Value Shaping for Deep Reinforcement Learning via Large Language Models

Submission ID: 7464

Abstract

Reinforcement Learning (RL) excels in long-term tasks but suffers from sample inefficiency in complex environments. While prior knowledge can address this, RL lacks mechanisms for effective integration. We propose Value Initialization and Adaptive Shaping (VIAS), a framework that uses feedback from large language models as external guidance to enhance RL learning efficiency. VIAS leverages LLMs as critics to shape Q-values, enabling agents to warm-start with informed estimates and adapt policies through heuristic-driven bootstrapping. Evaluated in the VirtualHome environment, VIAS demonstrates significant gains in sample efficiency and performance over conventional RL methods, highlighting its potential for complex applications. Webpage: https://tdqn-vias.github.io/

1 Introduction

Reinforcement Learning (RL) has shown impressive capabilities in enabling agents to learn long-term goal-oriented tasks through interaction with complex environments in multiple domains, such as games and robotics [Vinyals *et al.*, 2019; Hu *et al.*, 2024]. However, a notable limitation of RL lies in its reliance on sample-intensive exploration, which can make learning inefficient in large, complex settings. One can mitigate the computational issue of RL by warm-starting or regularizing learning with prior knowledge [Nair *et al.*, 2020; Cheng *et al.*, 2020].

In recent years, Large Language Models (LLMs) have achieved remarkable progress across various natural language processing tasks, ranging from text summarization to code generation [Zhang *et al.*, 2024; Vaithilingam *et al.*, 2022]. Despite their successes, LLMs face challenges in decision-making tasks, especially in dynamic environments where interactions are required [Wei *et al.*, 2022; Yao *et al.*, 2024]. This limitation arises from the inherent misalignment between the static, knowledge-encoded nature of LLMs and the interactive demands of RL with environments.

To address these limitations and improve the sample efficiency of RL, we propose a novel framework, Value Initialization and Adaptive Shaping (VIAS), that leverages LLM-derived insights as a form of external feedback to enhance the value-based RL process. Specifically, VIAS uses LLM feedback for an RL agent to "warm-start" policy learning with a foundational understanding of its environment and guide bootstrapping during training, providing the agent with heuristic-guided value updates that support effective policy adaptation. In VIAS, LLMs function as external critics that assess potential actions by offering insight into the long-term utility of states, thereby shaping the Q-values that guide the agent's decisions. This feedback enables the agent to prioritize meaningful actions and avoid irrelevant paths early in training, thus accelerating policy learning without compromising adaptability. In designing VIAS, we also explore various prompt structures to optimize LLM responses, focusing on alignment with the agent's task objectives. Careful prompt engineering ensures that LLM feedback is not only relevant but also actionable, providing context-sensitive evaluations that effectively shape the agent's learning trajectory.

We evaluated the effectiveness of VIAS in a simulated household environment, where both state and action spaces are described in natural language, implying commonsense knowledge for everyday activities. This setup allows us to test VIAS' ability to leverage LLM feedback in a realistic, language-based setting and assess its impact on sample efficiency. Our experiments demonstrate that VIAS outperforms conventional RL approaches and other LLM-based methods, significantly improving sample efficiency and enabling the agent to generalize effectively to new tasks. We further conducted a comprehensive analysis of the performance of our approach using three different LLMs to assess the sensitivity of VIAS to the choice of LLM and real-world robot demonstration to validate our approach in a real-world scenario. These results highlight VIAS' potential for enhancing RL performance in complex, real-world applications.

2 Related Work

Introducing prior knowledge to facilitate efficient policy learning has been a significant area of the RL research, aiming to reduce sample complexity and improve learning efficiency. Various approaches have been explored to incorporate prior knowledge, such as imitation learning, where agents learn from demonstrations provided by experts [Hussein *et al.*, 2017], and reward shaping, which adjusts the reward signal based on prior information to guide agent be-

