Proceedings of the 41st Chinese Control Conference
July 25-27, 2022, Hefei, China

Shared Autonomy Based on Human-in-the-loop Reinforcement Learning with Policy Constraints

Ming Li[1], Yu Kang[1,2,3,*], Yun-Bo Zhao[1], Jin Zhu[1], Shiyi You[1]

1. Department of Automation, University of Science and Technology of China, Hefei 230026, China
E-mail: lm92@mail.ustc.edu.cn, kangduyu@ustc.edu.cn, ybzhao@ustc.edu.cn, jinzhu@ustc.edu.cn, ysy3765@mail.ustc.edu.cn

2. Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230088, China

3. Institute of Advanced Technology, University of Science and Technology of China, Hefei 230088, China

Abstract: In shared autonomous systems, humans and agents cooperate to complete tasks. Since reinforcement learning enables agents to obtain good policies through trial and error without knowing the dynamic model of the environment, it has been well applied in shared autonomous systems. After inferring the target from human inputs, agents trained by RL can accurately act accordingly. However, existing methods of this kind have big problems: the training of reinforcement learning algorithms require lots of exploration, which is time-consuming, lack of security guarantee and likely to cause great losses in the training process. Moreover, most of shared control methods are human-oriented, and do not consider the situation that humans may make wrong actions. In view of the above problems, this paper proposes human-in-the-loop reinforcement learning with policy constraints. In the training process, human prior knowledge is used to constrain the exploration of agents to achieve fast and efficient learning. In the process of testing we incorporate policy constraints in the arbitration to avoid serious consequences caused by human mistakes.

Key Words: Shared Autonomy, Reinforcement Learning, Human-in-the-loop, Policy Constraints

1 Introduction

Artificial Intelligence (AI) has come a long way in the past few decades. Among numerous solutions to sequential decision problems, Reinforcement Learning (RL) stands out. RL agents can learn through continuous interaction with the environment to obtain data, which has great advantages in solving model-free problems. Nowadays, RL has been well used in many fields, such as Game [1], Finance [2], Medicine [3]and so on. Google recently managed to control plasma autonomously in a fusion device using RL [4].

We focus on shared autonomous systems, in which RL plays a good role. In fact, many problems are intractable for humans or intelligent agents alone [9]. For example, it is difficult for human beings to realize the motion and attitude control of high-degree of freedom objects such as multi-arm robot and quadrotor aircraft, but it is very simple for machines. On the contrary, the problem of selecting targets is too difficult for machines. Shared autonomous systems aim to combine human actions and agent's policies to accomplish related tasks or improve policy performance [5–7]. Shared autonomy, where actions are performed after interactions between humans and agents, has been proven to achieve better performance than a single human or agent and is more practical in the application of real scenes [10]. Because of the advantages of RL, the shared autonomous system trained by RL can solve the sequential decision problems without environment model, which is a great improvement.

However, even though RL helps the shared autonomous systems solve some sequential decision problems and makes a great breakthrough in this field, it is not feasible to rely only on RL algorithms for many realistic complex tasks [8]. It's important to note that in RL algorithms, agents need a lot of interaction data to get a good strategy because of the high-dimensional nature of the action space and the state space. However, there will be huge costs and security threats due to lots of exploration in the real environment, which prevent the development of RL in the real world. For example, traffic accidents and vehicle damage are prone to occur in the training process of autonomous vehicles. Moreover, it'll tend to occur overfitting for RL agents which means once the actual environment changes greatly, the original strategy will become ineffective.

In order to solve these problems, some researchers propose to use human prior knowledge to accelerate the training process, such as introducing expert strategies to avoid dangerous actions in the exploration process [24], setting specific constraint functions [12], and letting agents learn target strategies through human demonstrations [13]. Siddharth proposed human-in-the-loop RL [14], which combined the advantages of RL in dealing with model-free sequential decision problems with human's prior knowledge. But in this method, on the one hand, human participation in the training of RL agents only depends on the setting of rewards, which can accelerate the training, but ignores the fatal defect of RL: the lack of safety assurance in the training process. Because agents need to explore, they are prone to make dangerous behaviors and cause great losses. On the other hand, the algorithm is human-centered in the execution process, but ignores situations in which human may make bad decisions, such as the wrong steering, due to fatigue driving in the assisted autopilot system. However, these methods mentioned above only help agents learn strategies, which are not necessarily optimal. In fact, these methods will obtain suboptimal solutions due to the bias of human prior knowledge.

Inspired by Tong's proposal of Constraint Sampling Reinforcement Learning (CSRL) [15], we propose a human-in-the-loop RL method under the policy constraints(HRLPC) to solve the problems of the above method. In view of the lack of security guarantee of RL algorithm and the problem

This work was supported by the National Key Research and Development Program of China (No. 2018AAA0100800)

* Corresponding author: Yu kang

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on November 10,2022 at 06:25:00 UTC from IEEE Xplore. Restrictions apply.

