Climb with SHERPA: Heuristic-Guided Reinforcement Learning via Segmented Experience Relay

1st Minji Kim
Department of Computer Science and Engineering
Seoul National University
Seoul, Republic of Korea
mjkim@bi.snu.ac.kr

2nd Ganghun Lee
Interdisciplinary Program in Artificial Intelligence and AIIS
Seoul National University
Seoul, Republic of Korea
zxc8594@snu.ac.kr

3rd Minsu Lee*
School of Artificial Intelligence Convergence
Sungshin Women's University
Seoul, Republic of Korea
mslee@sungshin.ac.kr

4th Byoung-Tak Zhang*
Department of Computer Science and Engineering
Interdisciplinary Program in Artificial Intelligence and AIIS
Seoul National University
Seoul, Republic of Korea
btzhang@bi.snu.ac.kr

***Abstract*—In sparse-reward, long-horizon domains, reinforcement learning (RL) often suffers from slow convergence and instability, complicating robotic manipulation. Previous heuristic-guided approaches have relied on step-level actions and imitation loss, but struggle to maintain temporal coherence or solve multi-stage tasks. We present SHERPA (Segmented Heuristic Experience Relay for Policy Assistance), which alternates control between a heuristic policy and an RL policy in contiguous segments, preserving coherent sub-trajectories and yielding more stable learning. Unlike purely imitative objectives, SHERPA leverages expert-like heuristic guidance while optimizing its policy through RL, thereby enabling performance that ultimately surpasses the heuristic itself. This behavior can be further strengthened by incorporating phase-specific rewards naturally derived from heuristic rules. Across ten tasks in the Fetch and Panda suites, including four long-horizon benchmarks, SHERPA consistently outperforms RL, IL, and heuristic-guided baselines and demonstrates robustness even under degraded heuristics. Real-world experiments on a UR5 robot further confirm SHERPA's practical scalability.**

***Index Terms*—Reinforcement Learning, Transfer Learning, Deep Learning in Grasping and Manipulation**

I. Introduction

Reinforcement Learning (RL) has recently shown impressive capabilities across a wide range of real-world applications, including artwork generation [1], autonomous driving [2], and logistics management [3]. Despite these advances, RL still faces fundamental challenges, with high sample complexity remaining one of the most critical issues, particularly in sparse-reward, long-horizon tasks. These difficulties become especially severe in robotics, where agents must learn from sparse and delayed rewards under high-dimensional continuous control and costly real-world interactions, further slowing learning and undermining stability [4].

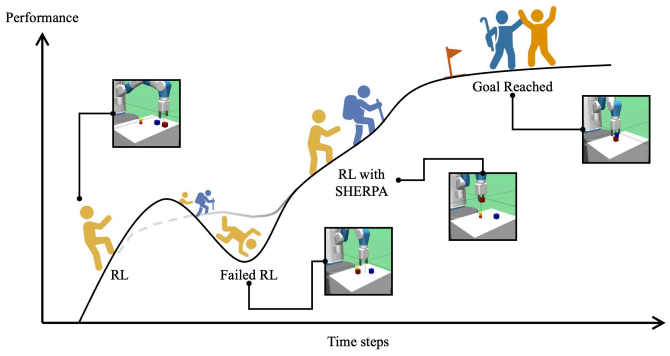

Fig. 1. Conceptual representation of SHERPA's role in the learning phase. SHERPA's heuristic guidance helps the learning policy in RL to avoid task failures and achieve successful performance.

To overcome these limitations, one common approach leverages prior knowledge or pre-trained models—often referred to as warm start optimization—to accelerate initial learning [5]. This prior knowledge can take various forms, including engineered heuristics, analytical controllers, or transferred policies [6]. Another related approach is imitation learning (IL), where agents mimic expert demonstrations; however, IL is limited by its reliance on expensive static datasets and poor generalization beyond the demonstration distribution, emphasizing the need for more interactive ways to encode expert knowledge into RL.

We propose SHERPA (Segmented Heuristic Experience Relay for Policy Assistance), a novel framework that leverages heuristic expert knowledge while continuing to optimize the

*Corresponding authors: Minsu Lee and Byoung-Tak Zhang

task policy, rather than merely mimicking the heuristic. Within each episode, SHERPA alternates control between the RL policy and a heuristic policy in contiguous segments, allowing the agent to explore autonomously while periodically receiving task-relevant guidance. Unlike imitation-based designs, heuristic experience is integrated directly into the RL update, enriching the replay buffer without overriding the agent's own optimization signals, so that the learned policy can ultimately surpass the heuristic guidance it initially relied upon.

We evaluate our approach on various robotic object manipulation tasks in the Fetch and Panda environment from OpenAI Gym. Comparisons against standalone RL, IL, the recent heuristic-integrated RL approach (RLingua) [7], and pure heuristic baselines show that SHERPA not only facilitates faster learning but also surpasses the performance of the underlying heuristics, while consistently outperforming IL.