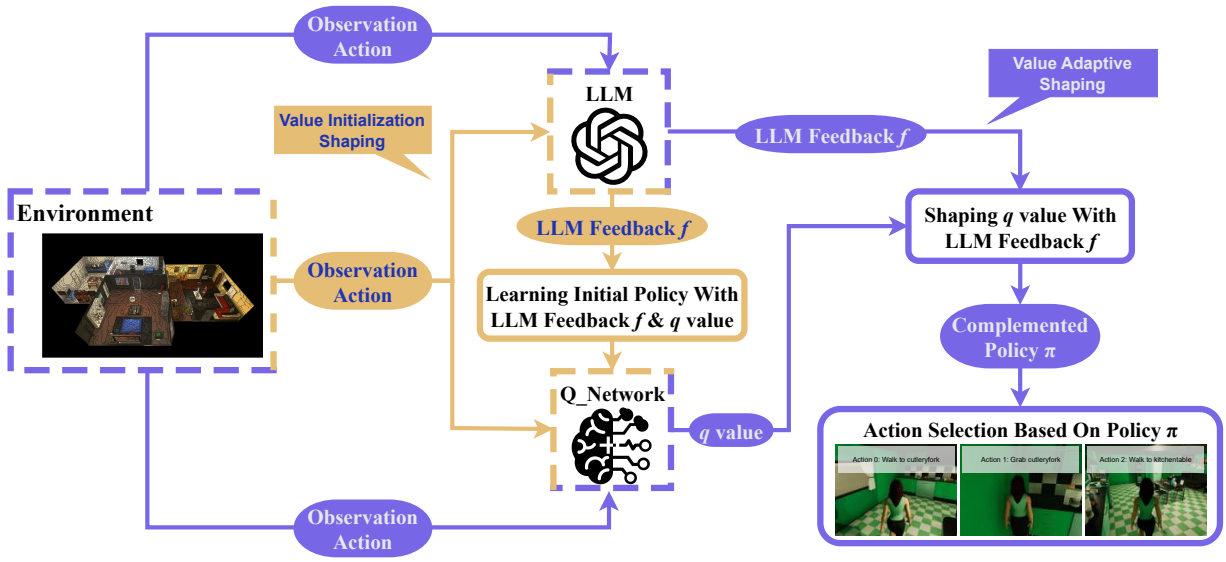


Figure 1: Overview of VIAS: The feedback from an LLM is used to initialize and update the Q-Network before and during training.

havior [Wiewiora *et al.*, 2003]. Techniques such as behavior cloning [Torabi *et al.*, 2018] have been employed to extract valuable insights from expert data. However, the dependency on high-quality expert demonstrations can be a limitation when such data is scarce or expensive to obtain. Methods like potential-based reward shaping ensure that the optimal policy remains unchanged while accelerating the learning process [Ng *et al.*, 1999]. Various approaches have been proposed to provide shaping rewards, such as human feedback and declarative knowledge [Knox and Stone, 2009; Hayamizu *et al.*, 2021]. Nonetheless, providing human feedback is tedious, and designing effective shaping functions requires domain expertise and can introduce bias if not carefully crafted.

There has been growing interest in LLMs to support RL for enhancing decision-making processes, as surveyed in a recent article [Cao *et al.*, 2024]. For example, LLMs have been used as world models to provide agents with a comprehensive understanding of environmental dynamics [Lin *et al.*, 2023; Chen *et al.*, 2022; Micheli *et al.*, 2022]. LLMs have also been utilized as policies to map observations to actions in decision-making tasks directly. For instance, SayCan [Ahn *et al.*, 2022] combines an LLM with a robotic affordance model to generate feasible action plans for robots in real-world environments. Similarly, frameworks like ReAct [Yao *et al.*, 2022] and VOYAGER [Wang *et al.*, 2023] integrate reasoning and acting by prompting LLMs to produce both reasoning traces and actions, enabling agents to perform tasks that require planning and interaction with environments. In addition, LLMs were fine-tuned to adapt agents to specific environments, which requires substantial computational resources [Mezghani *et al.*, 2023; Carta *et al.*, 2023]. Compared with those methods that treated RL as a black box and directly biased the selection of actions, VIAS assists RL by shaping its value functions.

Recent research has explored leveraging LLMs to automate the design of reward functions in reinforcement learning, reducing the reliance on manual reward crafting and domain expertise [Yu *et al.*, 2023; Ma *et al.*, 2023]. Some approaches focused on generating reward functions from high-level task descriptions, e.g., using language models to translate natural language instructions into reward functions [Kwon *et al.*, 2023; Du *et al.*, 2023]. Additionally, LLMs have been utilized to generate shaping rewards, providing supplementary information that facilitates learning [Chu *et al.*, 2023]. Those approaches allow an agent to steer its learning process by generating auxiliary objectives and intrinsic rewards through LLMs without expert intervention. EAGER incorporates question generation and answer metrics to assess the trajectory quality relative to a goal [Carta *et al.*, 2022]. ELLM incentivizes goal achievement based on suggestions from LLMs, directing agents toward meaningful behaviors [Du *et al.*, 2023]. These methods automated reward designs via LLMs and then let agents learn using standard RL methods, whereas VIAS enables RL agents to leapfrog before the trial and error process.

3 Preliminaries

This section outlines the foundational concepts and notations essential for understanding the proposed framework, Value Initialization and Adaptive Shaping (VIAS). We begin by formalizing the Reinforcement Learning (RL) problem, where states and actions are represented as text. Then, we explain how value-based RL algorithms solve such problems.

3.1 Reinforcement Learning for Natural Language

We formulate Reinforcement Learning (RL) problems as Partially Observable Markov Decision Processes (POMDPs), a tuple $(\mathcal{S}, \mathcal{A}, T, \mathcal{O}, \Omega, R, \gamma)$. $\mathcal{S}$ represents the state space. $\mathcal{A}$ is the action space. T is the state transition function. $\mathcal{O}$ is

the observation space. Ω is the observation function. R is the reward function and γ is the discount factor. In the text-based environment, the agent receives a textual description $o \in \mathcal{O}$ that encapsulates the relevant properties of the underlying state $s \in \mathcal{S}$ and takes a textual action command $a \in \mathcal{A}$ at the timestep t. A state $s \in \mathcal{S}$ defines the system dynamics and all properties of the environment. The objective is to learn a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that maximizes the expected cumulative discounted reward, defined as:

$$G(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right], \quad (1)$$

where s_t and a_t denote the state and action at timestep t, respectively. Value-based RL approaches learn to approximate a Q-value function, which indicates the expected cumulative rewards. Since the agent is not capable of observing states, the agent is required to estimate Q-values, $Q(h, a; \theta)$, using an observation history $h = (o_1, a_1, o_2, a_2, \cdots, o_t)$. Given the current Q-value estimate, the agent policy selects an action based on the history h_t at timestep t:

$$\pi(a_t|h_t; \theta) = \underset{a}{\operatorname{argmax}}\, Q(h_t, a; \theta). \quad (2)$$

