nature communications

Article

https://doi.org/10.1038/s41467-025-66009-y

Discovery of the reward function for embodied reinforcement learning agents

Received: 13 May 2025

Accepted: 29 October 2025

Published online: 11 December 2025

Check for updates

Renzhi Lu[1], Zonghe Shao[1], Yuemin Ding[2,3], Ruijuan Chen[4], Dongrui Wu[5], Housheng Su[5], Tao Yang[6], Fumin Zhang[7], Jun Wang[8], Yang Shi[9], Zhong-Ping Jiang[10], Han Ding[11] & Hai-Tao Zhang[1]

Reward maximization is a fundamental principle in both the survival and evolution of biological organisms. In particular, in the contexts of cognitive science and embodied agents, reward-driven behavior has been widely regarded as important in terms of governing complex cognitive abilities such as perception, imitation, and learning. Among the frameworks that are aimed at establishing such abilities, reinforcement learning (RL), which leverages reward maximization to facilitate intelligent decision-making in embodied agents, has been proven to be particularly promising. Importantly, the inherent complexity and uncertainty of real-world tasks pose significant challenges when designing effective reward functions for embodied RL agents. Conventional methods typically rely on manually engineered or externally tuned reward signals, and therefore require significant domain expertise, associated with considerable human efforts and a long convergence time; these issues may even trigger mission failure. This work introduces a bilevel optimization framework that discovers optimal reward functions for embodied reinforcement learning agents through a mechanism called regret minimization. The approach accelerates policy optimization and enhances adaptability across diverse tasks. These findings can support the broader adoption of embodied RL agents in the behavioral and computational sciences and neurosciences, thereby paving the way for artificial general intelligence.

Embodied artificial intelligence (EAI) aims to enable embodied agents with specific morphologies (e.g., joints, limbs, and motion capabilities) to learn effective control policies[1]. A long-standing debate in this field concerns whether these policies should be general[2] or task-specific[3]. Recent advances in large-scale data technology and cloud computing[4] have fueled interest in artificial general intelligence (AGI), which aims to develop EAI systems exhibiting general adaptability and embodied intelligence[5,6]. Naturally, this motivates the pursuit of universal control policies that allow embodied agents to adapt seamlessly to various scenarios[7]. However, existing machine learning frameworks seldom provide robust solutions for such generality. Reinforcement learning (RL), a versatile decision-making paradigm based on reward maximization[8], has achieved notable breakthroughs across various domains over the past few years. For example, RL has surpassed human performance on Atari games[9] and StarCraft II[10]. Another breakthrough in RL is that of robotic control. Robots acquire policies in a similar way to human learning[11] and even engage in lifelong learning[12]. Recently, RL has demonstrated immense potential in biomedical applications, ranging from uncovering cellular movement mechanisms[13] to facilitating rehabilitation[14]. Most recently, DeepSeek has incentivized the reasoning of large language models (LLMs) via RL without supervised fine-tuning as a preliminary step and has achieved remarkable reasoning capabilities[15]. These advances suggest that RL could serve as a foundational framework for general EAI.

EAI and associated abilities can be understood as efforts that are aimed at maximizing rewards, which is a core principle of both

A full list of affiliations appears at the end of the paper. e-mail: yueminding@tecnun.es; zht@mail.hust.edu.cn

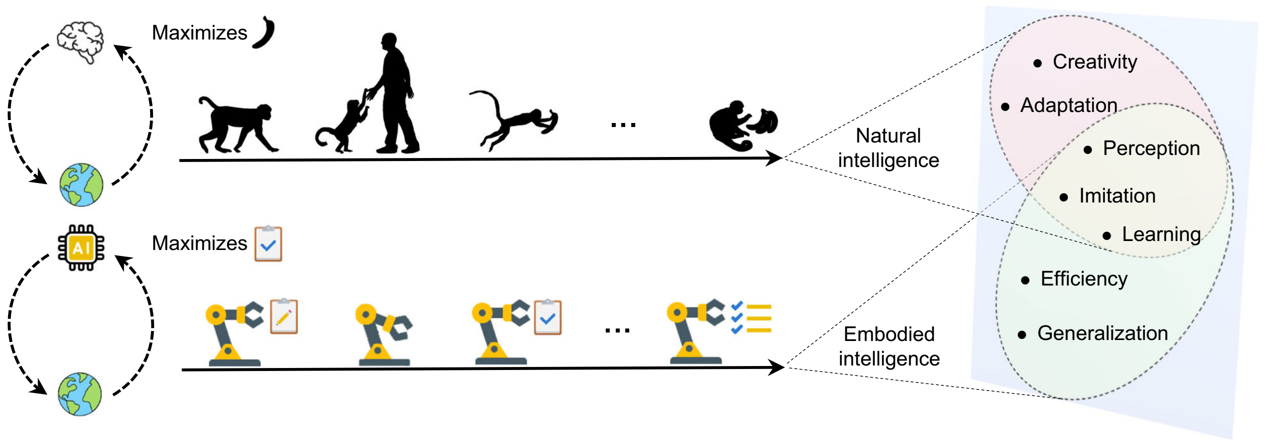

Fig. 1 | The reward maximization principle of natural and embodied intelligence. The reward maximization principle drives agents to develop intelligence and associated cognitive abilities. For example, a monkey can learn to shake hands with humans to maximize its banana rewards, whereas a robot can acquire the ability to follow predefined instructions, thereby optimizing task execution. Attainment of these objectives requires the development of complex behaviors that reflect diverse aspects of cognitive intelligence, such as perception, imitation, and learning. Silhouettes adapted from PhyloPic.org (credits: Andrew A. Farke, Zimices, Kai R. Caspar); icons from Icons8.com.

biological[16] and artificial systems[17,18]. Accordingly, reward is sufficient to drive behavior that involves abilities intrinsic to both natural and embodied intelligence, including perception, imitation, learning, and generalization[19–21]. This may be surprising, as the diversity of such abilities seems to be at odds with a general objective. However, the natural world, in which animals and humans live, and the environments that EAI systems encounter are inherently complex, and sophisticated attributes are required to succeed in such settings. For example, as shown in Fig. 1, the monkey brain can be conceptualized as exhibiting a form of natural intelligence, and the behaviors thereof can be interpreted as following a policy that maximizes cumulative rewards (bananas). Natural organisms tend to embrace various types of cognitive intelligence that improve life[22]. Likewise, in robotic EAI systems, the reward maximization principle leads the robots to develop diverse intelligence-driven abilities[17]. The physical structure of the robot functions as the execution system that receives orders from an embodied intelligent brain that optimizes task execution. During this process, the robotic agent seeks to perform optimally and, at times, even demonstrates capabilities in certain areas that closely resemble those of natural intelligence[12,23,24]. Therefore, the general objective of reward maximization includes many or all of the goals of intelligence[25–27]. Furthermore, embodied RL agents that learn how to maximize rewards via trial-and-error can acquire behaviors that exhibit most of the capabilities described above. Some researchers have suggested that powerful embodied RL agents could open a path toward AGI[28,29]. Specifically, during RL, rewards serve as immediate feedback signals to evaluate the embodied agent's ability, shaping the outcomes of RL in terms of imparting intelligence to the agent[30].

The quality of the reward function plays a pivotal role in fostering human-like intelligence, as it directly influences the performance and generalizability of RL intelligence, particularly in open-world EAI systems[7]. On the one hand, different reward signals shape distinct learning dynamics, giving rise to diverse forms of RL intelligence[31,32]. On the other hand, a high-quality reward function may offer deeper insights into the underlying mechanisms, driving RL intelligence development and the emergent capabilities. Traditionally, reward function design relies on an iterative, manual trial-and-error process[33], as illustrated in Fig. 2a. This manual process is not only time consuming and resource intensive[34] but also susceptible to overfitting[35], creating learned agents that display unintended behaviors, such as reward hacking[36], or even inducing the cobra effect paradox. Unsurprisingly, researchers agree that automatic design of the reward function would both eliminate the significant manual demand and improve agent performance[37], in turn broadening the applicability of RL across a wide variety of tasks. The design of reward functions for embodied RL agents can be automated via two primary approaches: acquiring external information[38–42] or leveraging self-learning processes[43]. When considering access to external information, reverse engineering the reward function offline using expert demonstrations[38] facilitates rapid application of RL to complex tasks[40,44]. Meanwhile, active learning of reward functions via human feedback[45] enables real-time and interactive reward optimization[39,42]. The two approaches are shown in Fig. 2b. Inverse RL (IRL) and active reward learning (ARL) have significantly improved RL performance in a variety of simulated environments[44], including robotics[46] and LLMs[47]. However, IRL often requires optimal demonstrations[41], and ARL needs dense and substantial human feedback, which can be challenging in real-world EAI tasks[48]. Thus, reliance on external information remains a barrier to the swift deployment of embodied RL in real-world scenarios[49]. A promising alternative approach is automatic learning of the reward function on the basis of the past performance of the agent[50–52]. Among the various methods, intrinsic motivation-based reward learning is the most widely adopted[43]. Here, the reward function is generated via heuristics[53] or task-independent metrics[54]. However, despite progress in constructing and exploring reward function structures from various perspectives, a widely applicable method of reward function design that is fully independent of external knowledge has yet to be found. Furthermore, clarifying the formalization of the optimal reward function remains a challenging task.

This work overcomes the existing research bottleneck and takes a significant step toward discovering the optimal reward functions for embodied RL agents. The core insight is that the optimal reward function can be automatically discovered on the basis of the past performance of embodied RL agents with minimal reliance on task-specific prior knowledge. This enables the embodied RL agent to determine the optimal policy rapidly while ensuring convergence. Building on this insight, an approach for discovering the optimal reward function for embodied RL agents is proposed in this work. In this context, the optimal reward function is first defined, and a framework is then developed that seamlessly integrates the search for the optimal reward function with the RL process. While previous works have explored various definitions of the optimal reward function[55,56], this work formulates an intuitive and straightforward definition based on regret minimization, as shown in Fig. 2c. This yields a practical form