The key contributions of this work are as follows:

- **Framework**: We propose SHERPA, a novel RL framework that alternates control between rule-based heuristic and learned policies in a relay-like, segmental manner to enhance learning efficiency and policy adaptability.
- **Performance**: In challenging long-horizon robotic tasks where prior heuristic-guided RL methods such as RLingua fail, SHERPA achieves consistently superior performance, outperforming standard RL, IL, and LLM-augmented baselines even with imperfect heuristics.
- **Scalability**: SHERPA is a framework for improving RL policies by leveraging suboptimal controllers, enabling scalable integration of heuristics ranging from handcrafted rules to LLM-generated guidance.

II. Related Work

A. Incorporating Prior Knowledge in RL

Recent studies have shown that RL can be significantly accelerated by incorporating prior knowledge such as heuristics, pretrained models, human feedback, or language instructions to guide exploration and improve sample efficiency [8]–[10]. These approaches introduce structural inductive biases that help mitigate cold-start issues and enable faster convergence. More recently, LLMs have been used for reward design, policy initialization, and action guidance, with particular success in generating dense or adaptive reward functions from natural language or multimodal inputs [11]–[13]. LLMs have also been explored as action-generating agents. Among heuristic-integrated RL methods, RLingua introduces an imitation loss that enables agents to learn from heuristic actions during training. In contrast, SHERPA incorporates heuristic experience directly into the RL update via a relay-based interaction mechanism, avoiding explicit imitation objectives while preserving the agent's optimization process. This design enables effective use of structured guidance without requiring curated demonstrations or additional supervision.

B. Imitation Learning

Imitation learning (IL) enables agents to acquire complex behaviors by using expert demonstrations rather than learning solely from environment rewards. Early methods such as Behavior Cloning (BC) [14] directly map observations to actions but suffer from compounding errors when encountering unseen states. DAGGER [15] addresses this by iteratively querying the expert to augment the training dataset. Adversarial approaches, including GAIL [16] and AIRL [17], introduce a discriminator to distinguish expert from agent behavior, with AIRL additionally recovering a reward function through inverse RL. Non-adversarial methods such as ValueDICE [18] and IQ-Learn [19] frame IL as a distribution-matching problem, avoiding reward modeling. To address the constraints of online interaction, offline and off-policy IL methods have been developed. Examples include BCO [20], IQL [21], DemoDICE [22], and learning from imperfect demonstrations [23].

Despite these advances, IL-based methods remain heavily dependent on high-quality expert data, which is costly or impractical in domains like robotics involving complex object manipulation. Above all, this dependency imposes a ceiling on achievable performance, leaving room for further optimization.

III. Method

This section introduces SHERPA (Segmented Heuristic Experience Relay for Policy Assistance), a framework where heuristic actions are injected into episodes in segmented form as partial demonstrations from arbitrary states and absorbed into the policy through standard RL optimization—enabling implicit knowledge transfer without explicit IL. As illustrated in Figure 2, SHERPA integrates heuristics into the RL framework by defining them as rule-based policies generated by advanced LLMs from human-provided task descriptions.

A. Preliminaries

RL problems are commonly formulated as Markov Decision Processes (MDPs). An MDP is defined by a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $P(s_{t+1} \mid s_t, a_t)$ is the state transition probability function, $R(s_t, a_t)$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. At each discrete time step t, the agent observes a state $s_t \in \mathcal{S}$, selects an action $a_t \in \mathcal{A}$ according to its policy $\pi(a_t \mid s_t)$, receives a scalar reward $r_t = R(s_t, a_t)$, and transitions to a new state s_{t+1} determined by P. The objective of the agent is to learn an optimal policy π^* that maximizes the expected cumulative reward

$$G_t = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \,\middle|\, s_t \right]. \tag{1}$$

B. Segmented Heuristic Experience Relay for Policy Assistance

SHERPA is an RL framework that leverages rule-based heuristics by injecting them segmentally into episodes and incorporating their outcomes into the policy through standard RL optimization. To distinguish the source of experience, SHERPA maintains two separate replay buffers:

$$\mathcal{D}_{\text{H}} = \{(s_t, a_t^{\pi_H}, r_t, s_{t+1}) \mid a_t^{\pi_H} = \pi_H(s_t)\} \tag{2}$$

$$\mathcal{D}_{\text{RL}} = \{(s_t, a_t^{\pi_\theta}, r_t, s_{t+1}) \mid a_t^{\pi_\theta} \sim \pi_\theta(a \mid s_t)\}, \tag{3}$$