3.2 Template-Based Deep Q-Networks

Template-based Deep-Q Network (TDQN) [Hausknecht *et al.*, 2020] is an extension of LSTM-DQN [Narasimhan *et al.*, 2015], which represents textual states and actions using embedding vectors to learn accurate Q-value approximations. Textual observations are tokenized and encoded using neural network architectures designed for extracting meaningful features from natural language. TDQN employs Gated Recurrent Units (GRUs) as its primary feature extractor. These GRU-based encoders transform textual inputs into latent vectors, preserving semantic and contextual information crucial for Q-value prediction. For instance, in a text-based environment describing a current situation as *"Goal: put a plate on a table. Obs: You are in a kitchen, ..., Previous Action: walk to kitchen,"* the encoder processes the text to capture the relationships between the goal, current observations, and the previous action. These structured representations are then passed to a multi-layer perceptron (MLP), which predicts the Q-values for each possible action, enabling the selection of the most suitable action for the given state. To address the challenges of combinatorial action spaces, which complicate exploration and learning, TDQN adopts a template-based action representation. The action space is decomposed into tuples of $\langle verb, arg_0, arg_1 \rangle$. $verb$ represents a verb phrase template with placeholders for objects, e.g., $\langle put\ OBJ\ on\ OBJ \rangle$. The placeholders *OBJ* are dynamically replaced with specific objects identified in the environment, which are specified with arg_0 and arg_1. For instance, the action tuple $\langle put\ OBJ\ on\ OBJ, plate, table \rangle$ is rendered into the fully instantiated action *"put a plate on a table"*, which the agent can execute. TDQN updates its parameters θ using the Deep Q-Network (DQN) algorithm.

4 LLM Heuristic Guided RL

We present an approach that integrates an LLM to enhance the efficiency of reinforcement learning. Our method leverages the LLM's capabilities to provide heuristic feedback in the form of Q-values—expected cumulative rewards—which directly influence the agent's learning process. Unlike conventional approaches that rely on reward signals or handcrafted features as heuristic guidance, our approach differentiates itself by using the LLM's feedback to shape the Q-values. This integration serves two primary functions: initializing the RL agent with well-informed Q-values, effectively warming up the agent to explore the environment, and continuously refining the Q-values during training by incorporating heuristic guidance from the LLM, which can be especially effective in the context-rich environment.

4.1 Value Initialization Shaping

Value Initialization Shaping (VIS) initializes Q-values by leveraging observation-action pairs (o, a) obtained before training. This approach reduces the likelihood of the agent's unreasonable actions during its initial exploration, effectively jumpstarting the learning process. By minimizing uncertainty and random exploration, the agent can quickly adapt its policy based on the guidance provided by the LLM.

To initialize the Q-values, an initial buffer $\mathcal{D}_{init}$ is constructed by collecting observation-action pairs (o, a) from the environment. The process may involve an arbitrary agent or human player interacting with the environment. Unlike previous works that use LLMs to estimate immediate rewards, VIS employs a prompt strategy to request the LLM's estimation of the expected cumulative reward for taking an action a, in the observation o. The LLM outputs a heuristic value f representing the expected cumulative reward for the given observation-action pair. This approach ensures that the LLM's feedback aligns with the agent's long-term goals rather than focusing solely on short-term outcomes.

The heuristic values f are used to initialize the Q-values, shaping the initial policy based on the LLM's insights. The pre-training process learns the parameter θ using the heuristic values f generated by minimizing the loss between the LLM-provided heuristic values and the initial Q-values:

$$\mathcal{L}_{\textit{pre-train}}(\theta) = \mathbb{E}_{(o,a)\sim\mathcal{D}_{init}}\left[\left(f - Q(o, a; \theta)\right)^2\right], \quad (3)$$

where the replay buffer $\mathcal{D}_{init}$ contains recent transitions. This initialization strategy reduces the exploration burden in complex state-action spaces, allowing the agent to avoid unproductive or suboptimal trajectories during early training. Consequently, the agent starts with a policy that is not random but informed by the LLM's heuristic knowledge.

4.2 Value Adaptive Shaping

Beyond the initialization phase, the LLM provides heuristic feedback during the RL training. This feedback serves as an auxiliary signal, complementing the standard reward signal provided by the environment. We refer to this shaping mechanism as Value Adaptive Shaping (VAS). At each decision point during training, we query the LLM to obtain heuristic Q-values f for the current observation-action pairs (o, a). These heuristic values provided during training guide the agent towards more promising actions based on prior knowledge encoded in the LLM, even if the agent encounters unseen observations.