https://doi.org/10.1038/s41467-025-66009-y

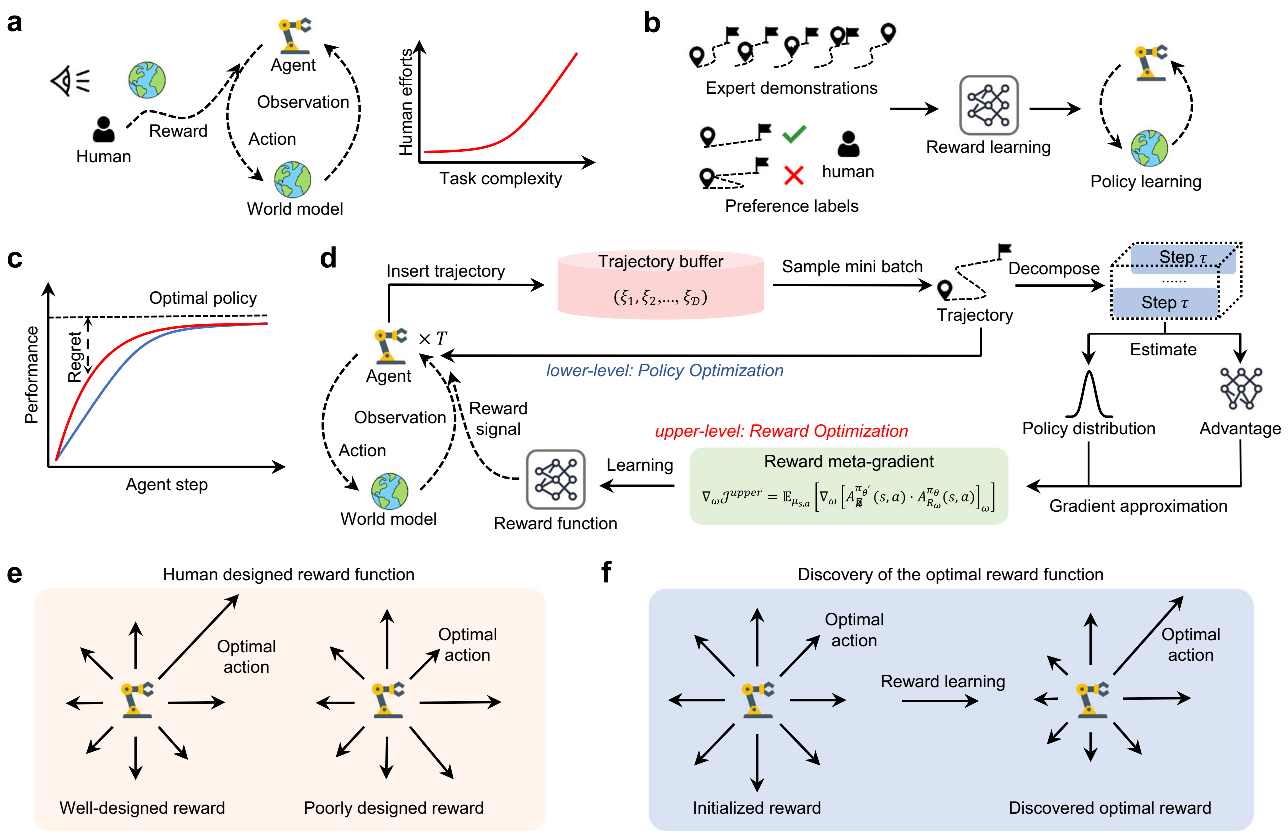

Fig. 2 | Motivation and framework of the proposed approach. a Humans observe the world and manually design a reward function, which becomes increasingly difficult as task complexity increases. **b** Two schemes for designing reward functions automatically. One involves learning from expert demonstrations, and the other leverages human preference labels. **c** Regret of policies learned with different reward functions. The red and blue lines correspond to a more and a less effective reward function, respectively. A better reward function is associated with less policy regret and faster convergence. **d** Overview of the proposed bilevel optimization framework for optimal reward function discovery. In the lower-level, the embodied RL agent interacts with a world model, receives reward signals, and optimizes policy. Interaction trajectories are stored in a buffer. In the upper-level, a mini-batch of trajectories is sampled and decomposed into interaction steps for estimation of the policy distribution and the advantage function. The reward function is then updated via the approximated meta gradient (Theorem 2). This iterative process ensures simultaneous optimization of both the reward function and the policy of the RL agent. **e** Comparison of well-designed and poorly designed reward functions. A well-designed reward function assigns high rewards (represented by the arrow length) to the optimal action. A poorly designed reward function fails to distinguish the optimal action, resulting in incorrect actions by the agent. **f** Discovery of the optimal reward function. The reward function is randomly initialized and then learned by the proposed framework. The discovered reward function accurately identifies the optimal action and assigns an appropriate reward to each action. Specifically, higher rewards are associated with more effective actions, whereas lower rewards are assigned to less effective actions. Icons from Icons8.com.

of the optimal reward function that can be easily applied in most RL settings. The discovery of the optimal reward function within the embodied RL agent is conceptualized as a meta learning problem, enabling the integration of optimal reward function discovery with the pursuit of the optimal policy. Although meta learning problems have been explored in prior works[50,57,58], we here propose a meta gradient algorithm specifically tailored to this setting, creating a bilevel optimization framework that incorporates the RL process, in which RL policy optimization is the lower-level process and the discovery of the optimal reward function is the upper-level process. The overall bilevel optimization framework is shown in Fig. 2d. Regret minimization is the optimization objective of the upper-level. At the upper level, a reward function is established to provide reward signals for the agent. At the lower level, the embodied RL agent interacts with a world model and receives reward signals generated by the upper-level reward function. The RL agent seeks to maximize the received rewards. During practical implementation of the proposed approach, the RL agent continuously optimizes its policy, maximizing the cumulative reward signals until a terminal state is reached, at which point the interaction trajectory is stored in a trajectory buffer. During upper-level optimization, a mini-batch of trajectories is randomly sampled from the trajectory buffer and decomposed into multiple interaction steps. These steps are subsequently used to estimate the policy distribution and the advantage function, which serve as the basis of approximation of the reward meta gradient via Theorem 2. After the reward function is updated in the upper-level, the embodied RL agent resumes its interaction with the world model, receiving new reward signals. This alternating optimization is iteratively performed to improve both the policy and the reward function. Furthermore, this method is extended to accommodate both value-based and policy-based agents, and its generalizability is rigorously explored through theoretical analysis. The reward function discovered by the proposed framework is shown to effectively identify the optimal action for the embodied RL agent in a given state, as shown in Fig. 2e, f. Compared to manually designed reward function, the discovered reward function assigns more appropriate rewards to each action, thereby mitigating the risk of incorrect decisions that may arise from poorly designed reward functions (right side of Fig. 2e). For additional implementation details, including the collaborative optimization of the reward function and RL policy, refer to the pseudo code provided in Supplementary Methods. Empirically, the effectiveness of the proposed approach is evaluated in highly realistic simulators and real-world scenarios in a progressive manner, including various tasks such as sparse-reward tasks, high-dimensional control tasks, and real-world tasks. In contrast to the sparse rewards offered in the classic control environments of OpenAI, the reward functions discovered using the proposed approach are dense and, therefore, effectively guide agents that learn policies. In high-dimensional tasks with MuJoCo, a comparison of manually designed reward functions and those autonomously discovered by the proposed approach reveals that the latter effectively focus on critical states, such as success or failure. In real-world evaluation scenarios involving automated energy management of data centers and unmanned systems control of unmanned aerial vehicles (UAVs), the optimal reward function discovery approach not only equips diverse embodied RL agents to perform real-world tasks more effectively than before but also improves their adaptability and accelerates learning. In addition, the proposed approach enables the identification of critical states (e.g., UAV crashes) and reveals the intrinsic latent relationships between rewards and states, despite the absence of an explicit mathematical model. These findings highlight the potential of the proposed framework as a robust means of autonomous discovery of optimal reward functions for embodied RL agents. Consequently, this framework contributes to the scalability of embodied RL agents and highlights its potential for achieving general EAI of real-world scenarios. Furthermore, this work marks a significant milestone in machine learning paradigms, potentially supporting the deployment of embodied RL agents across diverse domains, including the computational and behavioral sciences and neuroscience.

Results

In this section, the proposed approach toward the discovery of optimal reward functions for embodied RL agents is evaluated. Specifically, a series of challenging tasks is first selected, including sparse-reward tasks and high-dimensional control tasks, wherein embodied RL agents may exhibit slow convergence and difficulty converging. Additionally, the approach is assessed in terms of its ability to assist embodied RL agents to address complex real-world tasks in two widely used scenarios. Furthermore, the reward functions discovered by the proposed approach are compared to the baseline rewards, including sparse reward signals, manually designed reward functions, and rewards from three state-of-the-art reward learning methods, including LIRBO[50], Tomax[31], and SASR[54]. The comparison serves to illustrate further the relationship between the reward functions and the tasks, as well as how these functions help embodied RL agents to achieve better performance. The experimental setups for both the numerical experiments and the real-world tasks, and the details on how the comparative methods are implemented, are described in the Supplementary Experimental Details.

Case study of sparse-reward tasks

To examine the effectiveness of the proposed approach when external reward signals are sparse, four sparse external reward tasks are selected from the classical OpenAI control benchmarks: CartPole-v1, AcroBot-v1, FourRoom-v0, and LunarLander-v2, as shown in Fig. 3a. The task setup adheres to the guidelines of OpenAI Gym[59], with the original sparse rewards serving as the baseline for comparison. The sizes of the observation spaces for the four tasks and the action spaces of the agent are shown in Fig. 3b. Deep Q-network (DQN)[60] and proximal policy optimization (PPO)[61] serve as the benchmark RL algorithms. To demonstrate the generalizability of the proposed optimal reward function discovery approach, neither the architecture nor the parameters of the PPO and DQN agents are specifically modified for the tasks. All hyperparameters and network architectures remain consistent across the experiments, as detailed in Supplementary Experiment Details. Each experiment is repeated at least five times with different random seeds, and the average rewards and standard deviations during learning serve as metrics to ensure fairness.

Figure 3c, d shows the average episodic rewards of embodied RL agents when using our proposed approach compared to those of agents that receive sparse reward signals and reward signals from baseline methods. Agents under the proposed reward function outperform the baseline agents in terms of both convergence speed and learning stability. When receiving sparse reward signals, the agent finds it challenging to learn the optimal policy and achieve fast convergence because of the limited number of samples that can be effectively utilized, resulting in low sample efficiency. This low efficiency requires the agent to engage in extensive trial-and-error, thereby slowing the convergence rate. Reward function design methods have relieved this challenge. LIRBO replaces sparse rewards with intrinsic reward signals, and SASR uses a shaping auxiliary function to the same end. Tomax directly learns historical maximum rewards but sometimes does not perform well, as the initial random exploration in sparse-reward tasks does not deliver an adequate reward to the agent. In contrast, the proposed approach delivers dense reward signals directly, thereby improving the sample efficiency. As a result, the agent learns the optimal policy over fewer trials, as demonstrated by the rapid increases in the rewards during the iterations shown in Fig. 3c, d. Moreover, guided by the proposed approach, agents (especially DQN) quickly discover and maintain optimal policies. The superiority of the proposed method is directly attributable to the enhanced sample efficiency afforded by the dense reward signals. The reward function discovered by the proposed approach supports stable convergence, which is guided by the upper-level optimization objective of performance maximization. When the reward function maximizes the upper-level objective, policy learning during lower-level optimization continues to converge toward the optimal policy rather than arbitrarily diverging. Furthermore, in the more complex LunarLander-v2 task, simple sparse rewards fail to guide the creation of a feasible policy. In contrast, using the proposed approach, the embodied RL agents overcome initial exploration errors and ultimately learn policies that maximize the cumulative rewards. Figure 3e shows the average episodic returns and standard errors throughout the entire learning process. The PPO and DQN agents that employ our proposed approach comprehensively outperform the agents under baseline methods. Additionally, as shown in Fig. 3c–e, our approach enhances learning stability and the convergence speed.

Figure 4a–d illustrates the reward signals received by this embodied RL agent during a single interaction episode across four tasks, wherein signals are delivered by both the discovered and the sparse-reward function. In the sparse-reward context, the agent receives reward signals only when it attains the final goal or fails to do so, significantly hindering its ability to learn policy from experience samples, thereby leading to slow learning. In contrast, the proposed approach takes a significant step toward overcoming this challenge. In simple terms, it delivers dense reward signals, which very effectively accelerate learning. Additionally, given the upper-level objective maximization, the reward function learns to assign high-value rewards to goal states, encouraging the agent to focus quickly on the goal state.

