, N. Liu, and P. S. Yu, "Large language models for robotics: A survey," *arXiv preprint arXiv:2311.07226*, 2023.

[14] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *Proceedings of ICML*, 2022.

[15] M. Ahn et al., "Do as I can, not as I say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[16] D. Driess et al., "PaLM-E: An embodied multimodal language model," 2023.

[17] J. Liang et al., "Code as policies: Language model programs for embodied control," *arXiv preprint arXiv:2209.07753*, 2022.

[18] J. Wang et al., "Large language models for robotics: Opportunities, challenges, and perspectives," *Journal of Automation and Intelligence*, 2024.

[19] H. Jeong, H. Lee, C. Kim, and S. Shin, "A survey of robot intelligence with large language models," *Applied Sciences*, vol. 14, no. 19, p. 8868, 2024.

[20] M. Abid, T. Akhtar, and H. Bhatt, "Uncertainty quantification in steady-state heat transfer: A comprehensive analysis of DRAM and MCMC methods with applications to thermal systems," *Spectrum of Engineering and Management Sciences*, vol. 3, no. 1, pp. 63–75, 2025.

[21] Y. Chen, J. Arkin, Y. Zhang, et al., "AutoTAMP: Autoregressive task and motion planning with LLMs as translators and checkers," in *Proceedings of ICRA*, 2024.

[22] A. Garcez and L. Lamb, "Neurosymbolic AI: The third wave," *Artificial Intelligence Review*, vol. 56, pp. 12 387–12 406, 2023.

[23] Z. Wan et al., "Towards cognitive AI systems: A survey and prospective on neuro-symbolic AI," *arXiv preprint arXiv:2401.01040*, 2024.

[24] L. De Raedt, S. Dumancic, R. Manhaeve, and G. Marra, "From statistical relational to neurosymbolic artificial intelligence: A survey," *Artificial Intelligence*, vol. 328, 2024.

[25] B. C. Colelough and W. Regli, "Neuro-symbolic AI in 2024: A systematic review," *arXiv preprint arXiv:2501.05435*, 2025.

[26] M. Abid and M. Saqlain, "Optimizing diabetes data insights through KMapper-based topological networks: A decision analytics approach for predictive and prescriptive modeling," *Management Science Advances*, vol. 1, no. 1, pp. 1–19, 2024.

[27] C. R. Garrett, R. Chitnis, R. M. Holladay, et al., "Integrated task and motion planning," *Annu. Rev. Control Robot. Auton. Syst.*, vol. 4, pp. 265–293, 2021.

[28] W. Zhai, J. Liao, Z. Chen, B. Su, and X. Zhao, "A survey of task planning with large language models," *Intelligent Computing*, vol. 4, p. 0124, 2025.

[29] B. Bousetouane, "Agentic LLM-based robotic systems for real-world applications: A review," *Frontiers in Robotics and AI*, 2025.

[30] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[31] L. M. Thanh, L. H. Thuong, P. T. Loc, and C.-N. Nguyen, "Delta robot control using single neuron PID algorithms based on recurrent fuzzy neural network identifiers," *Int. J. Mech. Eng. Robot. Res.*, vol. 9, no. 10, pp. 1411–1418, 2020.

[32] A. Gholami, T. Homayouni, R. Ehsani, and J. Q. Sun, "Inverse kinematic control of a delta robot using neural networks in real-time," *Robotics*, vol. 10, no. 4, p. 115, 2021.

[33] Y. Fan, H. Huang, and C. Yang, "Fixed-time incremental neural control for manipulator based on composite learning with input saturation," *Actuators*, vol. 11, no. 12, p. 373, 2022.

[34] M. dos Santos Lima, V. A. Kich, R. Steinmetz, and D. F. Tello Gamarra, "Delta robot control by learning systems: Harnessing the power of deep reinforcement learning algorithms," *Journal of Intelligent & Fuzzy Systems*, vol. 46, no. 2, pp. 4881–4894, 2024.