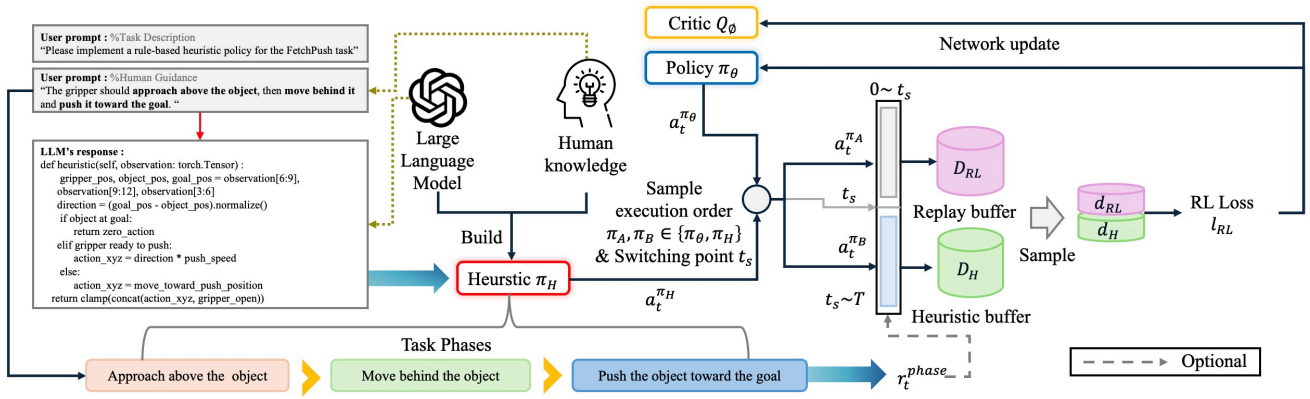

Fig. 2. An overview of SHERPA. Given a task description, an LLM generates a heuristic policy π_H. For each training episode, SHERPA randomly determines the execution order between the heuristic policy π_H and the learning policy π_θ, and samples a switching point t_s. The actions a_t from π_H and π_θ are stored in separate replay buffers, $\mathcal{D}_{\text{H}}$ and $\mathcal{D}_{\text{RL}}$, respectively, and are used equally to compute the RL losses. Optionally, a phase-specific reward signal r_t^{phase} can be applied to further guide the training process.

where π_H is a heuristic policy.

At the beginning of each episode, SHERPA randomly samples two variables: the **switching point** $t_s \sim \mathcal{U}(1, l-1)$, where l is the episode length, and the **execution order**, represented as a random permutation $\pi_A, \pi_B \in \{\pi_\theta, \pi_H\}$ with $\pi_A \neq \pi_B$. The policy π_A governs the initial segment until t_s, after which control is relayed to π_B. This relay-based handover ensures that both policies contribute to the trajectory while maintaining temporal coherence. Actions at each timestep t are then selected as:

$$\hat{a}_t = \begin{cases} a_t^{\pi_A}, & \text{if } t < t_s \\ a_t^{\pi_B}, & \text{otherwise} \end{cases} \tag{4}$$

Traditional methods often use expert or heuristic actions via imitation losses, such as behavior cloning:

$$\mathcal{L}_{\text{BC}} = \mathbb{E}_{(s, a^{\pi_H}) \sim \mathcal{D}_{\mathcal{H}}} \left[\| \pi_\theta(s) - a^{\pi_H} \|^2 \right] \tag{5}$$

where the policy is explicitly trained to match heuristic actions drawn from a dataset $\mathcal{D}_{\mathcal{H}}$. In contrast, SHERPA does not aim to imitate the heuristic policy. Instead, it learns from the executed behaviors of both the heuristic and agent policies through reinforcement learning alone.

During training, SHERPA samples an equal number of transitions from $\mathcal{D}_{\text{H}}$ and $\mathcal{D}_{\text{RL}}$, and computes a mean squared error (MSE) loss against Q-value targets:

$$\mathcal{L}_{\text{MSE}} = (Q_\phi(s_i, a_i) - y_i)^2 + (Q_\phi(s_j, a_j) - y_j)^2, \tag{6}$$

where $(s_i, a_i, r_i, s_i') \sim \mathcal{D}_{\text{H}}$ and $(s_j, a_j, r_j, s_j') \sim \mathcal{D}_{\text{RL}}$. The targets are computed using the current policy:

$$y_i = r_i + \gamma Q_{\phi'}(s_i', \pi_\theta(s_i')), \quad y_j = r_j + \gamma Q_{\phi'}(s_j', \pi_\theta(s_j')). \tag{7}$$

The policy is trained with samples from both $\mathcal{D}_{\text{H}}$ and $\mathcal{D}_{\text{RL}}$ to maximize the expected Q-value:

$$\pi_\theta \leftarrow \pi_\theta + \eta_\pi \left(\nabla_\theta Q_\phi(s_i, \pi_\theta(s_i)) + \nabla_\theta Q_\phi(s_j, \pi_\theta(s_j)) \right), \tag{8}$$

where $s_i \sim \mathcal{D}_{\text{H}}$ and $s_j \sim \mathcal{D}_{\text{RL}}$.

Notably, SHERPA enables learning from both RL and heuristic signals using return-maximizing objective, without behavior cloning or auxiliary objectives. The segmented relay mechanism is particularly effective for long-horizon tasks, as it exposes the agent to intermediate states that are otherwise difficult to reach, facilitating exploration while allowing the learned policy to surpass the heuristic.

C. Reward Augmentation from SHERPA

To further support learning in long-horizon tasks, SHERPA introduces a *phase reward* mechanism. Unlike the environment's native reward signal, this reward is granted when the agent reaches a milestone state identified by condition-based rules in the heuristic policy. These milestones correspond to key subgoals such as "object grasped" or "goal region entered," decomposing the task into meaningful intermediate phases.