Figure 4e–h illustrates the reward surfaces of the sparse reward function and the discovered reward function for each discrete action in Acrobot-v1 task. The reward surfaces are visualized as functions of the two rotational joint angles, θ_1 and θ_2, when their angular velocities $\dot{\theta}_1$ and $\dot{\theta}_2$ are 0. Figure 4e shows that the sparse reward function assigns a reward of -1 for all state and action. Figure 4f–h shows the rewards assigned by our reward function for specific actions: application of -1 torque (action 0), 0 torque (action 1), or 1 torque (action 2) to the actuated joint. The results show that the reward function of this work assigns higher values to actions involving -1 and 1 torque when the joint angles deviate from the downward resting position. This encourages the agent to apply torque in situations when momentum generation can be effectively created. Conversely, the action corresponding to 0 torque receives higher rewards when the agent is near

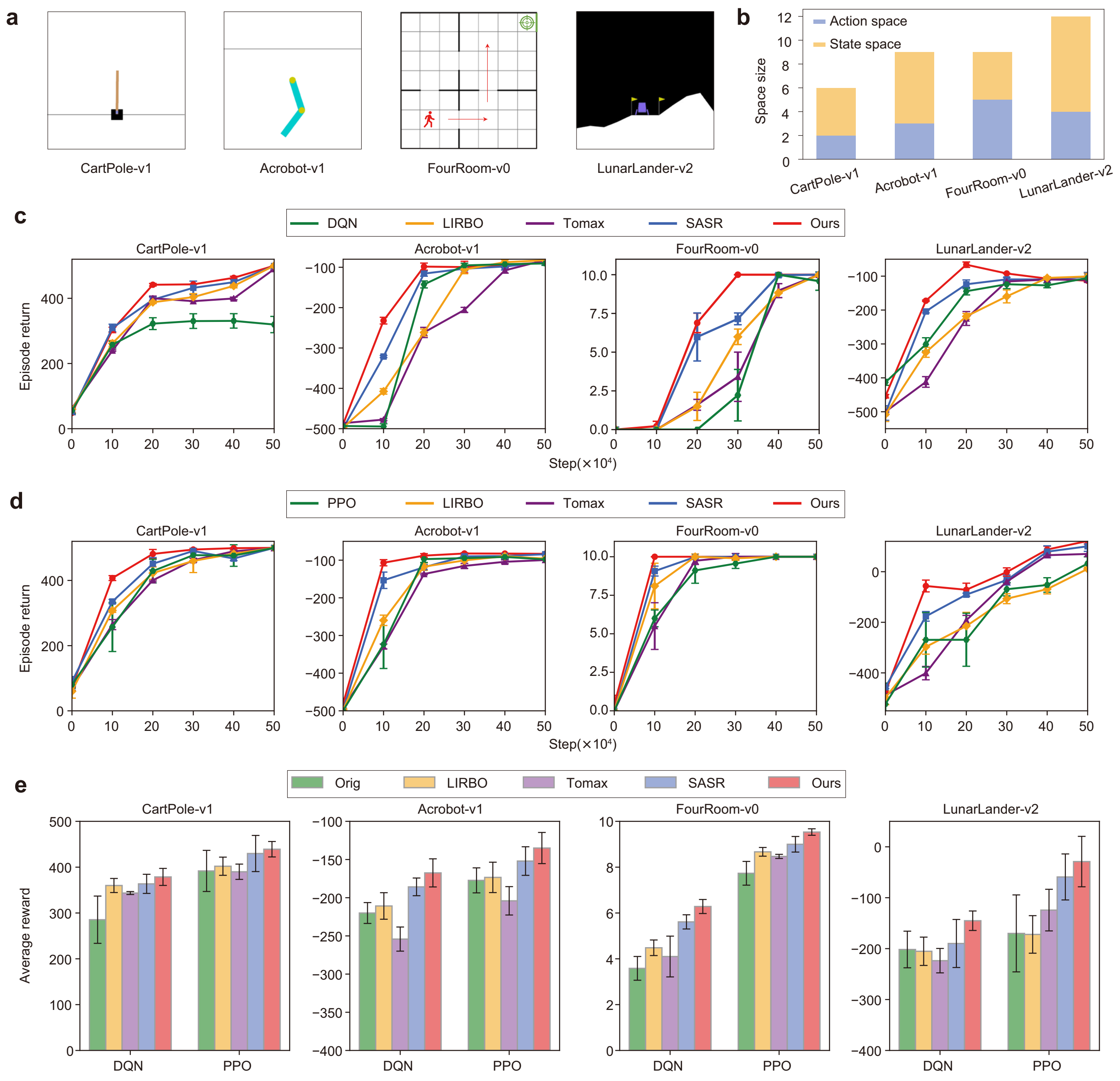

Fig. 3 | Evaluation results for sparse-reward tasks. a The environments used in the sparse-reward task, including CartPole-v1, Acrobot-v1, FourRoom-v0, and LunarLander-v2. **b** The sizes of the observation and action spaces of the four tasks, illustrating the varying complexity levels. **c** The reward curves created during the learning processes of the DQN agent with different reward functions on different tasks. The solid lines show the average rewards, and the error bars indicate the standard deviations across five trials using different random seeds. **d** The reward curves created during the learning processes of PPO agent with different reward functions on different tasks. **e** The average rewards obtained by the embodied RL agent during all episodes, with the standard deviations of the rewards received when different algorithms performed each task.

the neutral region, indicating a learned preference for stabilization or energy conservation when the system is aligned. This structured reward landscape is consistent with the task objective of swinging the outer link above a target height, demonstrating that the reward function of this work successfully captures meaningful control behaviors aligned with the dynamics of the Acrobot-v1 task. Visualizations of the reward functions at different angular velocities can be found in the Supplementary Figs. 2–4.

In summary, attainment of an optimal policy using sparse rewards is challenging because feedback is limited. However, the proposed approach substantially improves learning efficiency by continuously adjusting the reward on the basis of past interactions. Consequently, the approach not only accelerates convergence of the embodied RL agent but also stabilizes the optimized policy because the reward and policy adjustments are synchronous. The results consistently indicate that the proposed approach discovers the optimal reward function and enables agents to overcome the challenges posed by sparse reward signals.

Case study of high-dimensional control tasks

To evaluate the performance of the proposed approach for embodied agents with complex joint structures, four additional complex high-dimensional continuous control tasks are utilized, in which the embodied RL agents have a complex set of joints, rendering it challenging to achieve the goals. The four tasks, Reacher-v2, Hopper-v2, HalfCheetah-v2 and Humanoid-v2, as shown in Fig. 5a, are designed by

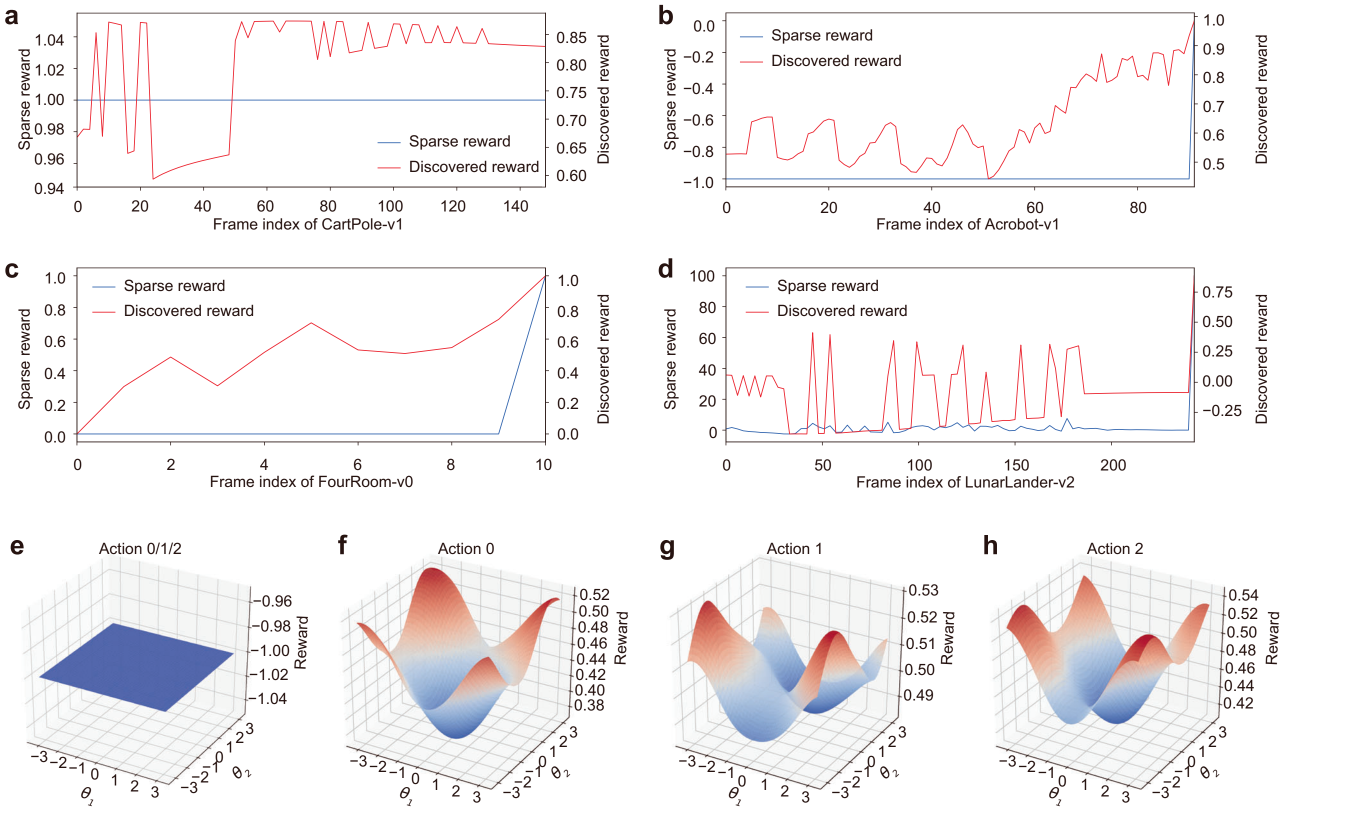

Fig. 4 | Reward visualization. a Reward signals during a single interaction episode of the CartPole-v1 task. The sparse reward signals (blue lines) and the reward signals discovered in this work (red lines) are shown. **b** Reward signals during a single interaction episode of the Acrobot-v1 task, showing the difference between the sparse and discovered reward signals. **c** Reward signals during a single interaction episode of the FourRoom-v0 task, illustrating the increased signal density achieved in this work. **d** Reward signals during a single interaction episode of the LunarLander-v2 task. **e** The surface of the sparse reward function. For any state and action, the sparse reward function always delivers a reward signal of -1. **f** The reward surface for action 0 (-1 torque) of Acrobot-v1 as a function of the joint angles θ_1 and θ_2 with angular velocities fixed at 0. **g** The reward surface for action 1 (0 torque). **h** The reward surface for action 2 (1 torque).

OpenAI using the MuJoCo physics simulator, which is a free and open-source physics engine that is aimed at facilitating research and development in robotics and other areas requiring fast and accurate simulations. The number of joints controlled in the four tasks is shown in Fig. 5b, ranging from a simple two-link robotic arm with only two joints (Reacher-v2) to a humanoid intelligence with 17 joints (Humanoid-v2). For the four tasks, the original reward functions designed by OpenAI and rewards from other methods serve as the baselines. Two agents, including the continuous versions of the soft actor-critic (SAC)[62] and PPO[61], serve as the base RL agents. The agent parameters are not fine-tuned for specific tasks but are instead maintained as general-purpose parameters. Throughout the experiments, all hyperparameters remain consistent. For each task, five trials are executed in parallel with different random seeds. During each trial, the agent interacts with the environment for 10^6 steps, and the reward is upper-level optimized after every 2048 interactions. Additional experimental details can be found in Supplementary Experiment Details.

Figure 5c, d shows the cumulative reward curves of embodied RL agents after each episode across the four tasks, with the error bars indicating the standard deviation across the five parallel experiments. The proposed approach exhibits superior performance, even outperforming the reward functions of OpenAI that are carefully designed by humans. Compared to other baseline methods, our approach is more effective when PPO and SAC agents are employed. The convergence speed is faster and the episodic returns are higher. SASR and LIRBO perform similarly, but SASR achieves better when a PPO agent is applied because SASR samples the success rate of historical data. Figure 5c, d also reveals that the proposed approach accelerates agent convergence and improves the cumulative expected returns, particularly when the number of agent joints increases. Figure 5e presents the means and variance of the cumulative rewards given during learning and compares the rewards of the proposed approach with those afforded by the baseline rewards. It is clear that, for both the SAC and PPO agents, the proposed approach not only outperforms the baseline but also reduces risk induced by randomness.