[35] M. Abid and M. L. Ali, "Enhancing software effort estimation: A comparative analysis of machine learning models with correlation-based feature selection," *Sustainable Machine Intelligence Journal*, vol. 12, pp. 1–17, 2025.

[36] S. Khosravi and A. Akbari, "Experimental study on a novel simultaneous control and identification of a 3-DOF delta robot using model reference adaptive control," *Mechatronics*, vol. 86, 2022.

[37] B. Chen, Z. Xu, S. Kirmani, et al., "SpatialVLM: Endowing vision-language models with spatial reasoning capabilities," in *Proceedings of CVPR*, 2024.

[38] K. Rana, J. Haviland, S. Garg, et al., "SayPlan: Grounding large language models using 3D scene graphs for scalable robot task planning," in *Proceedings of CoRL*, 2023.

[39] W. Hunt, S. D. Ramchurn, and M. D. Soorati, "A survey of language-based communication in robotics," *arXiv preprint arXiv:2406.04086*, 2024.

[40] M. Abid, S. Bukhari, and M. Saqlain, "Enhancing software effort estimation in healthcare informatics: A comparative analysis of machine learning models with correlation-based feature selection," *Sustainable Machine Intelligence Journal*, vol. 10, pp. 50–66, 2025.

[41] J. Wang et al., "Large language models for robotics: Opportunities, challenges, and perspectives," *Journal of Automation and Intelligence*, 2024.

[42] B. Amin, *Mistral expands its reach in the SLM space*, TechTalks, Oct. 2024.

[43] Y. Zheng, Y. Chen, B. Qian, X. Shi, Y. Shu, and J. Chen, "A review on edge large language models: Design, execution, and applications," *ACM Computing Surveys*, vol. 57, no. 8, pp. 1–35, 2025.

[44] D. Jiang et al., "From CLIP to DINO: Visual encoders shout in multi-modal large language models," *arXiv preprint arXiv:2310.08825*, 2023.

[45] M. Abdin et al., "Phi-4 technical report," *arXiv preprint arXiv:2412.08905*, 2024.

[46] H. Touvron et al., "LLaMA: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[47] Data Science Dojo, *Phi-3 and beyond: Top small language models of 2024*, `https://datasciencedojo.com`, 2024.

[48] Mistral AI, *Introducing les ministraux: Edge-optimized models*, `https://mistral.ai`, Oct. 2024.

[49] H. Kress-Gazit et al., "Robot learning as an empirical science: Best practices for policy evaluation," *arXiv preprint arXiv:2409.09491*, 2024.

[50] J. Faigl, M. Kulich, and L. Přeučil, "Goal assignment using distance cost in multi-robot exploration," in *Proc. IEEE/RSJ IROS*, 2012, pp. 3741–3746.

Appendix A (Supplementary Material)

Additional experimental data and analysis that corroborate the conclusions in the main paper are provided in this supplementary section. These findings provide more clear and precise information about ablation results, task-wise convergence patterns, and model-specific behavior. Every experiment adheres to the same procedure outlined in Section 4.

A1. Success Rate by Language Model and Task

Success rates for each spatial task are displayed by language model in Figure A1. The LLM+DL framework significantly increases success rates compared to LLM-only control in almost all configurations, while absolute performance differs among models.