For formalization, we define a condition set $\mathcal{C}$ of K mutually exclusive conditions predefined in π^H:

$$\mathcal{C} = \{C_1, C_2, \ldots, C_K\}, \qquad C_k : \mathcal{S} \to \{0, 1\}, \tag{9}$$

$$d_k \in \{0, 1\}. \tag{10}$$

Here, $C_k(s_t) = 1$ indicates that the k-th condition is satisfied at state s_t, and d_k is an indicator initialized to 0, set to 1 one step after the first satisfaction of condition k and remaining 1 thereafter. The phase reward r_t^{phase} is then defined as

$$r_t^{\text{phase}} = C_k(s_t) \cdot (1 - d_k), \tag{11}$$

and the total training reward is $r_t = r_t^{\text{env}} + r_t^{\text{phase}}$. This preserves reward sparsity while providing informative feedback in tasks where final rewards are delayed or difficult to achieve.

IV. Experiment

In our experimental evaluation, we investigate four key questions: 1) Performance Comparison: What performance advantages does SHERPA offer over existing methods on

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT METHODS ON SHORT-HORIZON TASKS

Category	Method	FetchReach		FetchPush		FetchPickPlace		FetchSlide		PandaPush		PandaPick		
		Max	AUC	Max	AUC	Max	AUC	Max	AUC	Max	AUC	Max	AUC	Norm
IL	GAIL	100.0	39.86	52.0	5.82	44.0	3.42	36.0	2.46	64.0	8.70	48.0	4.37	0.117
	AIRL	100.0	98.96	96.0	37.22	80.0	22.13	20.0	0.25	60.0	8.14	40.0	4.07	0.304
	IQ-Learn	100.0	2.15	32.0	4.20	28.0	2.46	10.0	0.29	52.0	11.04	50.0	9.85	0.054
	ValueDICE	100.0	72.46	76.0	28.21	50.0	7.17	52.0	12.49	98.0	49.27	100.0	73.60	0.451
	DemoDICE	100.0	**99.82**	50.0	6.41	19.2	3.82	28.0	1.38	72.0	26.24	100.0	62.82	0.347
RL	TD3	20.2	17.98	0.8	0.02	0.4	0.01	1.4	0.15	0.6	0.01	0.2	<0.01	0.031
	SAC	0.2	<0.01	1.0	0.04	1.0	0.03	1.8	0.08	16.6	7.63	8.8	3.11	0.019
RL+Heu	RLingua	100.0	98.84	100.0	**85.35**	88.6	45.82	16.8	3.06	63.0	42.99	64.4	43.68	0.584
	SHERPA (Ours)	100.0	96.99	100.0	82.25	100.0	**84.64**	83.8	**43.80**	100.0	**96.35**	100.0	**95.54**	**0.989**
Heuristic	–	100.0	–	75.8	–	94.5	–	52.2	–	82.6	–	90.8	–	–

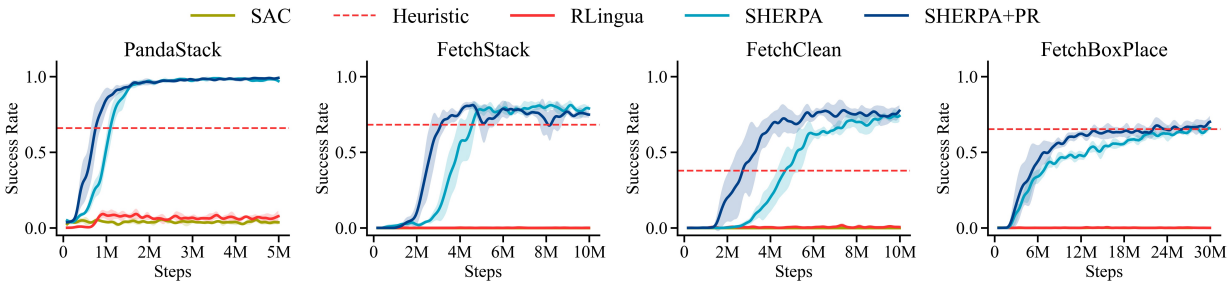

Fig. 3. Comparison of learning curves between SHERPA and baselines on long-horizon tasks.

short- and long-horizon tasks? 2) Ablation on SHERPA Variants: How do different heuristic injection strategies and the use of phase rewards affect performance? 3) Robustness to Heuristic Quality: How robust is SHERPA to variations in the quality of the heuristic policy? 4) Real-world Scalability: Can SHERPA be effectively extended and applied to real-world robotic domains?