Figure 6 shows the distributions of the manual reward functions (Fig. 6a) and those discovered via the proposed approach (Fig. 6b) in a state space mapped in two dimensions using the t-SNE method[63], that aids intuitive visualization of the reward distributions. Visualizations of the reward distributions for other reward function design methods are provided in the Supplementary Fig. 5. Details regarding the implementation of t-SNE are also included in Supplementary Experiment Details. The discovered reward function produces rewards that are very close to those of the carefully designed reward function, particularly in terms of the overall distributions of rewards and states. Such similarity can be attributed to extensive sampling of the action space during optimization. Figure 6a shows that, using the carefully designed reward function, only a few specific states are assigned exceptionally low or high rewards, whereas most states receive less extreme rewards. The specific states are those in which the embodied agents experience collisions, exceed boundaries, or fail to complete the desired action within the maximum number of steps. In contrast, the reward function of the proposed method (Fig. 6a) identifies specific states and assigns them lower or higher reward signals. For example, both reward functions assign relatively low rewards to the states in the lower-right corner of the leftmost figure (Reacher-v2) of Fig. 6. In these states, the

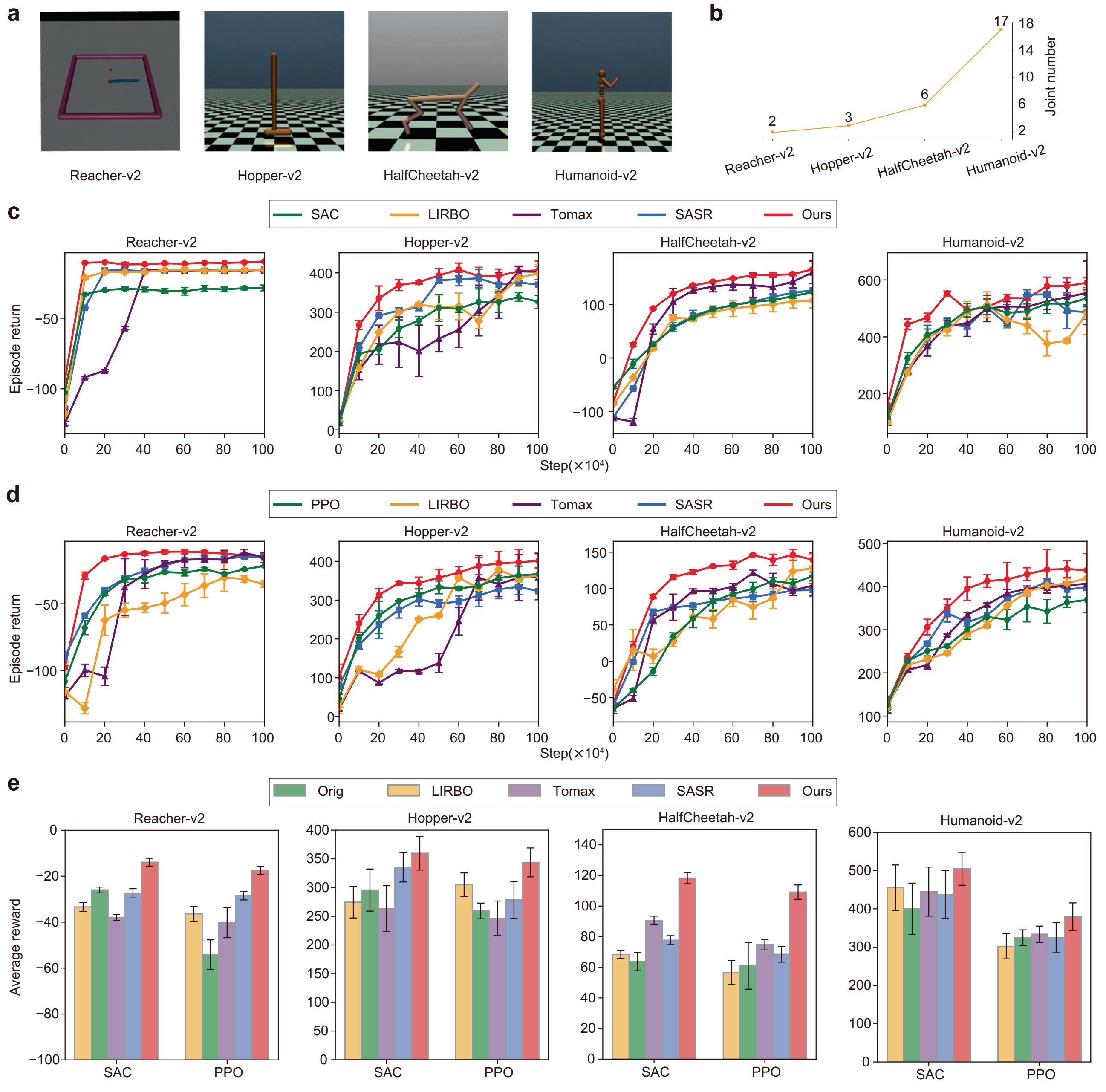

Fig. 5 | Evaluation results for high-dimensional control tasks. a The high-dimensional control tasks, including Reacher-v2, HalfCheetah-v2, Hopper-v2 and Humanoid-v2. **b** The number of controllable joints increases from Reacher-v2 to Humanoid-v2, reflecting the increasing complexity of the tasks. **c** The reward curves during the learning processes of the SAC agent across the four tasks. The solid lines show the average episode rewards and the error bars indicate the standard deviations over five trials with different random seeds. **d** The reward curves during the learning processes of the PPO agent across the four tasks. **e** The average rewards with their standard deviations for different RL agents and different tasks. The results highlight the improved performance of the proposed approach across all tasks.

robotic arm fails to complete the task. This phenomenon occurs because the objective function is set before the reward function is optimized, as the reward optimization objective is explicitly tied to the agent's ultimate goal in this work. Furthermore, the discovered reward function allocates more reasonable reward signals across different states, reflected in Fig. 6b as a more concentrated distribution of similar reward values. In contrast, the manual reward function in Fig. 6a gives anomalous reward distributions because of inevitable (human) imperfections. The findings suggest that the discovered reward function not only captures the intrinsic mechanism underlying the reward relationships but also learns the optimal mapping between states and rewards. This implies that the design of reward functions for embodied RL agents with complex joints will no longer constitute a major barrier when training such agents. Using the proposed approach, even when the intricate operational mechanisms of an embodied agent are not clearly understood, a well-optimized reward function can help the agent to exhibit intelligent behavior.

Case study of real-world tasks

To demonstrate the real-world applicability of the proposed approach, real-world assessment scenarios in which RL has been widely applied but still faces significant challenges are evaluated, including autonomous energy management of data centers and unmanned systems control of UAVs.

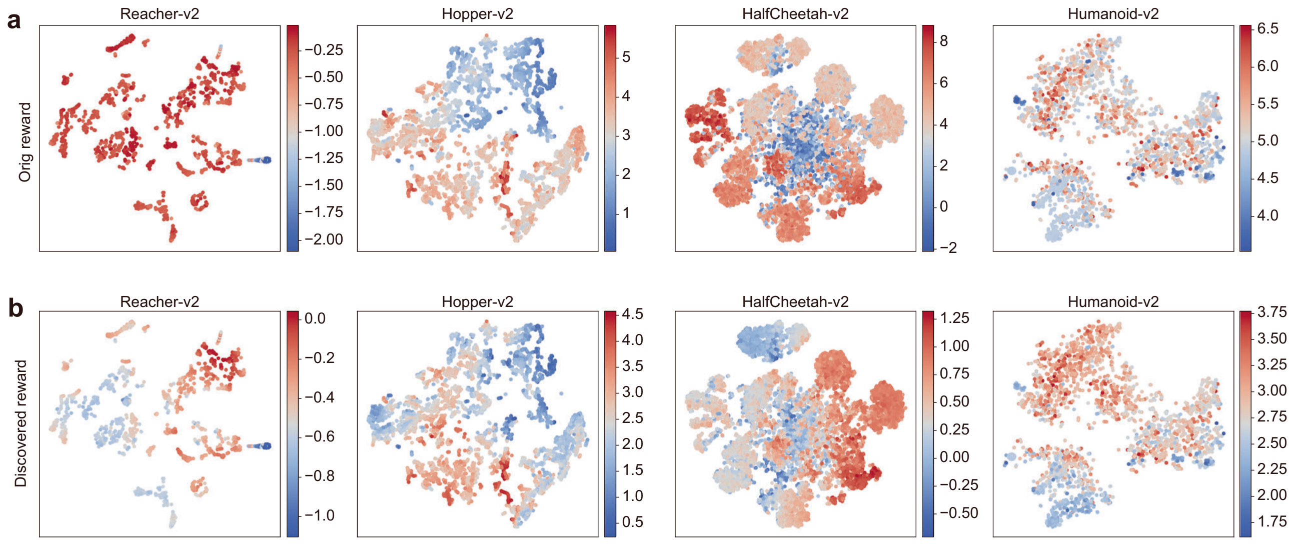

Fig. 6 | Reward distributions in the state space. a The state--reward distributions, wherein each point represents a state projected onto a two-dimensional space via the t-SNE method. The color of each point indicates the reward, with red denoting higher rewards and blue representing lower rewards. **b** The reward distribution of the discovered reward function is better than that of the original reward function because the method assigns extreme rewards to critical states and gives more reasonable rewards for most states. This demonstrates the advantages of the proposed approach in guiding agents to explore more effectively the state space.

Case study of energy management task. Reducing the energy consumption of data centers has garnered increasing attention. Many researchers seek to leverage RL methods to automate energy management and reduce the energy and economic losses. However, it is difficult to balance the dynamic environment of modern data centers because of the complex nonlinear interactions among the equipment and the way in which humans operate the equipment. Furthermore, each data center has a unique architecture and environment, meaning that any RL agent designed specifically for that center may not function well in other centers. Traditional reward functions based on the difference between the data center temperature and a standard temperature often fail to capture the interactions. A universal framework for automatic design of reward functions is needed. In this section, the performance of the proposed approach in terms of assisting RL to reduce energy consumption in data centers is evaluated. An embodied RL agent is trained using real-world data to adjust the cooling tower activity by reference to the data center temperature. The trained agent is then tested over a complete operational cycle (typically 1 year).

During training, the embodied RL agent is trained over 200 rounds. In each round, the agent learns for 30 days, and makes a decision every minute. The cooling tower uses these decisions to move the data center temperature to the appropriate range, thereby consuming energy, as shown in Fig. 7a. Certain variables affecting the temperature, including the ambient temperature, user count, and data rate, are randomly processed in each round. If the data center temperature exceeds the acceptable range, learning is terminated, and the agent is considered to have failed. The trained agent is then tested over a one-year period. The agent adjusts the data center settings at one-minute intervals. To fully demonstrate the performance of the proposed approach in assisting various types of RL, three different RL agents (DQN[60], SAC[62], and PPO[61]) are tested and the energy consumption levels compared to that of a data center without RL-based regulation.

Figure 7b shows the rate of reduction in energy consumption when applying baseline RL and RL guided by the reward design approaches in data centers over a 1-year period. The numerical values are the percentage reduction rates. Figure 7c illustrates the total energy consumptions over 3, 6, 9, and 12 months for each RL agent. Although all RL methods reduce energy consumption, the reward function discovery approach further reduces consumption, particularly over longer times. Notably, as RL algorithms become more advanced, from the DQN to the more sophisticated SAC, the energy reduction rate increases (from 21% for DQN to 52% for SAC). The superiority of the proposed approach is more obvious when compared with the baseline models. With our method, all RL agents (including DQN, PPO, and SAC) achieve energy reductions that exceed 60%. These results highlight the generality and effectiveness of our approach for RL agents involved in energy management and other tasks. Therefore, the proposed optimal reward function discovery approach can be applied to RL agents operating in data centers and elsewhere. Potential applications include improvements in the conversion efficiencies of power plants (extraction of more energy from the same), reductions in energy and water consumption during semiconductor manufacturing, and improved throughput of manufacturing facilities.

Case Study of Unmanned Systems Control Task. In recent years, UAVs have been increasingly deployed in applications such as tracking, capturing, and disaster relief. However, flexible deployment requires precise environmental modeling, which is difficult in most real-world scenarios. RL has emerged as an effective solution to this problem, as in other areas of robotics. However, when controlling UAVs, the reward function requires iterative adjustments, although a reasonably well-defined reward function can be derived via dynamic modeling. In this section, the effectiveness of the discovered reward function is evaluated by applying it to an autonomous UAV quadrotor engaged in a tracking task. The experiments are based on Pybullet, and the UAV model is implemented via ArduPilot and PX4, as shown in Fig. 8a. The UAV must autonomously track a series of trajectory target points specified by the environment, and UAV observations are limited only to the two nearest targets at any given time. Three state-of-the-art RL algorithms, PPO[61], SAC[62], and twin delayed deep deterministic policy gradient (TD3)[64], are chosen as baseline agents for comparisons. No task-specific adjustments are made. Each RL agent interacts with the environment over 10^6 time steps. Multiple randomized seeds are used to parallelize the experiments. Additional experimental details are provided in Supplementary Experiment Details.