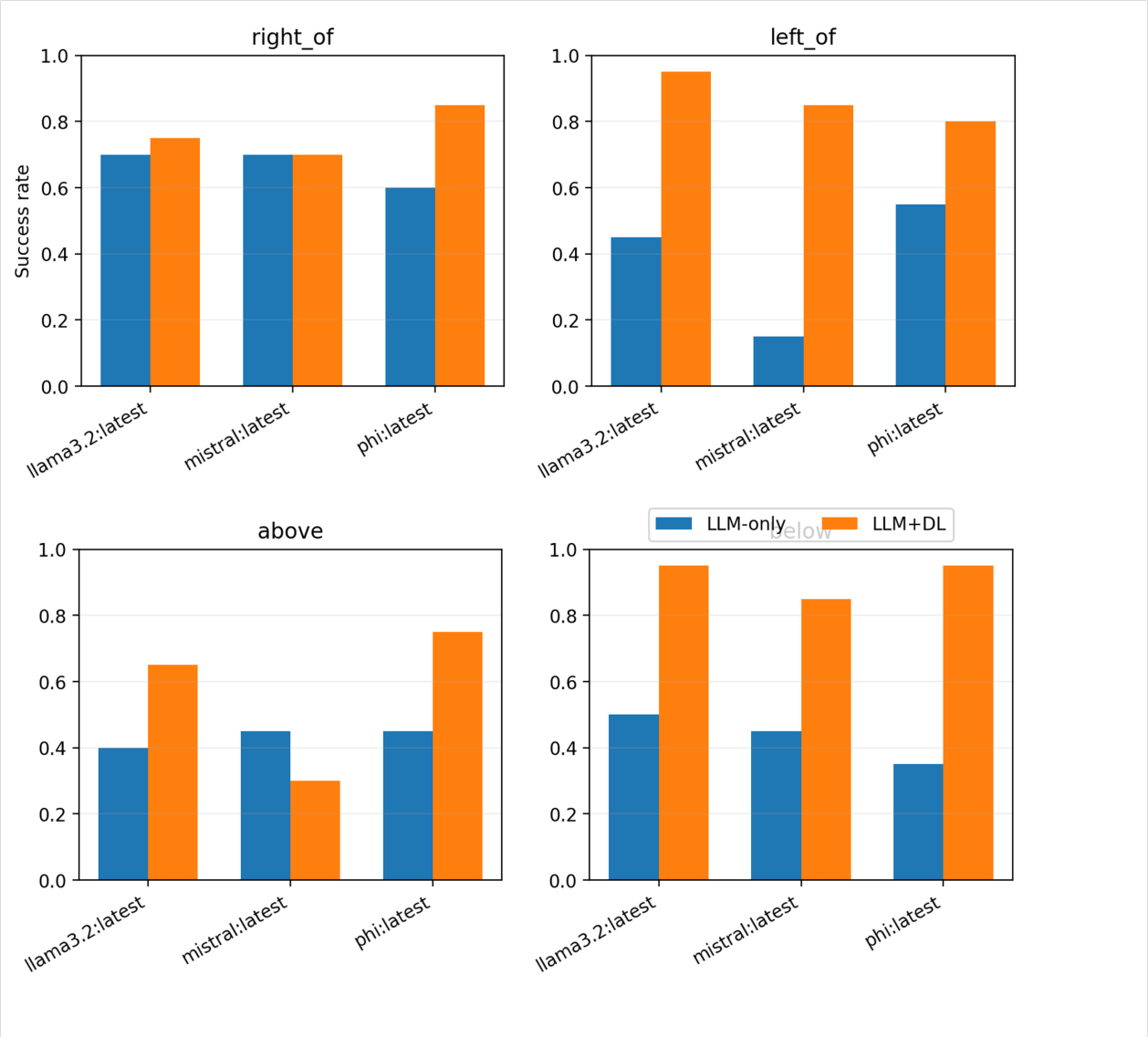

Figure A1. Success rate by task and language model. The neuro-symbolic LLM+DL framework improves reliability across all evaluated LLMs, including smaller models such as Phi.

A2. Average Steps by Language Model

Figure A2 provides an additional analysis of efficiency by breaking down the average number of steps needed for successful episodes by task and language model. While LLM+DL consistently converges faster across all models, the LLM-only control shows substantial volatility and noticeably increased step counts.