A. Experimental Setup

1) Tasks: We evaluate our approach on ten robotic manipulation tasks drawn from two widely used benchmark suites: `Fetch` [24] and `Panda` [25]. They use different robot embodiments and base simulators: the `Fetch` suite uses the MuJoCo simulator, whereas the `Panda` suite is based on PyBullet. This allows us to evaluate the framework under different contact and friction dynamics, even when the task contexts are similar. The Fetch suite includes `FetchReach`, `FetchPush`, `FetchPickAndPlace`, `FetchSlide`, `FetchStack`, `FetchClean`, and `FetchBoxPlace`; the Panda suite includes `PandaPush`, `PandaPickAndPlace`, and `PandaStack`. Among these, `PandaStack`, `FetchStack`, `FetchClean` and `FetchBoxPlace` are classified as long-horizon tasks involving multiple sequential subgoals. All tasks adopt a sparse reward setting, where the agent receives a binary success signal only upon task completion. For real-world validation, a subset of these tasks was also evaluated on a UR5 platform (Sec. IV-D).

2) Methods: We compare SHERPA against RL algorithms, widely used IL methods, and a recent heuristic-guided framework. SHERPA is built upon SAC [26] without Hindsight Experience Replay (HER) [27], while the RL baselines SAC and TD3 [28] are evaluated with HER enabled to ensure competitive performance. For IL baselines, we include GAIL [16], AIRL [17], IQ-Learn [19], ValueDICE [18], and DemoDICE [22]. We also evaluate RLingua [7], which injects heuristic actions probabilistically with an imitation loss; as the official implementation is unavailable, we reimplement it from the paper with HER. RLingua serves as a key comparison point given its structural differences from SHERPA (step-wise vs. segmented injection; imitation-based vs. return-maximizing objective). To analyze component contributions, we consider two SHERPA variants: (1) HERPA, which replaces segmented relay with probabilistic step-wise injection; and (2) +PR, which augments training with phase rewards.

3) Preparation of Heuristics: The heuristics in SHERPA are rule-based policies generated by providing a human-written task description to GPT-4o (Figure 2), each consisting of conditional statements that define a step-by-step task strategy. To assess robustness to heuristic quality, we use two variants: a *superior* version manually refined for maximum performance, and an *inferior* version intentionally simplified.

TABLE II
ABLATION RESULTS ON SHORT- AND LONG-HORIZON TASKS, EVALUATED BY MAX (%) AND AUC (%)

	HERPA		SHERPA		HERPA+PR		SHERPA+PR	
Task	Max	AUC	Max	AUC	Max	AUC	Max	AUC
Short-horizon tasks								
FetchReach	100	**98.9**	100	97.0	–	–	–	–
FetchPush	100	64.8	100	**82.2**	100	70.4	100	72.8
FetchPick&Place	99.8	82.9	100	84.6	100	78.7	100	**85.6**
FetchSlide	76.8	40.8	83.8	43.8	76.0	30.5	91.6	**48.6**
PandaPush	100	**97.1**	100	96.4	100	96.5	100	96.4
PandaPick&Place	100	**95.6**	100	95.5	100	93.7	100	94.6
Long-horizon tasks								
PandaStack	12.9	4.3	100	80.1	10.6	4.23	100	**86.9**
FetchStack	0.00	0.00	89.0	53.0	0.00	0.00	89.2	**58.8**
FetchClean	0.00	0.00	83.8	36.8	0.00	0.00	87.4	**54.6**
FetchBoxPlace	1.00	<0.01	76.5	48.0	1.00	<0.01	78.6	**54.1**

Unless otherwise noted, the superior heuristic is used in the main experiments, with the ablation study comparing both.

B. Experimental Result and Comparison

1) Enabling RL training: Table I summarizes performance across six representative tasks, each evaluated over five fixed seeds. We report *Max* (average peak success rate per seed) and *AUC* (area under the success-rate curve, normalized by $T_{\max}$):

$$\text{AUC}(\%) = \frac{1}{T_{\max}} \int_0^{T_{\max}} f(x)\, dx. \tag{12}$$

We additionally report a normalized score (*Norm*), the average normalized AUC across tasks, to jointly assess final capability and learning efficiency.

In the simplest task, `FetchReach`, IL methods, RLingua, and SHERPA all achieved near-perfect Max, whereas RL baselines failed even with HER. In more challenging tasks with randomized goals, SHERPA not only matched but surpassed the heuristic policies in both Max and AUC—for instance, 83.8 Max and 43.80 AUC on `FetchSlide` versus ValueDICE's 52.0 and 12.49. The same trend held across the Panda suite. These results, culminating in the highest normalized score (Norm = 0.989), confirm that segmented heuristic integration with return-maximizing optimization yields both efficient and consistently strong performance.

2) Long-Horizon Domain Performance: Long-horizon tasks are particularly challenging for RL agents due to the combination of sparse rewards and extended temporal credit assignment. While RLingua shares with SHERPA the high-level idea of leveraging heuristic knowledge, its design focuses on short- to medium-horizon tasks and lacks mechanisms to sustain learning signals across temporally distant subgoals. Consequently, existing step-wise heuristic-guided methods such as RLingua show limited progress in long-horizon settings. To demonstrate this distinction, we evaluate SHERPA and SHERPA+PR on four long-horizon tasks: `FetchStack`, `PandaStack`, `FetchClean`, and `FetchBoxPlace`. We compare against heuristic policy, RLingua, and SAC.