Figure 8b shows the reward curves when agent learning is guided by the reward function of the present work, reward function constructed on the basis of dynamic principles, and other reward design

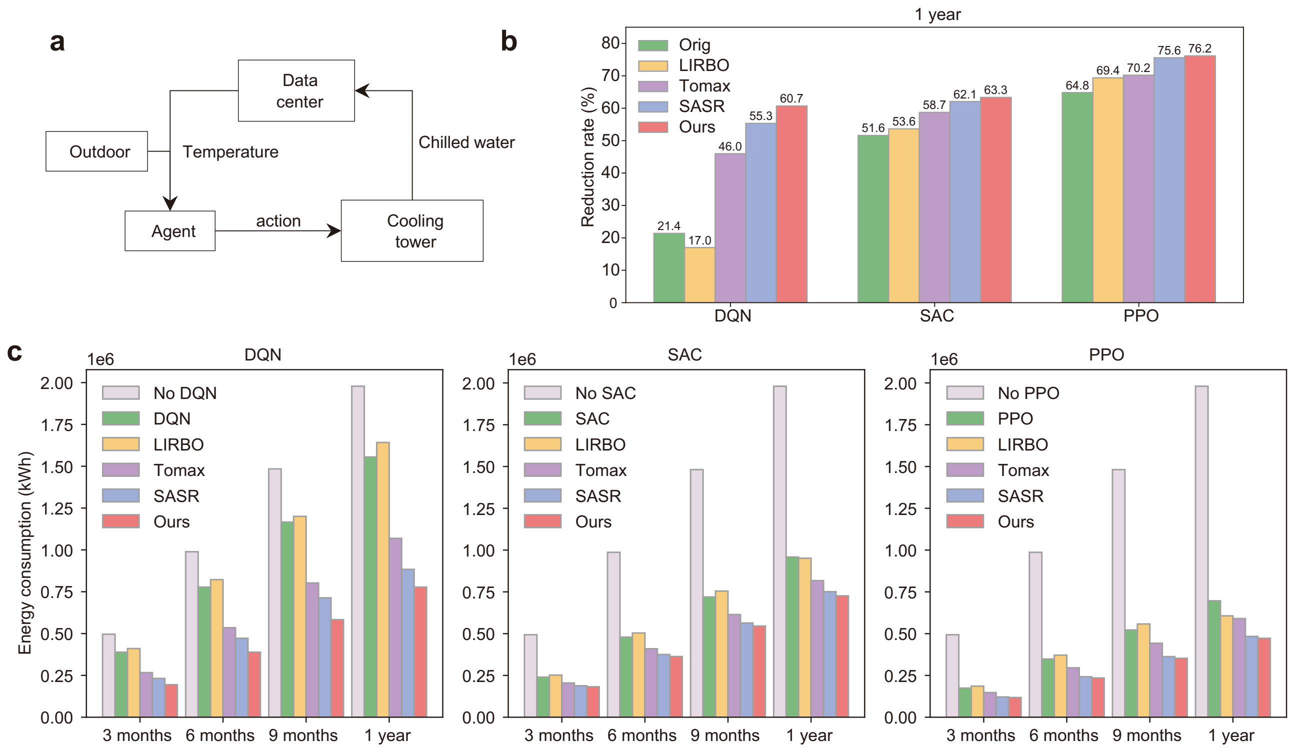

Fig. 7 | Evaluation results for the energy management task. a Schematic of the data center cooling task, where an RL agent optimizes cooling system based on temperature and silicon data. **b** The 1-year reductions in energy consumption afforded by different reward design methods for different RL algorithms (DQN, SAC, and PPO). The agents that receive rewards from the discovered reward functions perform significantly better than others. **c** Energy consumption over 3, 6, 9, and 12 months. The RL approach using the discovered optimal reward function reduce energy consumption more effectively than do the comparative methods.

methods. Error bars represent standard deviations. Across the three different RL agents, the proposed approach significantly accelerates learning. For the PPO agent, our approach significantly improves both learning speed and the final performance, enabling the agent to successfully accomplish the task, whereas the hand-designed and other comparison methods fail to do so. For the SAC and TD3 agents, the improvement in learning speed is less pronounced, but our method performs comparably and ensures stable learning. These results demonstrate that the proposed reward function discovery approach is particularly useful when traditional rewards are insufficient, consistently ensuring robust performance across different RL algorithms. Figure 8c shows the numbers of target trajectory waypoints successfully attained by the UAV during the interactions, supporting the same conclusion.

The differences between the discovered reward function and the carefully designed reward function based on dynamic principles are illustrated in Fig. 8d, e. For the manually designed reward, the UAV receives a reward of 100 upon reaching the target position and -100 upon crashing, while rewards for all other states remain relatively small. Such a design leads to a very sparse reward distribution, as shown on the left side of Fig. 8d. When further analyzing the reward distributions, we exclude those of special states; the resulting filtered state space is that of Fig. 8e. These visualizations clearly demonstrate the advantages of our approach. First, our method fully captures the critical states when a UAV attains a target trajectory point or crashes. Specifically, in Fig. 8d, the lower red box refers to states wherein the UAV successfully attains a target point, associated with a strong positive reward, and the upper blue box highlights crash states associated with negative rewards. These critical states are appropriately highlighted in the reward space. More importantly, our approach significantly reduces reward sparsity by delivering denser and more informative reward signals throughout the state space, rather than concentrating the rewards on only a few special states. Furthermore, our discovered reward function effectively encodes the underlying theoretical relationship between states and rewards. This is evidenced by the fact that states with similar dynamics are assigned reward values that closely match those calculated when using the physical principles of the system. Our method not only preserves the essential characteristics of the task but also facilitates more efficient learning by providing richer feedback to the agent. Visualizations of the reward distributions for other reward function design methods are provided in Supplementary Fig. 6 for further comparison.

The experimental results of data center energy management and unmanned systems autonomous control indicate that the proposed framework effectively addresses existing RL challenges, such as adaptation to dynamically changing environments. Moreover, the distributions of the discovered reward function align with those of reward functions derived via dynamical modeling, particularly in terms of the emphasis on critical states and the overall distribution of rewards in the state space. The experimental findings support the suggestion that the proposed framework can be applied to real-world applications in embodied RL systems.

Discussion

The resolution of the general control problems of EAI systems requires the development of universal policies that are embodied by RL agents that learn via trial-and-error, then progressively approach or indeed surpass human decision-making. Traditionally, researchers who seek to enable embodied RL agents to perform effectively in open-world tasks have to extensively analyze the characteristics of the world model, to ensure the reliability of the designed reward functions. Such a case-specific approach yields only a fragmented understanding of the problem, as in the parable of the blind man and the elephant, requiring substantial human effort while frequently failing to yield an agent that meets expectations. Inspired by the principle of reward

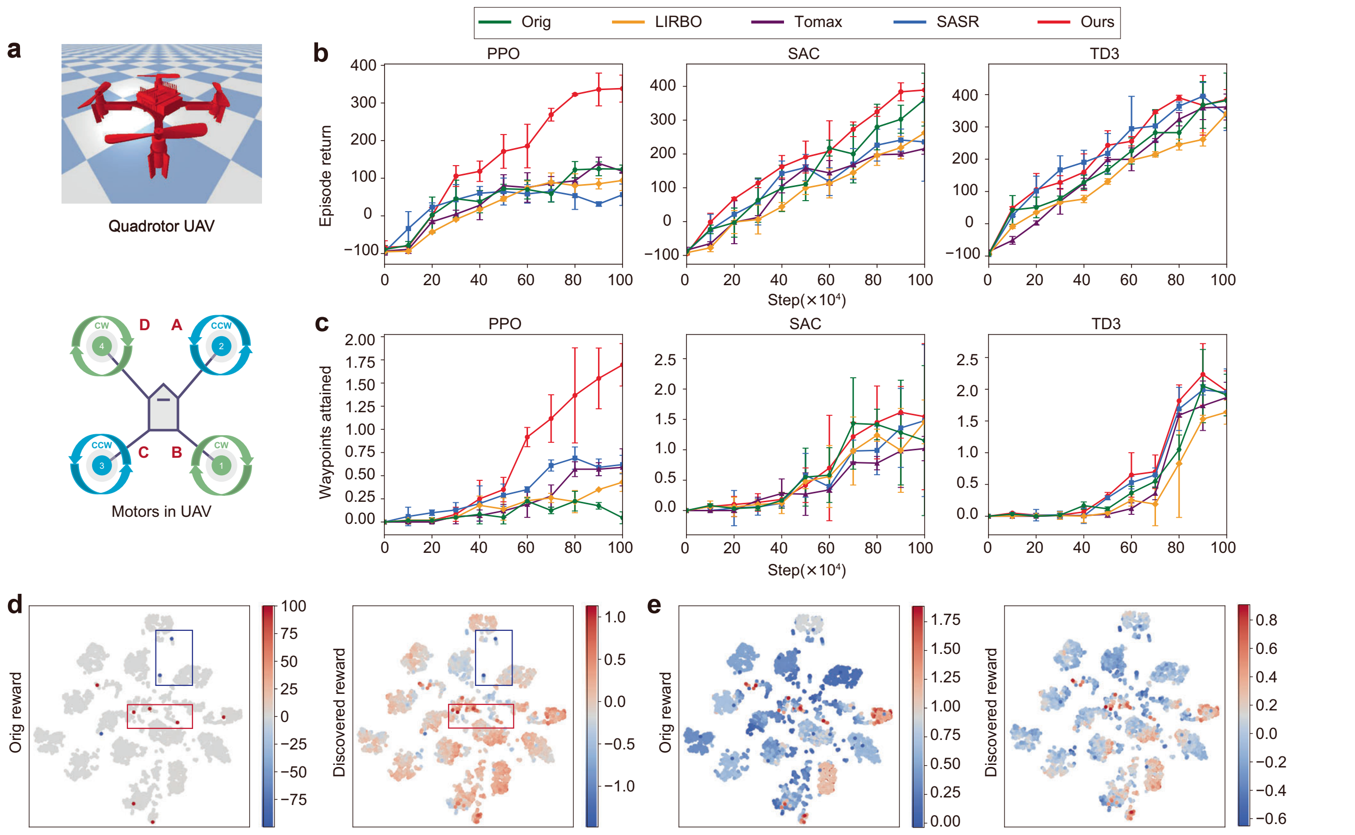

Fig. 8 | Evaluation results for unmanned systems control task. a The Quadrotor UAV (adapted from PyFlyt) and the powered motors in a world model, reproduced from ArduPilot. **b** Learning curves showing reward progression during UAV trajectory tracking using the PPO, SAC, and TD3 algorithms. The solid lines show the average rewards, and the error bars indicate the standard deviations using different random seeds. The proposed approach is associated with faster convergence and better performance. **c** The numbers of target trajectory points attained by the UAV during learning. The solid lines show the average rewards, and the error bars indicate the standard deviations using different random seeds. **d** The state space visualizations of the UAV task, showing the reward distributions of the original reward function derived using dynamic principles and the discovered reward function. The discovered reward function successfully captures critical states. **e** The state space visualizations after removal of certain states associated with extremely high or low rewards, such as those involving collisions or attainment of waypoints. The reward distribution resembles those of reward functions derived using dynamic principles.

maximization observed in nature and cognitive neuroscience, this work introduces a framework for the automatic discovery of optimal reward functions for embodied RL agents. This breakthrough demonstrates that overcoming trial-and-error with serendipitous learning remains an essential paradigm in disruptive innovation. By leveraging a predefined definition of the optimal reward function, the framework iteratively discovers the optimal reward function as the agent learns its policy. The proposed approach empirically affords superior performance in real-world tasks, marking a crucial waypoint toward the realization of general EAI systems. The experiments are carefully organized into distinct tasks that rigorously test and explore how the proposed approach discovers the optimal reward function. The reward function effectively guides embodied RL agents in solving tasks. An analysis of reward distributions in the state space reveals that the framework both determines the latent relationships between rewards and agent states and captures distinctive states (e.g., task success and failure) within the world model. This highlights the potential of this method to develop generalized embodied agents for open-world control tasks.