To integrate this feedback, we augment the standard Q-learning update rule with a heuristic adjustment term:

$$Q(o,a) \leftarrow Q(o,a) + \alpha TD + \beta[f - Q(o,a)], \quad (4)$$

where the Temporal Difference (TD) error is defined as:

$$TD = r + \gamma \max_{a'} Q(o',a') - Q(o,a). \quad (5)$$

Here, α is the learning rate, and β is the heuristic factor controlling the influence of the LLM's feedback f. We integrate the heuristic feedback within a replay buffer $\mathcal{D}$ to efficiently utilize both experiential and heuristic information. Each experience tuple stored in the buffer includes the heuristic value f from the LLM:

$$\mathcal{D} \leftarrow \mathcal{D} \cup \{(o,a,r,o',f)\}.$$

During training, we sample mini-batches of experiences from $\mathcal{D}$ and update the Q-values using Equation (4). This approach allows the agent to learn from past experiences while adjusting Q-values toward the heuristic estimates.

4.3 TDQN+VIAS

We combine the two mechanisms above, Value Initialization Shaping (VIS) and Value Adaptive Shaping (VAS), into a unified framework called Value Initialization and Adaptive Shaping (VIAS). The VIAS framework can be used with arbitrary value-based RL algorithms designed for the text interface. In this paper, we use TDQN as a backbone and call our method TDQN+VIAS. TDQN+VIAS initializes the agent's Q-values using heuristic values provided by the LLM for a set of observation-action pairs collected before training. This initial shaping guides the agent's early exploration and reduces the likelihood of unproductive actions. During the training phase, TDQN+VIAS continues incorporating heuristic feedback from the LLM to adaptively adjust the Q-values, ensuring that the agent remains aligned with long-term objectives and can benefit from the LLM's guidance even in previously unseen states.

Algorithm 1 outlines the TDQN+VIAS approach. In the **Initialization Phase** (Lines 2–7), we begin by setting up an empty initial replay buffer $\mathcal{D}$init. This buffer is used to store observation-action pairs along with their corresponding heuristic values obtained from the LLM before training. For each observation-action pair (o,a) collected before the training starts, the algorithm queries the LLM to obtain a heuristic value f. This heuristic value represents the LLM's estimated cumulative reward for taking action a in observation o. The tuple (o,a,f) is then stored in the initial replay buffer $\mathcal{D}$init. Subsequently, the algorithm initializes the Q-value function $Q(o,a;\theta)$ by minimizing the pre-training loss function $\mathcal{L}_{\textit{pre-train}}(\theta)$ (Line 7).

In the **Training Phase** (Lines 8–21), we train the agent over T episodes. As in the initialization phase, TDQN+VIAS initializes a replay buffer to store trajectories during training. For each episode, the environment is initialized by setting the initial observation o. The training loop continues until goal conditions are satisfied. At each time step within an episode, the agent selects an action a based on the current Q-values $Q(o,a)$. This selection is typically performed using a policy derived from Q, such as the soft-max policy. The selected action a is then executed in the environment, resulting in the observation of an immediate reward r and the next observation o'. Querying the LLM to obtain a heuristic value f for the current observation-action pair, the agent stores the experience in the replay buffer $\mathcal{D}$ for later updates (Line 16). A mini-batch of experiences is then sampled from the replay buffer $\mathcal{D}$ for training, using prioritized experience replay. The Q-value $Q(o,a)$ is then updated by incorporating both the TD error and the heuristic adjustment from the LLM (Line 18).

This integrated approach allows the agent to benefit from both the initial shaping provided by the LLM during the pre-training phase and the adaptive shaping during the training phase. We expect that, by leveraging the LLM's heuristic guidance before and throughout the learning process, the agent can accelerate its convergence, improve its policy performance, and enhance the exploration efficiency.

Algorithm 1 TDQN+VIAS

Input: Observation space $\mathcal{O}$, action space $\mathcal{A}$, discount factor γ, learning rates α, heuristic factor β, total episodes T

1: **Initialization Phase:**
2: Initialize empty replay buffer $\mathcal{D}_{\text{init}}$
3: **for** each observation-action pair (o,a) collected beforehand **do**
4: Query LLM to obtain heuristic value f
5: Store (o,a,f) in $\mathcal{D}_{\text{init}}$
6: **end for**
7: Initialize Q-values $Q(o,a;\theta)$ by using Equation (3)
8: **Training Phase:**
9: Initialize empty replay buffer $\mathcal{D}$
10: **for** episode $= 1$ to T **do**
11: Initialize observation o
12: **while** not terminal **do**
13: Select action a using policy derived from Q
14: Execute action a, observe reward r and next observation o'
15: Query LLM to obtain heuristic value f
16: Store (o,a,r,o',f) in $\mathcal{D}$
17: Sample mini-batch from $\mathcal{D}$
18: Update Q-value using Equations (4) and (5)
19: Update observation $o \leftarrow o'$
20: **end while**
21: **end for**

5 Experiments

In this section, we evaluate the effectiveness of the TDQN+VIAS approach within the VirtualHome environment. We focus on understanding how leveraging large language models (LLMs) for value estimation can enhance sample efficiency and overall performance. We begin by introducing the VirtualHome environment and the tasks under consideration, then present the training setup, including hyperparameters, and conclude with details on our evaluation.