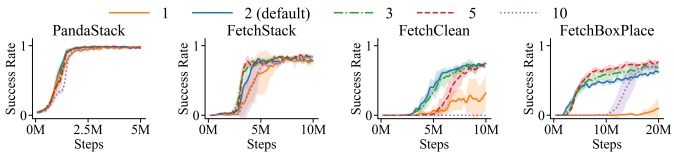

Fig. 4. Comparison of learning curves of SHERPA under different number of segments.

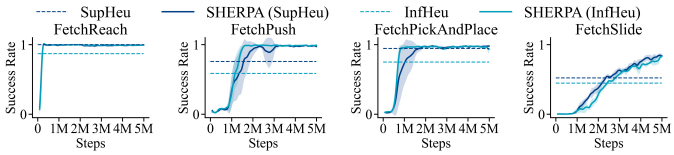

Fig. 5. Comparison of learning curves of SHERPA under different heuristic variations.

As shown in Figure 3, both SAC and RLingua fail to achieve measurable progress in any long-horizon task despite RLingua's reasonable performance on shorter-horizon domains, whereas SHERPA achieves consistent improvement across all four settings, surpassing the heuristic baseline. This stems from SHERPA's segmented relay, which preserves effective learning signals even when subgoals are far apart. SHERPA+PR further accelerates convergence in every long-horizon task, with benefits extending beyond long-horizon domains. Together, SHERPA's structured relay and phase-based rewards enable it to solve long-horizon problems with sparse reward.

3) Comparison with Imitation Learning: IL methods were evaluated using 500 successful trajectories per task collected via heuristics. As shown in Table I, IL methods achieved strong performance on simple tasks such as `FetchReach`, but consistently struggled on object-manipulation tasks with randomized goals. Increasing demonstrations from 500 to 2,500 yielded only marginal gains, suggesting that IL alone cannot handle sparse rewards and goal variability. Among IL baselines, AIRL was relatively robust due to its explicit reward estimation, yet SHERPA outperformed all IL methods by a substantial margin, achieving multiple-times-higher performance on tasks such as `FetchSlide`. Unlike static demonstrations, heuristics adapt to the agent's state during training, enabling further optimization.

C. Ablation Study

In our ablation study, we investigate three key aspects of SHERPA: (1) the effects of the segmented relay and phase rewards, (2) the impact of varying number of segments, and (3) the impact of heuristic performance.

1) Effects of Segmented Relay and Phase Rewards: We analyze the functional contributions of two core components in SHERPA: the segmented heuristic relay and phase-specific rewards. Table II shows that segmentation alone leads to substantial improvements in long-horizon tasks. While the performance differences between HERPA and SHERPA are marginal for short-horizon tasks, SHERPA significantly outperforms HERPA on long-horizon tasks. For example, in

PandaStack, the AUC increases from 4.3 to 80.1, and the Max success rate reaches 100. These results indicate that the segmented heuristic relay enables stable and progressive learning when subgoals are sparse or delayed.

Furthermore, phase-specific rewards provide additional gains when used alongside segmentation, as SHERPA+PR consistently improves over SHERPA. Specifically, the AUC increases from 36.8 to 54.6 in FetchClean and from 80.1 to 86.9 in PandaStack. In contrast, HERPA+PR does not exhibit similar improvements, suggesting that the effectiveness of phase-specific rewards is conditional on semantically segmented execution.

Overall, these findings highlight the segmented relay as a key enabler for learning in complex, long-horizon tasks. By organizing heuristic guidance in a segmented manner, SHERPA surpasses stochastic step-wise assistance used in HERPA. Building on this foundation, phase-specific rewards improve training efficiency and stability when combined with segmentation. Together, these components distinguish SHERPA-based methods from existing approaches and enable robust performance across a wide range of task complexities.

2) Impact of Varying Number of Segments: In Figure 4, we analyze intra-episode interaction by varying the number of segments to 1, 2 (default), 3, 5, and 10. Notably, the 1-segment setting corresponds to a configuration without relay, where no intra-episode guidance exchange occurs. When using a single segment, tasks composed of relatively simple and repetitive behaviors, such as Stack, can still be solved. However, tasks involving more complex behaviors fail under this setting. This failure arises because guidance becomes weakly aligned with the agent's on-policy state distribution, reducing the availability of state-relevant feedback during execution. When the number of segments exceeds 2, intra-episode relay becomes increasingly frequent, leading to fragmented relay within an episode. While finer-grained segmentation can occasionally improve asymptotic performance, it generally slows learning due to increased policy switching and degrades stability when segmentation becomes excessive. In particular, the FetchClean task fails to converge under high segmentation, which disrupts temporal coherence and diminishes the benefits of structured intra-episode guidance. Overall, the 2-segment design adopted in SHERPA provides a general and well-founded starting point for structured intra-episode relay, balancing guidance effectiveness with learning stability.