In summary, the proposed framework (details given in Fig. 2d) facilitates automatic discovery of the optimal reward functions for embodied RL agents, thereby avoiding complex manual construction of reward functions. By leveraging this framework, embodied RL agents efficiently develop general control policies tailored to a diverse range of real-world tasks, supporting the practical realization of EAI. In March 2025, ACM announced that Andrew G. Barto and Richard S. Sutton were recipients of the 2024 ACM A.M. Turing Award because they developed the conceptual and algorithmic foundations of RL. We believe that this work is a significant step toward a direct pathway facilitating the understanding and construction of AGI, thereby improving the overall intelligence of embodied RL agents and offering deeper insights into interpretable EAI systems.

Methods

Preliminaries

Notations. Throughout the article, the following symbols are used. $\mathcal{S}$: state space. $\mathcal{A}$: action space. π: policy. θ: parameters of π. R: reward function. ω: parameters of R. Π: policy space. $\mathcal{R}$: reward function space. π^*: optimal policy. R^*: optimal reward function. π^*_R: learned optimal policy under reward function R. $\pi^*_{R^*}$: learned optimal policy under optimal reward function R^*, and $\pi^*_{R^*} = \pi^*$. V^π_R: state value of policy π under reward function R. Q^π_R: state-action value of policy π under reward function R. A^π_R: state-action advantage of policy π under reward function R. G^π_R: discounted expected return of policy π under reward function R. ξ: trajectory, contains $(s^{(0)}, a^{(0)}, r^{(0)}, \ldots, s^{(t)}, a^{(t)}, r^{(t)}, \ldots)$, where $r^{(t)} = R(s^{(t)}, a^{(t)})$. δ: a small positive constant. $|*|$: absolute value of a real number.

The general RL problem without a designed reward function is the world model denoted as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \rho_0, T, \gamma)$. The initial state distribution is ρ_0, which is the probability distribution over the initial states. The state transfer matrix is $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, and $T(s, a, s')$

specifies the transition probability from state s to s' when the embodied RL agent chooses action a. The discount factor $\gamma \in [0, 1)$ determines the relative importance of immediate reward signals compared to future reward signals. We denote the world model $\mathcal{M}$ with a designed reward function R as $\mathcal{M}[R]$, which is a Markov decision process (MDP). In MDP, an agent observes state s, chooses action a based on policy π, receives rewards signal r from reward function R, and optimizes its policy π to π_R^* by maximizing G_R^{π}:

$$\pi_R^* \in \arg\max_{\pi \in \Pi} G_R^{\pi} = \mathbb{E}_{\xi \sim \pi}\left[\sum_{t=0}^{\infty} \gamma^t \cdot R(s^{(t)}, a^{(t)})\right], \tag{1}$$

where $\xi \sim \pi$ indicates that the trajectory ξ is derived by an agent obey policy π and interact with world model. Here, we assume that when the optimal reward function R^* is used to guide the agent's policy learning, we have $\pi_{R^*}^* = \pi^*$.

The optimal reward function problem

Before formally providing the definition of the optimal reward function, we first define the *proximal* between two reward functions and policies.

Definition 1. (Proximal). For two policies π_1 and π_2, if the absolute difference between $G_R^{\pi_1}$ and $G_R^{\pi_2}$ is bounded by a small positive constant, i.e., $|G_R^{\pi_1} - G_R^{\pi_2}| \le \delta$, then π_1 and π_2 are proximal with R, denoted as $\pi_1 \sim R\pi_2$. For two reward functions R_1 and R_2, if the absolute difference between $G_{R_1}^{\pi}$ and $G_{R_2}^{\pi}$ is bounded by a small positive constant for all policies $\pi \in \Pi$, i.e., $|G_{R_1}^{\pi} - G_{R_2}^{\pi}| \le \delta, \forall \pi \in \Pi$, then R_1 and R_2 are proximal, denoted as $R_1 \sim R_2$.

Proposition 1. Given a reward function R, if R is proximal to the optimal reward function R^*, then policy π_R^* will be proximal to the optimal policy π^* with R and R^*.

Proof. See Supplementary Theoretical Results. □

Definition 1 shows that when the absolute difference between G_R^{π} and $G_{R^*}^{\pi}$ is bounded by a small constant for any policy π, the reward function R qualifies as an approximately optimal reward function. However, in many practical applications, we are only interested in the performance of the learned optimal policy, i.e., π_R^*, rather than the entire policy space. Therefore, we relax the optimal condition based on Proposition 1 and only require the expected returns of the policies π_R^* and π^* are approximately equal under R^*. That means, the optimal reward function R^* can be obtained through minimizing the absolute difference between the $G_{R^*}^{\pi_R^*}$ and $G_{R^*}^{\pi^*}$, which is termed as the regret $\mathfrak{L}$ in this work:

$$\mathfrak{L}(\pi_R^*, \pi^* | R^*) = \left|G_{R^*}^{\pi^*} - G_{R^*}^{\pi_R^*}\right|. \tag{2}$$

Definition 2. A reward function R is defined as the optimal reward function if it minimizes the expected regret $\mathfrak{L}(\pi_R^*, \pi^* | R^*)$ over the possible reward space $\mathcal{R}$, where the policy π_R^* is learned by an embodied RL agent interacting with the world model $\mathcal{M}[R]$. Formally, it is expressed as:

$$R^* = \arg\min_{R \in \mathcal{R}} \mathbb{E}_{\pi_R^* \sim \mathcal{M}[R]}\left[\mathfrak{L}(\pi_R^*, \pi^* | R^*)\right], \tag{3}$$

where $\mathcal{R}$ denotes the space of all possible reward functions and $\pi_R^* \sim \mathcal{M}[R]$ indicates that the policy π_R^* is derived under the dynamics influenced by the reward function R.

The optimal reward function problem (ORFP) is defined in Definition 2. However, given an MDP $\mathcal{M}[R]$, a direct search for the optimal reward function via equation (3) remains challenging for the following two reasons:

(1) In practice, the target optimal reward function R^* is unknown, which makes it not possible to directly evaluate and compare regrets of different candidate reward functions.
(2) Determination of the optimal policy π^* for a given reward function R is not straightforward, rendering it challenging to compute the regret $\mathfrak{L}(\pi_R^*, \pi^* | R^*)$ via equation (2).

To address the first challenge, an additional reward function $\bar{R}$ is introduced to provide a unified evaluation of regret. The reward function $\bar{R}$ can be simply defined as both fixed and sparse. In this work, the function is associated with the ultimate goal of the embodied RL agent and gives a reward when the task is completed. We then measure the regret of different reward functions under the reward function $\bar{R}$:

$$\mathfrak{L}(\pi_R^*, \pi^* | \bar{R}) = \left|G_{\bar{R}}^{\pi^*} - G_{\bar{R}}^{\pi_R^*}\right|. \tag{4}$$

Although many researchers have sought to approximate regret[65], direct regret computation and minimization using equation (2) remain challenging. However, noting that the optimization objective of the optimal policy π^* under the reward function $\bar{R}$ is deterministic as $\bar{R}$ is fixed[66], we simplify the objective function in equation (3) and reformulate the process used to discover the optimal reward function from the minimization of regret to maximization of the objective value $G_{\bar{R}}^{\pi_R^*}$ by omitting $G_{\bar{R}}^{\pi^*}$:

$$\begin{aligned} R^* &= \arg\min_{R \in \mathcal{R}} \mathbb{E}_{\pi_R^* \sim \mathcal{M}[R]}\left|G_{\bar{R}}^{\pi^*} - G_{\bar{R}}^{\pi_R^*}\right|, \\ &= \arg\max_{R \in \mathcal{R}} \mathbb{E}_{\pi_R^* \sim \mathcal{M}[R]}\left[G_{\bar{R}}^{\pi_R^*}\right]. \end{aligned} \tag{5}$$

Intuitively, the definition of the optimal reward function in equation (5) holds for most RL problems.

The bilevel optimization framework

We next naturally integrate the process used to discover the optimal reward function of equation (5) into the RL process and define that process within a bilevel optimization framework. As ω and θ represent the parameters of reward function and policy, the bilevel optimization framework is formulated as follows:

$$\omega^* = \arg\max_{\omega \in \Omega} \mathcal{J}^{\text{upper}}(\theta, \omega) \triangleq \mathbb{E}_{\pi_{\theta^*} \sim \mathcal{M}[R_\omega]}\left[G_{\bar{R}}^{\pi_{\theta^*}}\right], \tag{6}$$

$$\theta^* = \arg\max_{\theta \in \Theta} \mathcal{J}^{\text{lower}}(\theta, \omega) \triangleq \mathbb{E}_{s \sim \rho^{\pi_\theta}, a \sim \pi_\theta(\cdot|s)}\left[\sum_{t=0}^{\infty} \gamma^t \cdot R_\omega(s^{(t)}, a^{(t)})\right], \tag{7}$$

where Ω and Θ denote the spaces of the possible reward function parameters and the policy parameters, respectively. $s \sim \rho^{\pi_\theta}$ denotes that the state s is sampled from the distribution ρ^{π_θ} and $a \sim \pi_\theta(\cdot|s)$ indicates that action a is sampled from policy $\pi_\theta(\cdot|s)$ given state s. To avoid notation bloat, we use π_{θ^*} below to denote the optimal policy π_R^* of reward function R even though R does not appear.

The solutions of the equations ((6)–(7)) can be sought using a straightforward, alternating optimization method that iteratively fixes either ω or θ and optimizes the other. Specifically, in our framework, given the reward function R_ω and policy π_θ, we begin by fixing the reward function R_ω and then optimizing the policy from π_θ to $\pi_{\theta'}$ according to equation (7). This step is a standard process whereby RL algorithms can be applied to solve lower-level problem. Subsequently, we fix the policy and optimize the reward function R_ω to $R_{\omega'}$ using equation (6). This alternating process continues until the optimal policy and optimal reward are discovered.

The lower-level problem. Given that the reward function R_ω is fixed, the lower-level problem is a standard RL policy optimization problem. In this problem, the embodied RL agent receives reward signals from the upper-level reward function R_ω and then interacts with the world model $\mathcal{M}$. Thus, the expected cumulative return $\mathcal{J}^{\text{lower}}(\theta,\omega)$ of the lower-level problem is defined by equation (7), and we rewrite it here:

$$\mathcal{J}^{\text{lower}}(\theta,\omega) \triangleq \mathbb{E}_{s\sim\rho^{\pi_\theta},a\sim\pi_\theta(\cdot|s)}\left[\sum_{t=0}^{\infty}\gamma^t\cdot R_\omega(s^{(t)},a^{(t)})\right]. \tag{8}$$

Lemma 1. (ref. 67). Given a standard RL problem denoted by a world model $\mathcal{M}[R]=(\mathcal{S},\mathcal{A},\rho_0,T,\gamma,R_\omega)$ with the objective function defined in equation (8), the policy gradient can be defined as follows:

$$\nabla_\theta\mathcal{J}^{\text{lower}}(\theta,\omega)=\mathbb{E}_{\mu_{s,a}}\left[\nabla_\theta\log\pi_\theta(s,a)\cdot Q^{\pi_\theta}_{R_\omega}(s,a)\right], \tag{9}$$

where $Q^{\pi_\theta}_R$ is the state–action value function of π_θ in the world model $\mathcal{M}[R]$ and $\mu_{s,a}$ is shorthand for $s\sim\rho^{\pi_\theta},a\sim\pi_\theta(\cdot|s)$.

We directly derive the gradient update form for the policy parameter θ of the lower-level optimization step based on Lemma 1:

$$\begin{aligned}\theta' &= \theta+\alpha\cdot[\nabla_\theta\mathcal{J}^{\text{lower}}(\theta,\omega)]_\theta\\ &=\theta+\alpha\cdot\mathbb{E}_{\mu_{s,a}}\left[[\nabla_\theta\log\pi_\theta(s,a)]_\theta\cdot Q^{\pi_\theta}_{R_\omega}(s,a)\right],\end{aligned} \tag{10}$$

where α denotes the learning rate during lower-level policy optimization.