5.1 Experiment Setup

VirtualHome is a complex 3D simulated household environment that supports a broad range of human-like activities. Each environment instance consists of interconnected rooms (e.g., kitchen, living room, and bedroom) populated with various interactive objects (e.g., tables, chairs, plates, refrigerators, and televisions). Agents can engage in realistic sequences of actions to perform tasks that mimic everyday household activities.

State Representation. States are represented as directed graphs, where nodes denote objects and edges indicate spatial or semantic relationships, such as adjacency, containment, or functional connections. Originally, VirtualHome employed a program-based representation for actions and states. To facilitate natural language-based RL, we have adapted the environment to provide all observations and actions in the natural language form. At each timestep, the agent receives a textual description of its surroundings and the objects it can interact with. This representation challenges the agent to interpret language, reason over spatial and semantic relationships, and select viable actions to achieve a high-level goal.

Action Space. The agent's action space comprises natural language instructions derived from templates. For instance, actions may include simple commands like *"walk to the kitchen"* or *"grab a plate"*, as well as more complex tasks such as *"put the apple on the table"*. The complexity of these actions requires the agent to identify the correct verbs, arguments, and object references from its textual observations.

Tasks. We evaluate TDQN+VIAS on three tasks of increasing complexity, each testing different aspects of the agent's ability to parse language, understand object affordances and sequence actions:

1. **Setup Table**: The agent must identify a target object and place it onto another specified object. This task involves locating items and spatial reasoning about object placement.
2. **Prepare Food**: Building on the first task, the agent may also need to retrieve food items from the refrigerator or gather utensils. This task challenges the agent's ability to handle multiple steps, navigate to different rooms, and identify food-related objects.
3. **Turn on TV**: The agent must find the television and locate the correct switch or remote to turn it on. This requires distinguishing the TV from other appliances and objects, and accurately executing the necessary action to power it on.

Complexity. To illustrate the complexity of these tasks, Table 1 presents a sample set of actions and the objects commonly found in each scenario. As the tasks increase in difficulty, the agent encounters a wider variety of objects, potential actions, and longer instructions, requiring more sophisticated reasoning and planning capabilities.

These tasks test the agent's ability to interpret natural language instructions, comprehend object functions, navigate a realistic virtual setting, and execute multi-step action sequences. By demonstrating competence across this spectrum, TDQN+VIAS showcases its potential for improving learning efficiency and overall task performance in text-based reinforcement learning environments.

Table 1: Example actions and objects in each task.

Setup Table	"grab a plate", "put the plate on the kitchentable"	plate, wineglass, cutleryfork, kitchentable, kitchencounter, etc
Prepare Food	"grab an apple", "open a fridge"	apple, cupcake, cereal, coffeepot, fridge, cutleryfork, etc
Turn on TV	"walk to livingroom", "turn on the TV"	remote, bookshelf, coffeepot, etc

5.2 Training Setup and Hyperparameters

Our VIAS approach is generally applicable to any value-based RL algorithm and is composed of VIS and VAS. Our baseline methods include the original Template-DQN (TDQN) approach, as well as a version of TDQN that is enhanced by reward shaping [Chu *et al.*, 2023]. Five approaches are evaluated and compared:

- **TDQN**: Naive language-based RL baseline.
- **TDQN + RS**: TDQN augmented with a separate reward shaping method employing LLM [Chu *et al.*, 2023].
- **TDQN + VIS**: TDQN with value initialization shaping using LLM feedback.
- **TDQN + VAS**: TDQN with value adaptive shaping using LLM feedback.
- **TDQN + VIAS**: Our proposed approach.

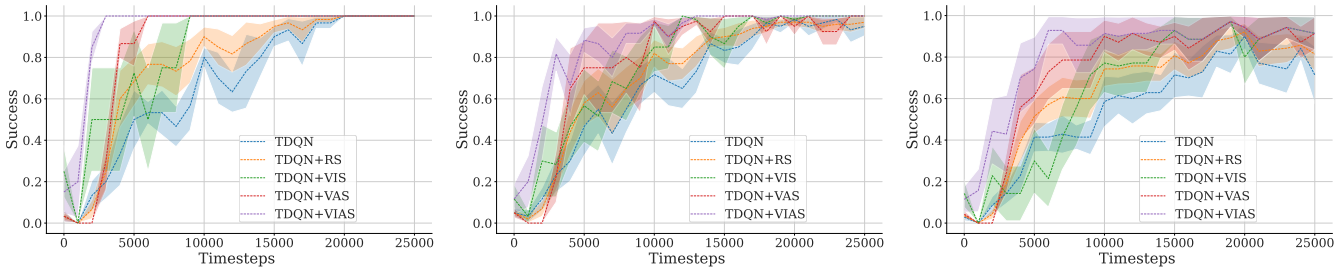
All methods learn Q-values from textual observations and actions (see Section 3.1).