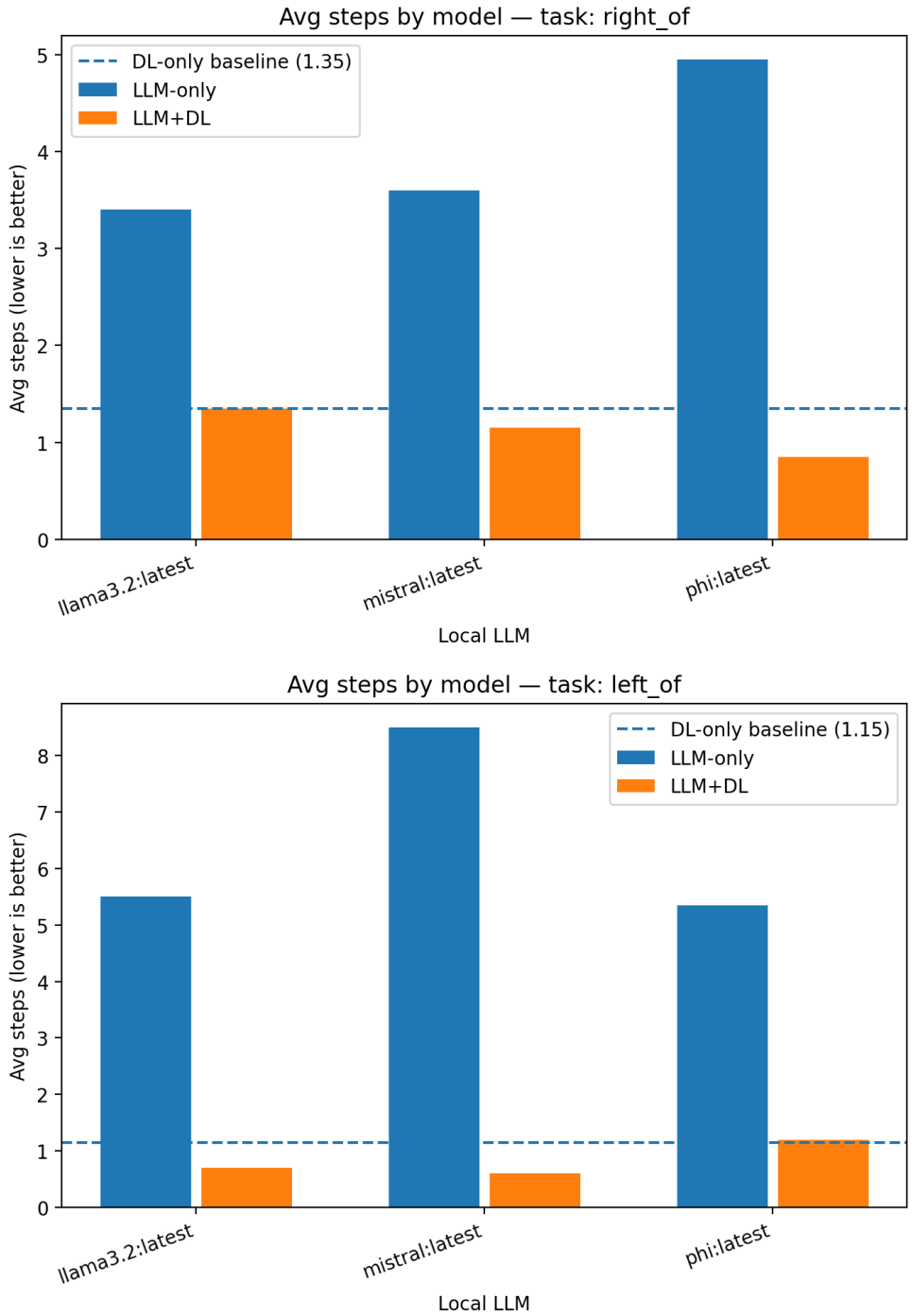

Figure A2. The average number of control steps for the `left_of` and `right_of` tasks by language model. For all models, LLM+DL achieves significant step count reductions.

A3. Convergence Behavior Across Tasks

Normalized distance-to-goal curves for each of the four spatial relations are shown in Figure A3. These figures show that while the LLM+DL framework generates smooth, monotonic trajectories with lower variance, LLM-only control frequently displays oscillatory or non-monotonic convergence.