The upper-level problem. The upper-level objective of equation (6) is defined by reference to equation (5), which we here restate:

$$\mathcal{J}^{\text{upper}}(\theta,\omega)\triangleq\mathbb{E}_{\pi_{\theta^*}\sim\mathcal{M}[R_\omega]}\left[G^{\pi_{\theta^*}}_R\right]. \tag{11}$$

As for lower-level optimization, we optimize the reward parameter ω via gradient updates. For a fixed θ' and ω, we update the reward parameter as follows:

$$\omega'=\omega+\beta\cdot\left[\nabla_\omega\mathcal{J}^{\text{upper}}(\theta,\omega)\right]_{\omega}, \tag{12}$$

where β is the learning rate of the upper-level problem.

As the policy parameter θ is influenced by the reward parameter ω given during RL, we model the policy parameters as a meta function of the reward parameters, i.e., $\theta(\omega)$. After constructing this meta function, we establish a relationship between the reward parameters ω and the policy parameters θ, in turn enabling the pseudo-updating of the reward function parameters ω in the upper-level optimization step, as follows:

$$\left[\nabla_\omega\mathcal{J}^{\text{upper}}\right]_\omega=\left[\nabla_\theta\mathcal{J}^{\text{upper}}\right]_{\theta'}\cdot\left[\nabla_\omega\theta(\omega)\right]_\omega. \tag{13}$$

Assumption 1. Assume that the policy distribution is continuous and differentiable with respect to the policy parameters.

On the basis of Assumption 1, we prove Theorem 1 that considers optimization of the reward function parameter.

Theorem 1. Given that the bilevel optimization problem in equations ((6)–(7)), the gradient of the upper-level objective $\mathcal{J}^{\text{upper}}$ with respect to the reward parameters ω is:

$$\left[\nabla_\omega\mathcal{J}^{\text{upper}}\right]_\omega=\mathbb{E}_{\mu'_{s,a}}\left[Q^{\pi_\theta}_R\cdot\left[\nabla_\theta\log\pi_\theta\right]_{\theta'}\right]\cdot\mathbb{E}_{\mu_{s,a}}\left[\left[\nabla_\omega Q^{\pi_\theta}_{R_\omega}\right]_\omega\cdot\left[\nabla_\theta\log\pi_\theta\right]_\theta\right], \tag{14}$$

where $\mu_{s,a}$ and $\mu'_{s,a}$ are the policy distributions of the previous and current steps respectively.

Proof. See Supplementary Theoretical Results. □

On the basis of Theorem 1, in the upper-level of the optimization framework, the reward function parameters can be updated as follows:

$$\begin{aligned}\omega'&=\omega+\beta\cdot\left[\nabla_\omega\mathcal{J}^{\text{upper}}\right]_\omega\\ &=\omega+\beta\cdot\mathbb{E}_{\mu'_{s,a}}\left[Q^{\pi_{\theta'}}_R\cdot\left[\nabla_\theta\log\pi_\theta\right]_{\theta'}\right]\cdot\mathbb{E}_{\mu_{s,a}}\left[\left[\nabla_\omega Q^{\pi_\theta}_{R_\omega}\right]_\omega\cdot\left[\nabla_\theta\log\pi_\theta\right]_\theta\right].\end{aligned} \tag{15}$$

However, it is difficult to obtain the policy gradient for value-based RL such as that of the deep Q-network (DQN)[9], and for both value-based and policy-based RL such as the soft actor–critic (SAC)[62]. Hence, we further develop Theorem 1 and extend the applicability thereof.

Theorem 2. Given the bilevel optimization problem in equations ((6)–(7)), the gradient of the upper-level objective $\mathcal{J}^{\text{upper}}$ with respect to the reward parameters ω in Theorem 1 can be formulated as:

$$\left[\nabla_\omega\mathcal{J}^{\text{upper}}\right]_\omega=\mathbb{E}_{\mu_{s,a}}\left[\nabla_\omega\left[\mathbb{E}_{\mu'_{s,a}}A^{\pi_{\theta'}}_R(s,a)\cdot A^{\pi_\theta}_{R_\omega}(s,a)\right]_\omega\right], \tag{16}$$

where $A^{\pi_\theta}_R(s,a)$ is the advantage function of policy π_θ under the world model $\mathcal{M}[R]$.

Proof. See Supplementary Theoretical Results. □

On the basis of Theorem 2, the optimization term in equation (15) can be formulated as:

$$\omega'=\omega+\beta\cdot\mathbb{E}_{\mu_{s,a}}\left[\nabla_\omega\left[\mathbb{E}_{\mu'_{s,a}}A^{\pi_{\theta'}}_R(s,a)\cdot A^{\pi_\theta}_{R_\omega}(s,a)\right]_\omega\right], \tag{17}$$

where the advantage function $A^{\pi_\theta}_R(s,a)$ is defined as the value advantage of the agent that selects action a in state s, expressed as:

$$A^{\pi_\theta}_R(s,a)=Q^{\pi_\theta}_R(s,a)-V^{\pi_\theta}_R(s), \tag{18}$$

where $Q^{\pi_\theta}_R(s,a)$ represents the expected return of taking action a in state s and $V^{\pi_\theta}_R(s)$ denotes the expected return of following policy π_θ in state s.

However, we cannot obtain the distribution $\mu'_{s,a}$ when optimizing the reward parameters from ω to ω', even if the policy has been optimized to $\pi_{\theta'}$. The reason is that during RL, we simulate the distribution by allowing the agent to interact with the world model and then store the trajectories. To address this limitation, we approximate the expectations of $\pi_{\theta'}$ in equation (17) using samples from π_θ as suggested by Lemma 2.

Assumption 2. Assume that the support of policy π covers the support of policy π', meaning that for any state-action pair (s, a) for which $\pi'(s,a)>0$, we also have $\pi(s, a) > 0$.

Assumption 2 guarantees that the importance sampling ratio $\frac{\pi_{\theta'}}{\pi_\theta}$ of Lemma 2 is always well defined, finite, and computationally stable during optimization.

Lemma 2. (ref. 68). Given two policies π and π', the expectation of a function $f(s, a)$ under π' can be estimated using samples from π and the importance sampling ratio. Specifically,

$$\mathbb{E}_{\pi'}[f(s,a)]=\mathbb{E}_\pi\left[\frac{\pi'(s,a)}{\pi(s,a)}f(s,a)\right], \tag{19}$$

where $\frac{\pi'(s,a)}{\pi(s,a)}$ is the importance sampling weight, assuming that $\pi(s, a) > 0$ for all actions a taken under π'.

Use of the importance sampling weight ensures consistent reweighting of samples from π to π', enabling unbiased gradient estimation even with different policy distributions. Therefore, on the basis of Lemma 2, we approximate the gradient term in equation (17) as:

$$\begin{aligned}\omega' &= \omega + \beta \cdot \left[\nabla_\omega \mathcal{J}^{\text{upper}}\right]_\omega \\ &= \omega + \beta \cdot \mathbb{E}_{\mu_{s,a}}\left[\nabla_\omega \left[\mathbb{E}_{\mu_{s,a}}\left[\frac{\pi_{\theta'}}{\pi_\theta} \cdot A^{\pi_\theta}_{\bar{R}}(s,a)\right] \cdot A^{\pi_\theta}_{R_\omega}(s,a)\right]_\omega\right].\end{aligned} \tag{20}$$

Assumption 3. Assume that the policy and reward lying between two consecutive optimization steps are stationary.

In practical implementations, policy updates can be reasonably as stable, implying that the policy distributions before and after the update remain near-identical. Consequently, Assumption 3 further simplifies equation (20). On the basis of Assumptions 2–3 and Lemma 2, in the upper-level of the optimization framework, the reward function parameter update method of equation (17) can be approximated as follows:

$$\begin{aligned}\omega' &= \omega + \beta \cdot \left[\nabla_\omega \mathcal{J}^{\text{upper}}\right]_\omega \\ &\approx \omega + \beta \cdot \mathbb{E}_{\mu_{s,a}}\left[\nabla_\omega \left[\mathbb{E}_{\mu_{s,a}}\left[A^{\pi_\theta}_{\bar{R}}(s,a)\right] \cdot A^{\pi_\theta}_{R_\omega}(s,a)\right]_\omega\right] \\ &\approx \omega + \beta \cdot \mathbb{E}_{\mu_{s,a}}\left[\nabla_\omega \left[A^{\pi_\theta}_{\bar{R}}(s,a) \cdot A^{\pi_\theta}_{R_\omega}(s,a)\right]_\omega\right].\end{aligned} \tag{21}$$

In contrast to equation (17), the proposed gradient update term in equation (21) does not require the policy gradient parameters. Rather, it relies only on the value function, enabling the update method for the reward parameter ω to be applied to both policy-based and value-based RL agents.

Data availability

There are no static data associated with the sparse-reward tasks and high-dimensional control tasks. All data are generated from scratch by the agent each time it learns in the classical OpenAI control environments (https://gymnasium.farama.org/environments/classic_control/) and MuJoCo environments (https://mujoco.org/). For the two real-world tasks, the environments used in this work are publicly available at https://github.com/zhshao17/Discovery-of-Optimal-Reward-function. The data generated in this study[69] are available at https://doi.org/10.5281/zenodo.17331039.

Code availability

All the code related to this study[70] was developed in Python and are publicly available at https://doi.org/10.5281/zenodo.17301991, which links to the GitHub repository: https://github.com/zhshao17/Discovery-of-Optimal-Reward-function.

References

1. Howard, D. et al. Evolving embodied intelligence from materials to machines. *Nat. Mach. Intell.* **1**, 12–19 (2019).
2. Ektefaie, Y. et al. Evaluating generalizability of artificial intelligence models for molecular datasets. *Nat. Mach. Intell.* **6**, 1512–1524 (2024).
3. Moor, M. et al. Foundation models for generalist medical artificial intelligence. *Nature* **616**, 259–265 (2023).
4. Cao, H. et al. The immense impact of reverse edges on large hierarchical networks. *Engineering* **36**, 240–249 (2024).
5. Nygaard, T. F., Martin, C. P., Torresen, J., Glette, K. & Howard, D. Real-world embodied ai through a morphologically adaptive quadruped robot. *Nat. Mach. Intell.* **3**, 410–419 (2021).
6. Bartolozzi, C., Indiveri, G. & Donati, E. Embodied neuromorphic intelligence. *Nat. Commun.* **13**, 1024 (2022).
7. Gupta, A., Savarese, S., Ganguli, S. & Fei-Fei, L. Embodied intelligence via learning and evolution. *Nat. Commun.* **12**, 5721 (2021).
8. Sutton, R. S. Reinforcement learning: An introduction. A Bradford Book (2018).
9. Mnih, V. et al. Human-level control through deep reinforcement learning. *Nature* **518**, 529–533 (2015).
10. Vinyals, O. et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature* **575**, 350–354 (2019).
11. Ju, H., Juan, R., Gomez, R., Nakamura, K. & Li, G. Transferring policy of deep reinforcement learning from simulation to reality for robotics. *Nat. Mach. Intell.* **4**, 1077–1087 (2022).
12. Meng, Y. et al. Preserving and combining knowledge in robotic lifelong reinforcement learning. *Nat. Mach. Intell.* **7**, 256–269 (2025).
13. Wang, Z., Xu, Y., Wang, D., Yang, J. & Bao, Z. Hierarchical deep reinforcement learning reveals a modular mechanism of cell movement. *Nat. Mach. Intell.* **4**, 73–83 (2022).
14. Yang, J., Soltan, A. A., Eyre, D. W. & Clifton, D. A. Algorithmic fairness and bias mitigation for clinical machine learning with deep reinforcement learning. *Nat. Mach. Intell.* **5**, 884–894 (2023).
15. Guo, D. et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature* **645**, 633–638 (2025).
16. Wang, G. et al. Extinction chains reveal intermediate phases between the safety and collapse in mutualistic ecosystems. *Engineering* **43**, 89–98 (2024).
17. Schultz, W. A dopamine mechanism for reward maximization. *Proc. Natl Acad. Sci.* **121**, 2316658121 (2024).
18. Lu, R. et al. Adaptive optimal surrounding control of multiple unmanned surface vessels via actor-critic reinforcement learning. *IEEE Trans. Neural Netw. Learn. Syst.* **36**, 12173–12186 (2025).
19. Schütt, H. H., Kim, D. & Ma, W. J. Reward prediction error neurons implement an efficient code for reward. *Nat. Neurosci.* **27**, 1546–1726 (2024).
20. Schaffner, J., Bao, S. D., Tobler, P. N., Hare, T. A. & Polania, R. Sensory perception relies on fitness-maximizing codes. *Nat. Hum. Behav.* **7**, 1135–1151 (2023).
21. Lu, R. et al. Deep reinforcement learning-based demand response for smart facilities energy management. *IEEE Trans. Ind. Electron.* **69**, 8554–8565 (2022).
22. Davidson, G., Todd, G., Togelius, J., Gureckis, T. M. & Lake, B. M. Goals as reward-producing programs. *Nat. Mach. Intell.* **7**, 205–220 (2025).
23. Kim, T. et al. Wing-strain-based flight control of flapping-wing drones through reinforcement learning. *Nat. Mach. Intell.* **6**, 992–1005 (2024).
24. Lee, J. et al. Learning robust autonomous navigation and locomotion for wheeled-legged robots. *Sci. Robot.* **9**, 9641 (2024).
25. Silver, D., Singh, S., Precup, D. & Sutton, R. S. Reward is enough. *Artif. Intell.* **299**, 103535 (2021).
26. Lu, R. et al. Reward shaping-based actor–critic deep reinforcement learning for residential energy management. *IEEE Trans. Ind. Inform.* **19**, 2662–2673 (2023).
27. Lu, R. et al. A novel hybrid-action-based deep reinforcement learning for industrial energy management. *IEEE Trans. Ind. Inform.* **20**, 12461–12475 (2024).
28. Khalvati, K., Park, S. A., Dreher, J.-C. & Rao, R. P. A probabilistic model of social decision making based on reward maximization. In: Proceedings of the 30th International Conference on Neural Information Processing Systems, pp. 2909–2917 (2016).
29. Duéñez-Guzmán, E. A., Sadedin, S., Wang, J. X., McKee, K. R. & Leibo, J. Z. A social path to human-like artificial intelligence. *Nat. Mach. Intell.* **5**, 1181–1188 (2023).
30. Kaufmann, E. et al. Champion-level drone racing using deep reinforcement learning. *Nature* **620**, 982–987 (2023).
31. Veviurko, G., Böhmer, W. & De Weerdt, M. To the max: reinventing reward in reinforcement learning. In: *Proceedings of the 41st*