For the LLM feedback, we use GPT-4o. We employ a few-shot prompting strategy in which the Q-value concept is first introduced, followed by several illustrative examples of states, actions, and associated Q-values. After establishing these examples, the current query is then presented to GPT-4o, allowing it to provide context-sensitive Q-value estimates. This approach ensures that the LLM guidance is both interpretable and grounded in prior examples.

We do not query the LLM at every timestep. Instead, we determine a query frequency based on the minimal number of steps required for an agent with an optimal policy to complete an episode. For instance, if the optimal policy resolves a scenario in eight steps, we may query the LLM every three to five steps to ensure computational efficiency while retaining the performance benefits of external feedback.

5.3 Evaluation Protocol

We periodically evaluate the agent's performance every 1,000 timesteps by running 10 evaluation episodes. During these evaluations, the exploration rate is set to zero, ensuring the agent relies solely on its learned Q-values without any exploratory actions. We the average success rates as the performance metric. By examining the metrics over the course of training, we gain insights into how TDQN + VIAS influences the agent's sample efficiency and overall task proficiency.



(a) Setup Table (b) Prepare Food (c) Turn on TV

Figure 2: The average success rate over 10 runs. The X-axis indicates the number of steps the agents interact with the environment. For each task, our approach is superior to the baselines.

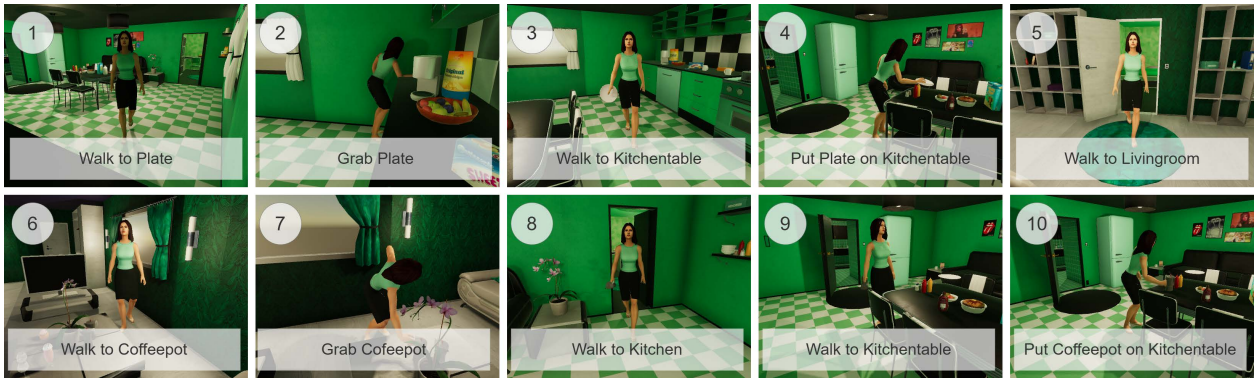


Figure 3: An example of action sequences from the learned policy in the Setup Table task.

5.4 Simulation Results

We evaluated our VIAS approach against two baseline methods: a naive template-based DQN (TDQN) and a reward-shaping approach using LLMs [Chu *et al.*, 2023]. We conducted experiments on three tasks with increased complexity: Setup Table, Prepare Food, and Turn on TV. Each task requires the agent to leverage textual observations and natural language instructions to interact with the environment and achieve the goal state efficiently.

Overall Performance Trends: Across all three tasks, TDQN + VIAS consistently demonstrated more rapid and stable convergence compared to the naive TDQN and the reward-shaping baseline Figure 2. On simpler tasks, such as Setup Table and Prepare Food, TDQN + VIAS agents began performing effectively from the early training stages, achieving higher success rates and shorter episode lengths much sooner than the other methods. This advantage is particularly noticeable in the initial phases of training, where VIAS's value-shaping capabilities provide a strong "warm start" to guide action selection. Figure 3 shows the sequence of actions and the corresponding outcomes.

Robustness in Complex Tasks: In the most challenging scenario, Turn on TV, the naive TDQN method struggled to identify the optimal strategy even after extensive training. By contrast, TDQN + VIAS made steady progress, significantly reducing the time required to find near-optimal policies. While reward shaping with LLMs improved upon the naive baseline, its performance gains were modest. In contrast, TDQN + VIAS delivered both the immediate benefits of better initial value estimates and the ongoing support during training to refine these values further.

Convergence and Sample Efficiency: The combination of Q-value initialization and value updates guided by LLM feedback resulted in the fastest convergence rates observed. VIAS's ability to integrate these value estimates early and refine them over time led to markedly shorter training durations and more sample-efficient learning. The improved sample efficiency is evident in terms of fewer required timesteps for high-reward policies, offering practical advantages in scenarios where simulation costs are substantial.

Our experiments show that TDQN + VIAS not only accelerates the initial learning phase through better-informed Q-values but also sustains this advantage throughout training. By leveraging LLM-based value shaping, the agent consistently outperforms baseline methods on tasks of varying complexity. The improvements in convergence speed, final success rate, and episode length underscore the potential of incorporating LLM feedback into value-based RL frameworks, ultimately promoting more efficient and effective policy learning.