3) Impact of Heuristic Performance: The superior heuristics used in the main experiments exhibit high performance. To assess SHERPA's robustness to heuristic quality, we intentionally degraded these heuristics, yielding the following reductions in success rates: (FetchReach) 100.0%→87.0% by lowering move velocity, (FetchPush) 75.8%→58.7% by relaxing rule-satisfaction thresholds, (FetchPickAndPlace) 94.5%→75.0% by lowering move velocity, and (FetchSlide) 52.2%→44.7% by removing contact-angle correction.

Figure 5 compares SHERPA's performance between inferior and superior heuristics. Surprisingly, SHERPA with inferior

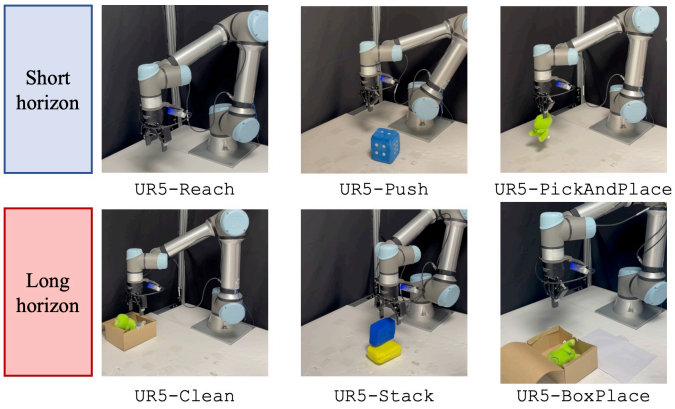

Fig. 6. Evaluation of SHERPA across real-world manipulation tasks (Reach, Push, Pick&Place, Stack, Clean, and BoxPlace).

heuristics exhibited only a slight or negligible drop in performance, far smaller than the direct performance gap between the superior and inferior heuristic baselines. Moreover, in every case, SHERPA surpassed the fixed heuristic's performance, demonstrating its ability to refine and improve upon imperfect guidance. In long-term training, the maximum success rates achieved with inferior heuristics were comparable to those achieved with superior heuristics.

These findings suggest that SHERPA's primary advantage lies not in the exact precision of heuristic actions, but in the consistent logical structure they provide. By correcting suboptimal actions along heuristic trajectories, SHERPA derives comparable benefits from both superior and inferior heuristics. In effect, heuristics serve as a "takeoff point," enabling the RL policy to progressively outperform them. This robustness implies that SHERPA does not require strictly precise heuristics—relatively coarse ones can still drive successful learning, while IL performance degrades with demonstration quality.

D. Real-World Experiments

We conducted sim-to-real transfer experiments on a UR5 platform across six tasks—three short-horizon and three long-horizon—as illustrated in Figure 6. The state input consists of the gripper position obtained from the robot's internal kinematics and the object position specified at the beginning of each episode.

In the short-horizon tasks, SHERPA handled randomized goals and object positions without difficulty. More importantly, all three long-horizon tasks also transferred successfully, demonstrating SHERPA's robustness to extended temporal dependencies in real-world settings. Among these, BoxPlace presented a notable sim-to-real gap: the physical lid structure differed from the simulated one. We addressed this by generating a revised heuristic based on sliding the lid rather than gripping it. The successful execution confirmed that SHERPA can readily adapt to real-world discrepancies through targeted heuristic modification, making it a practical framework for challenging long-horizon transfer scenarios.

V. Discussion

We present SHERPA, a novel framework that integrates heuristics into RL via a segmented relay mechanism, enabling a smooth transition from heuristic-driven to autonomous policy optimization in sparse-reward, long-horizon robotic domains. A key strength is that the agent refines heuristic-augmented trajectories through RL optimization, ultimately surpassing the heuristic's own performance. Across ten manipulation tasks including four long-horizon benchmarks, SHERPA consistently outperforms RL, IL, and heuristic-guided baselines, and unlike RLingua, maintains stable learning in extended multi-stage tasks even under imperfect heuristics. Real-world experiments on a UR5 robot further confirm that SHERPA transfers successfully to physical settings, including challenging long-horizon tasks with sim-to-real discrepancies.

Despite these strengths, heuristic stability is not guaranteed for unfamiliar tasks, and we found that a large target network update interval (e.g., every 128 steps) was critical to performance, likely mitigating discrepancies between the two policy sources. Future work will focus on addressing these limitations, extending SHERPA to broader classes of imperfect action-level controllers beyond LLM-generated guidance, and incorporating vision-based learning to broaden its applicability to real-world scenarios with high-dimensional perceptual inputs.

VI. Acknowledgment

This work was supported in part by the Korea Research Institute for Defense Technology Planning and Advancement (KRIT) grant funded by the Defense Acquisition Program Administration (DAPA) (No. KRIT-CT-23-003/20%, Development of AI Researchers Based on Deep Reinforcement Learning and Establishment of Virtual Combat Experiment Environment), and in part by the IITP (RS-2022-I1220951-LBA/10%, RS-2022-I1220953-PICA/10%), NRF (RS-2024-00353991-SPARC/10%, RS-2023-00274280-HEI/10%, RS-2024-00358416-AutoRL/20%), KEIT (RS-2025-25453780/10%), and KIAT (RS-2025-25460896/10%) grants funded by the Korean government.