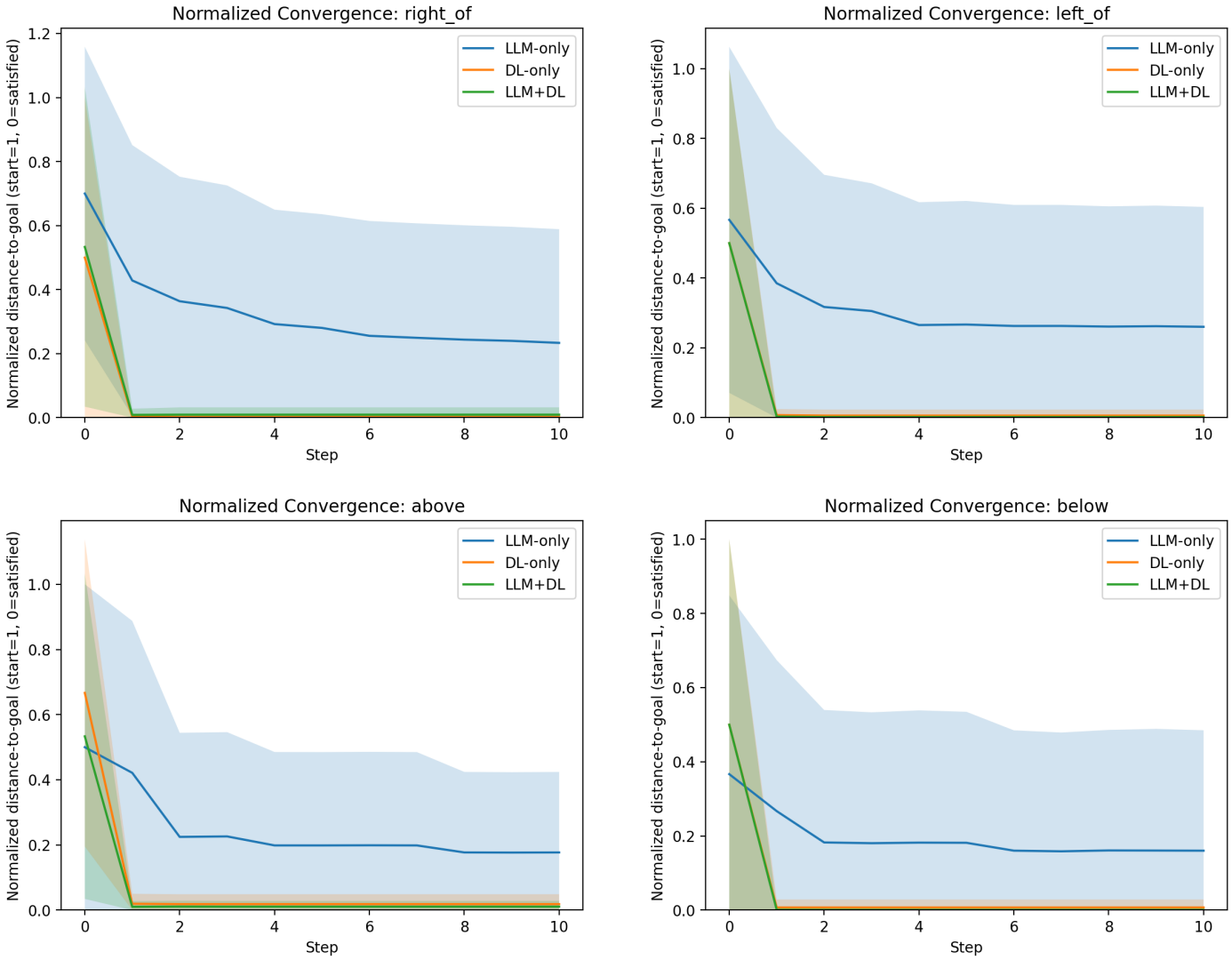

Figure A3. Normalized distance-to-goal over time for all spatial tasks. The proposed LLM+DL framework converges faster and more stably than the LLM-only control across all relations.