5.5 Other LLMs' Performance

A fundamental component of the VIAS framework is the use of an off-the-shelf LLM to provide heuristic guidance for policy learning. To assess the sensitivity of VIAS to the choice of LLM, we conducted a comprehensive analysis of its per-

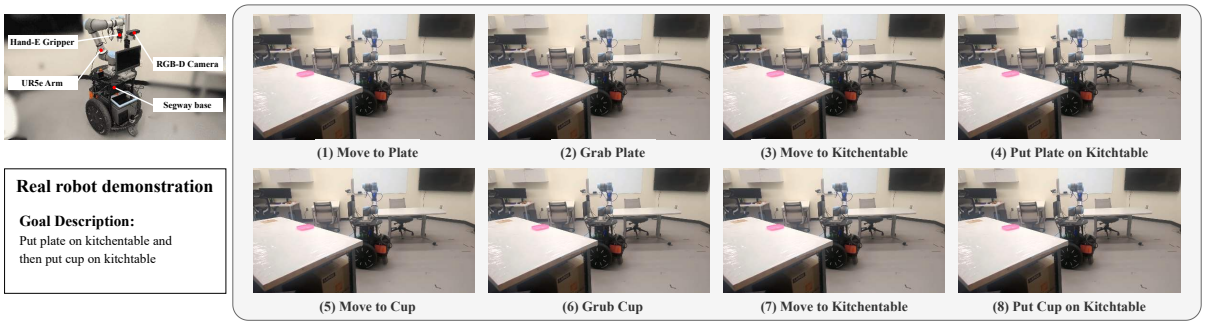


Figure 4: Action sequence of a real-robot demonstration showcasing a learned policy transferred from a simulator environment. The robot is tasked with setting up a table by placing a plate and a cup onto a kitchentable.

formance using three distinct LLMs. Specifically, we evaluated three versions of the VIAS system by integrating the following LLMs: GPT-4o, Gemini-2.0, and Claude-3.5. These models were selected for their diverse architectures and capabilities, enabling us to assess their respective influences.

The results of this analysis are summarized in Table 2, which presents a detailed comparison of the three VIAS implementations. Among the models, Gemini-2.0 consistently demonstrated superior performance across all tasks. Despite the advantage of Gemini-2.0, it is worth noting that none of the three models caused a significant degradation in the overall performance of VIAS. While the choice of LLM can influence the degree of performance enhancement, VIAS remains effective regardless of the specific model employed.

Table 2: Success rates of three implementations of VIAS using different off-the-shelf LLMs

	GPT-4o	**Gemini-2.0**	**Claude-3.5**
Setup Table	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
Prepare Food	0.99 ± 0.01	$\mathbf{1.00 \pm 0.00}$	0.95 ± 0.06
Turn on TV	0.91 ± 0.09	$\mathbf{0.96 \pm 0.06}$	0.94 ± 0.07

5.6 Real Robot Deployment

To validate the proposed approach in a real-world scenario, we conducted a demonstration using a mobile manipulator tasked with setting up a table according to a specified goal. The mobile manipulator used in this demonstration consists of a Segway base for navigation, a UR5e robotic arm equipped with a Hand-E gripper mounted on the Segway-base for manipulation, and an overhead RGB-D camera fixed relative to the robot for perception. This setup provided the necessary capabilities for the robot to perceive its environment, navigate within it, and interact with objects effectively.

For this experiment, the robot was assumed to possess a predefined set of fundamental skills for completing the table setup task. These skills included `grab`, `put`, and `move to`. The `grab` and `put` actions were implemented using the GG-CNN framework [Morrison *et al.*, 2018], which provides robust grasp detection capabilities. The `move to` action was implemented using a standard navigation stack, enabling the robot to plan and execute navigation.

The policy deployed during the demonstration was trained in a simulated environment to ensure robust learning before real-world deployment. During the deployment phase, we employed a template-based approach for observation descriptions, which converted visual information from the RGB-D camera into textual descriptions. This allowed the robot to process high-level semantic information about its environment. Additionally, to ensure task semantics remained consistent, we manually replaced instances of the action `walk` with `move to` in the observation descriptions.

The robot was tasked with completing the goal: `Goal: put the plate on the kitchentable and then put the cup on the kitchentable.` Using its predefined skills, the robot successfully executed the task by employing a sequence of `move to`, `grab`, and `put` actions. The robot demonstrated its ability to navigate effectively, identify the relevant objects, and perform the required manipulations. The sequence of actions and the corresponding outcomes are illustrated in Figure 4.

This demonstration highlights the capability of the proposed approach to transfer a trained policy from a simulated environment to a real-world setting while maintaining task accuracy and semantic consistency.

6 Conclusion

In this paper, we proposed the VIAS framework that integrates LLM-provided heuristic feedback into both the initialization and training phases of RL. By combining the strengths of applying heuristic feedback at the initialization and training phases, the agent benefits from a strong starting point and continuous guidance, leading to faster learning and improved policy performance. This unified approach demonstrates the potential of leveraging advanced language models to enhance decision-making processes in RL agents. Potential directions for future research include developing methods to dynamically adjust the heuristic factor β based on the agent's confidence or the quality of the LLM's feedback, integrating VIAS with reward shaping methods, and leveraging pre-trained vision-language models for visual perception and end-to-end behavior generations.