International Conference on Machine Learning, pp. 49455–49470 (2024).
32. Lu, R., Li, Y.-C., Li, Y., Jiang, J. & Ding, Y. Multi-agent deep reinforcement learning based demand response for discrete manufacturing systems energy management. *Appl. Energy* **276**, 115473 (2020).
33. Fluri, L. et al. The perils of optimizing learned reward functions: Low training error does not guarantee low regret. preprint arXiv:2406.15753 (2024).
34. Booth, S. et al. The perils of trial-and-error reward design: misdesign through overfitting and invalid task specifications. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 5920–5929 (2023).
35. Knox, W. B., Allievi, A., Banzhaf, H., Schmitt, F. & Stone, P. Reward (mis) design for autonomous driving. *Artif. Intell.* **316**, 103829 (2023).
36. Skalse, J., Howe, N. H., Krasheninnikov, D. & Krueger, D. Defining and characterizing reward hacking. In: *Proceedings of the 36th International Conference on Neural Information Processing Systems,* pp. 9460–9471 (2022).
37. Navalpakkam, V., Koch, C., Rangel, A. & Perona, P. Optimal reward harvesting in complex perceptual environments. *Proc. Natl Acad. Sci. USA* **107**, 5232–5237 (2010).
38. Ng, A. Y. & Russell, S. J. Algorithms for inverse reinforcement learning. In: *Proceedings of the 17th International Conference on Machine Learning*, pp. 663–670 (2000).
39. Daniel, C., Viering, M., Metz, J., Kroemer, O. & Peters, J. Active reward learning. In: *Robotics: Science and Systems*, vol. 98 (2014).
40. Balakrishnan, S., Nguyen, Q. P., Low, B. K. H. & Soh, H. Efficient exploration of reward functions in inverse reinforcement learning via bayesian optimization. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pp. 4187–4198 (2020).
41. Chen, L., Paleja, R. & Gombolay, M. Learning from suboptimal demonstration via self-supervised reward regression. In: *Conference on Robot Learning*, pp. 1262–1277 (2021).
42. Ouyang, L. et al. Training language models to follow instructions with human feedback. In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pp. 27730–27744 (2022).
43. Wang, Y., Hu, Y., Wu, F. & Chen, Y. Automatic reward design via learning motivation-consistent intrinsic rewards. preprint arXiv:2207.14722 (2022).
44. Palan, M., Shevchuk, G., Charles Landolfi, N. & Sadigh, D. Learning reward functions by integrating human demonstrations and preferences. In: *Robotics: Science and Systems* (2019).
45. Lu, Y., Wang, Q., Cao, H., Xu, X. & Zhang, M. Smoothed preference optimization via renoise inversion for aligning diffusion models with varied human preferences. In: *Proceedings of the 42nd International Conference on Machine Learning* (2025).
46. Wang, W., Li, R., Chen, Y., Diekel, Z. M. & Jia, Y. Facilitating human–robot collaborative tasks by teaching-learning-collaboration from human demonstrations. *IEEE Trans. Autom. Sci. Eng.* **16**, 640–653 (2018).
47. Christiano, P.F. et al. Deep reinforcement learning from human preferences. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 4302–4310 (2017)
48. Knox, W.B. et al. Models of human preference for learning reward functions. Preprint arXiv:2206.02231 (2022).
49. Knox, W. B. et al. Learning optimal advantage from preferences and mistaking it for reward. In: *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 38, pp. 10066–10073 (2024).
50. Stadie, B., Zhang, L. & Ba, J. Learning intrinsic rewards as a bi-level optimization problem. In: *Conference on Uncertainty in Artificial Intelligence*, pp. 111–120 (2020).
51. Devidze, R., Radanovic, G., Kamalaruban, P. & Singla, A. Explicable reward design for reinforcement learning agents. In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*, pp. 20118–20131 (2021).
52. Devidze, R., Kamalaruban, P. & Singla, A. Exploration-guided reward shaping for reinforcement learning under sparse rewards. In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pp. 5829–5842 (2022).
53. Chen, E., Hong, Z.-W., Pajarinen, J. & Agrawal, P. Redeeming intrinsic rewards via constrained optimization. In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pp. 4996–5008 (2022).
54. Ma, H., Luo, Z., Vo, T. V., Sima, K. & Leong, T.-Y. Highly efficient self-adaptive reward shaping for reinforcement learning. In: *Proceedings of the 13th International Conference on Learning Representations* (2025).
55. Sorg, J., Singh, S. & Lewis, R. L. Reward design via online gradient ascent. In: *Proceedings of the 23rd International Conference on Neural Information Processing Systems*-Vol. 2, pp. 2190–2198 (2010).
56. Grunitzki, R., Silva, B. C. & Bazzan, A. L. A flexible approach for designing optimal reward functions. In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 1559–1561 (2017).
57. Hu, Y. et al. Learning to utilize shaping rewards: a new approach of reward shaping. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pp. 15931–15941 (2020).
58. Liu, R., Bai, F., Du, Y. & Yang, Y. Meta-reward-net: implicitly differentiable reward learning for preference-based reinforcement learning. In: *Proceedings of the 36th International Conference on Neural Information Processing Systems,* pp. 22270–22284 (2022).
59. Brockman, G. Openai gym. preprint arXiv:1606.01540 (2016).
60. Mnih, V. Playing atari with deep reinforcement learning. preprint arXiv:1312.5602 (2013).
61. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal policy optimization algorithms. preprint arXiv:1707.06347 (2017).
62. Haarnoja, T., Zhou, A., Abbeel, P. & Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: *Proceedings of the 35th International Conference on Machine Learning*, pp. 1861–1870 (2018).
63. Maaten, L. & Hinton, G. Visualizing data using t-sne. *J. Mach. Learn. Res.* **9**, 2579–2605 (2008).
64. Fujimoto, S., Hoof, H. & Meger, D. Addressing function approximation error in actor-critic methods. In: *Proceedings of the 35th International Conference on Machine Learning*, pp. 1587–1596 (2018).
65. Wagenmaker, A. J., Chen, Y., Simchowitz, M. & Du, S., Jamieson, K. First-order regret in reinforcement learning with linear function approximation: A robust estimation approach. In: Proceedings of the 39th International Conference on Machine Learning, pp. 22384–22429 (2022).
66. Bellman, R. Dynamic programming. *Science* **153**, 34–37 (1966).
67. Sutton, R. S., McAllester, D., Singh, S. & Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In: *Proceedings of the 12th International Conference on Neural Information Processing Systems*, pp. 1057–1063 (1999).
68. Metelli, A. M., Papini, M., Montali, N. & Restelli, M. Importance sampling techniques for policy optimization. *J. Mach. Learn. Res.* **21**, 1–75 (2020).
69. Shao, Z. & Lu, R. Discovery of reward function for Natcomm. https://doi.org/10.5281/zenodo.17331039 (2025).

70. Shao, Z. zhshao17/Discovery-of-Optimal-Reward-function: NatComm_release. https://doi.org/10.5281/zenodo.17301991 (2025).

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grants 62225306, 62425602, U2141235, 62133003, 52188102, and 62373158; in part by the National Key Research and Development Program of China under Grant 2022ZD0119601; in part by Scientific and Technological Innovation 2030—Major Project 2021ZD0201300; in part by the State Key Laboratory of Mechanical Transmission for Advanced Equipment under Grant SKLMT-MSKFKT-202317; and in part by the Open Research Project of the State Key Laboratory of Industrial Control Technology, China, under Grant ICT2024B22. The computation is completed in the HPC Platform of Huazhong University of Science and Technology.

Author contributions

Conceptualization: R.L., H.Z., and Y.D.; Methodology: R.L., Z.S., and H.Z.; Software: R.L. and Z.S.; Formal Analysis: R.L., H.Z., Z.S., and R.C.; Data curation: R.L., Z.S., and Y.D.; Writing, original draft: R.L., H.Z., and Z.S.; Writing, review, and editing: Y.D., D.W., H.S., T.Y., F.Z., J.W., Y.S., Z.J., H.D., and H.Z.; Funding acquisition: H.D. and H.Z.; Supervision: H.D. and H.Z. All the authors interpreted the results, commented, and approved the final version of the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41467-025-66009-y.

Correspondence and requests for materials should be addressed to Yuemin Ding or Hai-Tao Zhang.

Peer review information *Nature Communications* thanks the anonymous, reviewer(s) for their contribution to the peer review of this work. A peer review file is available.

[1]School of Artificial Intelligence and Automation, Engineering Research Center of Autonomous Intelligent Unmanned Systems, Chinese Ministry of Education, State Key Laboratory of Digital Manufacturing Equipment and Technology, Institute of Artificial Intelligence, Huazhong University of Science and Technology, Luoyu Road 1037, Wuhan 430074 Hubei, China. [2]Tecnun School of Engineering, University of Navarra, Paseo de Manuel Lardizabal, 13, 20018 San Sebastián, Spain. [3]Ikerbasque Foundation, Plaza Euskadi 5, 48009 Bilbao, Spain. [4]Research Center for Applied Mathematics and Interdisciplinary Science, School of Mathematics and Statistics, Wuhan Textile University, No.1 Sunshine Avenue, Wuhan 430200 Hubei, China. [5]Key Laboratory of Image Processing and Intelligent Control, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Luoyu Road 1037, Wuhan 430074 Hubei, China. [6]State Key Laboratory of Synthetical Automation for Process Industrie, Northeastern University, Wenhua Road 3-11, Shenyang 110819 Liaoning, China. [7]Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Sai Kung Peninsula, Hong Kong, China. [8]Department of Computer Science and Department of Data Science, City University of Hong Kong, Kowloon, Hong Kong, China. [9]Department of Mechanical Engineering, University of Victoria, 3800 Finnerty Road, Victoria, Canada. [10]Department of Electrical and Computer Engineering, New York University, 370 Jay Street, Brooklyn, NY 11201, USA. [11]State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Luoyu Road 1037, Wuhan 430074 Hubei, China. ✉e-mail: yueminding@tecnun.es; zht@mail.hust.edu.cn