A4. Model-Specific Relative Improvements

Figure A4 shows the speedup factors and success-rate increases of LLM+DL over LLM-only control for each language model in order to measure the advantage of neuro-symbolic integration at the model level.

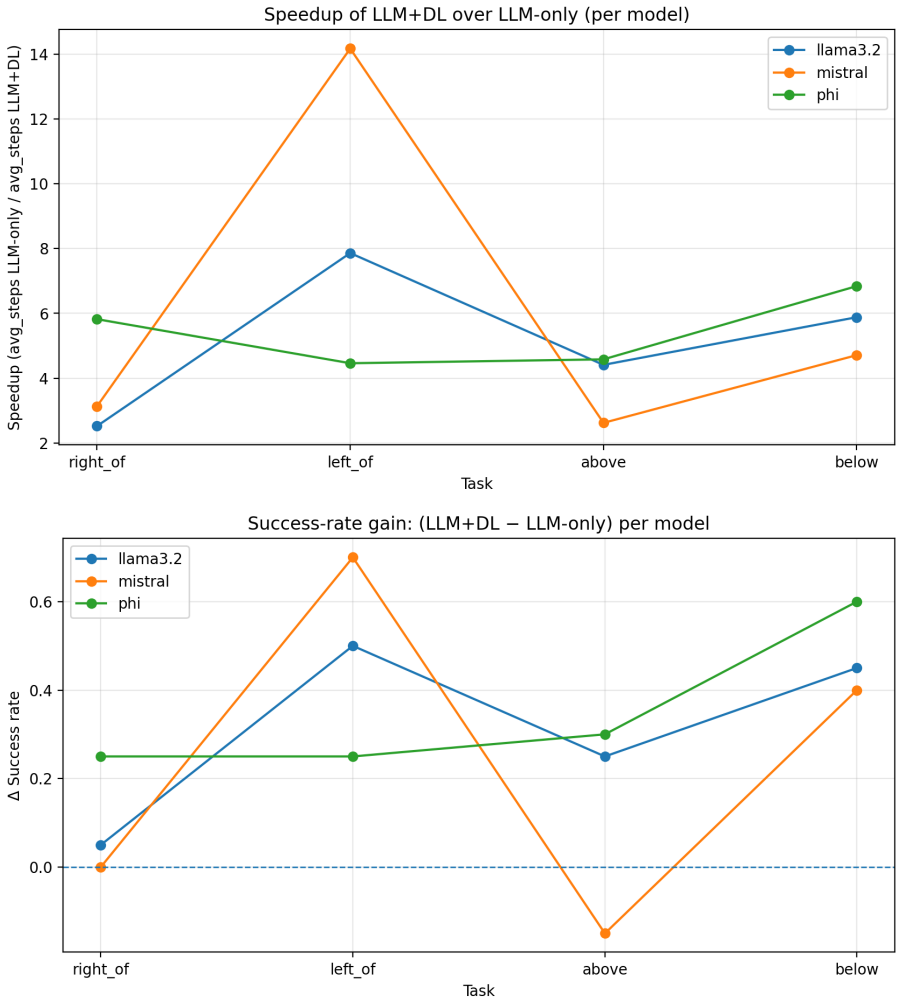

Figure A4. Model-specific speedup (left) and success-rate improvement (right) of LLM+DL over LLM-only control. Performance gains are consistent across all evaluated language models.

A5. Discussion of Supplementary Results

The additional findings support the paper's primary conclusions. In every in-depth analysis, the LLM+DL framework shows:

- Gains in performance that are consistent across different-scale language models
- Considerable decreases in convergence time and control steps
- Reduced variation and increased stability in closed-loop behavior

The efficacy of distinguishing symbolic reasoning from continuous execution in language-conditioned control tasks is further supported by these results.