References

[Ahn *et al.*, 2022] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[Cao *et al.*, 2024] Yuji Cao, Huan Zhao, Yuheng Cheng, Ting Shu, Yue Chen, Guolong Liu, Gaoqi Liang, Junhua Zhao, Jinyue Yan, and Yun Li. Survey on large language model-enhanced reinforcement learning: Concept, taxonomy, and methods. *arXiv preprint arXiv:2404.00282*, 2024.

[Carta *et al.*, 2022] Thomas Carta, Pierre-Yves Oudeyer, Olivier Sigaud, and Sylvain Lamprier. Eager: Asking and answering questions for automatic reward shaping in language-guided rl. *Advances in Neural Information Processing Systems*, 35:12478–12490, 2022.

[Carta *et al.*, 2023] Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning. In *International Conference on Machine Learning*, pages 3676–3713. PMLR, 2023.

[Chen *et al.*, 2022] Chang Chen, Yi-Fu Wu, Jaesik Yoon, and Sungjin Ahn. li2022pretrainedtransdreamer: Reinforcement learning with transformer world models. *arXiv preprint arXiv:2202.09481*, 2022.

[Cheng *et al.*, 2020] Ching-An Cheng, Andrey Kolobov, and Alekh Agarwal. Policy improvement via imitation of multiple oracles. *Advances in Neural Information Processing Systems*, 33:5587–5598, 2020.

[Chu *et al.*, 2023] Kun Chu, Xufeng Zhao, Cornelius Weber, Mengdi Li, and Stefan Wermter. Accelerating reinforcement learning of robotic manipulations via feedback from large language models. *arXiv preprint arXiv:2311.02379*, 2023.

[Du *et al.*, 2023] Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models. In *International Conference on Machine Learning*, pages 8657–8677. PMLR, 2023.

[Hausknecht *et al.*, 2020] Matthew Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. Interactive fiction games: A colossal adventure. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7903–7910, 2020.

[Hayamizu *et al.*, 2021] Yohei Hayamizu, Saeid Amiri, Kishan Chandan, Keiki Takadama, and Shiqi Zhang. Guiding robot exploration in reinforcement learning via automated planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 31, pages 625–633, 2021.

[Hu *et al.*, 2024] Jiaheng Hu, Rose Hendrix, Ali Farhadi, Aniruddha Kembhavi, Roberto Martin-Martin, Peter Stone, Kuo-Hao Zeng, and Kiana Ehsan. Flare: Achieving masterful and adaptive robot policies with large-scale reinforcement learning fine-tuning. *arXiv preprint arXiv:2409.16578*, 2024.

[Hussein *et al.*, 2017] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Comput. Surv.*, 50(2), April 2017.

[Knox and Stone, 2009] W Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: The tamer framework. In *Proceedings of the fifth international conference on Knowledge capture*, pages 9–16, 2009.

[Kwon *et al.*, 2023] Minae Kwon, Sang Michael Xie, Kalesha Bullard, and Dorsa Sadigh. Reward design with language models. *arXiv preprint arXiv:2303.00001*, 2023.

[Lin *et al.*, 2023] Jessy Lin, Yuqing Du, Olivia Watkins, Danijar Hafner, Pieter Abbeel, Dan Klein, and Anca Dragan. Learning to model the world with language. *arXiv preprint arXiv:2308.01399*, 2023.

[Ma *et al.*, 2023] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.

[Mezghani *et al.*, 2023] Lina Mezghani, Piotr Bojanowski, Karteek Alahari, and Sainbayar Sukhbaatar. Think before you act: Unified policy for interleaving language reasoning with actions. *arXiv preprint arXiv:2304.11063*, 2023.

[Micheli *et al.*, 2022] Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. *arXiv preprint arXiv:2209.00588*, 2022.

[Morrison *et al.*, 2018] Douglas Morrison, Peter Corke, and Jürgen Leitner. Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. *arXiv preprint arXiv:1804.05172*, 2018.

[Nair *et al.*, 2020] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[Narasimhan *et al.*, 2015] Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. Language understanding for text-based games using deep reinforcement learning. *arXiv preprint arXiv:1506.08941*, 2015.

[Ng *et al.*, 1999] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

[Torabi *et al.*, 2018] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.

[Vaithilingam *et al.*, 2022] Priyan Vaithilingam, Tianyi Zhang, and Elena L Glassman. Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models. In *Chi conference on human factors in computing systems extended abstracts*, pages 1–7, 2022.

[Vinyals *et al.*, 2019] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

[Wang *et al.*, 2023] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

[Wei *et al.*, 2022] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[Wiewiora *et al.*, 2003] Eric Wiewiora, Garrison W Cottrell, and Charles Elkan. Principled methods for advising reinforcement learning agents. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 792–799, 2003.

[Yao *et al.*, 2022] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[Yao *et al.*, 2024] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[Yu *et al.*, 2023] Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, et al. Language to rewards for robotic skill synthesis. *arXiv preprint arXiv:2306.08647*, 2023.

[Zhang *et al.*, 2024] Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B Hashimoto. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57, 2024.