References

[1] G. Lee, M. Kim, Y. Lee, M. Lee, and B.-T. Zhang, "Neural collage transfer: Artistic reconstruction via material manipulation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 2394–2405.

[2] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, 2021.

[3] Y. Yan, A. H. Chow, C. P. Ho, Y.-H. Kuo, Q. Wu, and C. Ying, "Reinforcement learning for logistics and supply chain management: Methodologies, state of the art, and future opportunities," *Transportation Research Part E: Logistics and Transportation Review*, vol. 162, p. 102712, 2022.

[4] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[5] Z. Wu, S. Ye, M. Natarajan, and M. C. Gombolay, "Diffusion-reinforcement learning hierarchical motion planning in multi-agent adversarial games," 2025. [Online]. Available: https://arxiv.org/abs/2403.10794

[6] L. Kästner, T. Buiyan, X. Zhao, L. Jiao, Z. Shen, and J. Lambrecht, "Arena-rosnav: Towards deployment of deep-reinforcement-learning-based obstacle avoidance into conventional autonomous navigation systems," 2021. [Online]. Available: https://arxiv.org/abs/2104.03616

[7] L. Chen, Y. Lei, S. Jin, Y. Zhang, and L. Zhang, "Rlingua: Improving reinforcement learning sample efficiency in robotic manipulations with large language models," *IEEE Robotics and Automation Letters*, 2024.

[8] A. Silva and M. Gombolay, "Encoding human domain knowledge to warm start reinforcement learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 6, 2021, pp. 5042–5050.

[9] Y. Du, O. Watkins, Z. Wang, C. Colas, T. Darrell, P. Abbeel, A. Gupta, and J. Andreas, "Guiding pretraining in reinforcement learning with large language models," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 23–29 Jul 2023, pp. 8657–8677. [Online]. Available: https://proceedings.mlr.press/v202/du23f.html

[10] A. Najar, O. Sigaud, and M. Chetouani, "Interactively shaping robot behaviour with unlabeled human instructions," *Autonomous Agents and Multi-Agent Systems*, vol. 34, no. 2, p. 35, 2020.

[11] Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar, "Eureka: Human-level reward design via coding large language models," 2024. [Online]. Available: https://arxiv.org/abs/2310.12931

[12] H. Li, X. Yang, Z. Wang, X. Zhu, J. Zhou, Y. Qiao, X. Wang, H. Li, L. Lu, and J. Dai, "Auto mc-reward: Automated dense reward design with large language models for minecraft," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16 426–16 435.

[13] J. Rocamonde, V. Montesinos, E. Nava, E. Perez, and D. Lindner, "Vision-language models are zero-shot reward models for reinforcement learning," 2024. [Online]. Available: https://arxiv.org/abs/2310.12921

[14] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, vol. 1, 1988.

[15] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[16] J. Ho and S. Ermon, "Generative adversarial imitation learning," *CoRR*, vol. abs/1606.03476, 2016. [Online]. Available: http://arxiv.org/abs/1606.03476

[17] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," *CoRR*, vol. abs/1710.11248, 2017. [Online]. Available: http://arxiv.org/abs/1710.11248

[18] I. Kostrikov, O. Nachum, and J. Tompson, "Imitation learning via off-policy distribution matching," *CoRR*, vol. abs/1912.05032, 2019. [Online]. Available: http://arxiv.org/abs/1912.05032

[19] D. Garg, S. Chakraborty, C. Cundy, J. Song, and S. Ermon, "Iq-learn: Inverse soft-q learning for imitation," *CoRR*, vol. abs/2106.12142, 2021. [Online]. Available: https://arxiv.org/abs/2106.12142

[20] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," *CoRR*, vol. abs/1805.01954, 2018. [Online]. Available: http://arxiv.org/abs/1805.01954

[21] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," *arXiv preprint arXiv:2110.06169*, 2021.

[22] G.-H. Kim, S. Seo, J. Lee, W. Jeon, H. Hwang, H. Yang, and K.-E. Kim, "Demodice: Offline imitation learning with supplementary imperfect demonstrations," in *International Conference on Learning Representations*, 2022.

[23] Z. Li, T. Xu, Z. Qin, Y. Yu, and Z.-Q. Luo, "Imitation learning from imperfection: Theoretical justifications and algorithms," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. [Online]. Available: https://openreview.net/forum?id=vO04AzsB49

[24] M. Plappert, M. Andrychowicz, A. Ray, B. McGrew, B. Baker, G. Powell, J. Schneider, J. Tobin, M. Chociej, P. Welinder, V. Kumar, and W. Zaremba, "Multi-goal reinforcement learning: Challenging robotics environments and request for research," 2018.

[25] Q. Gallouédec, N. Cazin, E. Dellandréa, and L. Chen, "panda-gym: Open-Source Goal-Conditioned Environments for Robotic Learning," *4th Robot Learning Workshop: Self-Supervised and Lifelong Learning at NeurIPS*, 2021.

[26] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[27] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 5048–5058.

[28] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," 2018. [Online]. Available: https://arxiv.org/abs/1802.09477