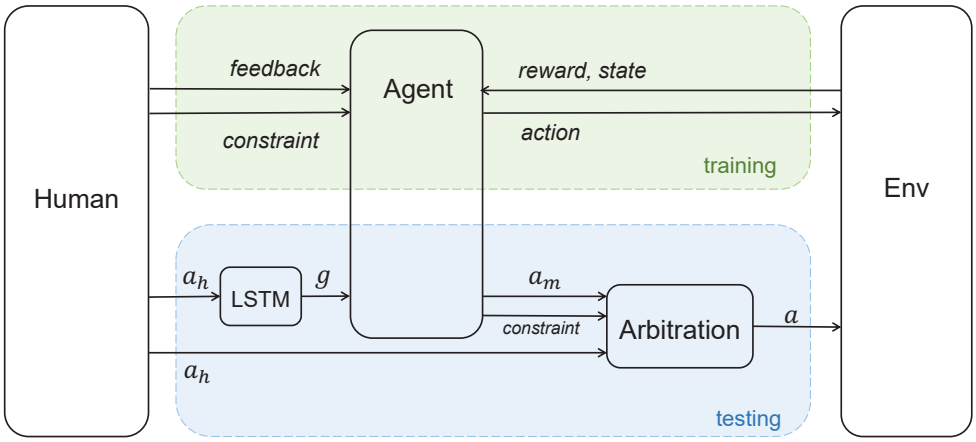


Fig. 1: The framework. (1) In the training process, human set policy constraints on RL based agent, and the agent will receive additional reward from human feedback in the interactions with the environment. (2) In the testing process, the agent deduces the target according to the output of the human, and obtains the final action through the arbitration mechanism under the restriction of the policy.

that it is easy to explore the defects of suboptimal solutions, different policy restrictions are artificially added to multiple agents, and then the policy elimination is constantly carried out in the training process. Finally, agents with optimal policy solutions are obtained while ensuring security. Unlike CSRL, we have added different types of policy constraints to the training and execution of human-machine systems. The agent's RL algorithm can converge to the optimal solution quickly in the training process, and at the same time, the monitoring mechanism is introduced in the execution process to limit the irrational decisions made by human beings, so as to realize efficient and safe human-machine system decision-making.

The rest of this paper is organized as follows. Section II introduces some related works. Section III elaborates our approach. Section VI describes our experimental process and analysis of experimental results. We conclude in section V.

2 Related Work

2.1 Human-in-the-loop RL

Unlike shared autonomy [10], which integrates user input and machine output to enable human-machine systems to successfully complete tasks during test time. Human-in-the-loop RL incorporates prior knowledge such as human feedback into the training loop without relying on the input of independent users [18]. All these RL algorithms are based on the human-in-the-loop framework and generate many methods from different perspectives such as the degree and way of human participation. In the training process, agents can learn from human demonstration, learn reward functions through Inverse Reinforcement Learning [16], or accelerate training and restrain agents' exploration through human feedback [17]. Michael proposed HG-Dagger(a variant of DAgger) [19] that guarantees the security of agent exploration through expert intervention. The essence of the above algorithms are to positively guide agents exploration based on human prior knowledge. The algorithm proposed by us is also based on the above human-in-the-loop framework in training time, but the biggest difference from other algorithms is the way of utilizing human prior knowledge.

2.2 Safe RL

Many people are concerned about the application of RL because of the high cost and security risks brought by exploration in the training process of RL algorithm in real scenes. Safety RL [20] can be defined as a process of policy learning that maximizes return expectation under the premise of ensuring reasonable system performance and meeting safety constraints during training and testing. Recently more and more people have been studying safe RL, such as constrained optimization methods [29], or using an estimate to predict cost [22]. Saunders [23] proposes a decider trained by imitation learning to decide when to intervene with agents who engage in dangerous behaviors. Zhenghao [24] proposes EGPO that guarantees exploration security, requiring only the guardian without structural assumptions. Different from the above methods, Our method considers the security of the system both in training time and testing time.

2.3 Constrained RL

Much previous work considered combinations of single or multiple constraints, such as rewards, dynamic transfer models and so on. Previously Altman [25] has studied constraint RL under the assumption that all the model information is known, such as transition probability, reward function, etc. Later, many scholars began to study the combination of various constraints, some of which can be unknown [26–28]. Recently constrained RL develops a lot of variants, such as algorithms based on Actor-Critic framework [28] or policy search [29]. The constraints we consider are consistent with Tong's weak constraint [15], which is to transform human prior knowledge into programmable formulas that can guide agents to reduce unnecessary and erroneous exploration.

3 Method

For shared autonomy systems, tasks cannot be completed by humans or machines alone. With human input, machines are no longer aimless and can accomplish high-

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on November 10,2022 at 06:25:00 UTC from IEEE Xplore. Restrictions apply.

dimensional control tasks. For these human-machine systems, we propose human-in-the-loop RL with policy constraints(HRLPC), which are applied in two parts: the training process and the testing process. The framework of the whole method is shown in Fig. 1.

We incorporate prior knowledge into the training process and impose different policy constraints on different agents. During training, agents with inappropriate policy restrictions are constantly eliminated and finally the remaining agent can obtain the best policy. During the test, the best agent and human make decisions together. Through the arbitration mechanism with security guarantee, the final behavior of the system is optimal and safe.

3.1 The Training Process

We choose a group of agents to learn based on our algorithm, and all agents share the experience pool. The algorithm we proposed combine Rainbow [30] with human prior knowledge which consists of reward feedback and policy constraints. During training we assume that the model is unknown and that the purpose of agents is fixed because there is no human input during training.

As a general characteristic of shared autonomous systems, the reward equation includes human feedback in addition to rewards for performing tasks.

$$R = R_{task}(s, a, s') + R_{feedback}(s, a, s') \tag{1}$$

Human feedback can help agents quickly find targets in training. Once agents succeed, we can stop providing feedback, which greatly reduces the training burden of humans.

We make different sets of policy constraints for each agent, which can be empty (no constraint). $C(s)$ represents the mapping from states to allowed actions, and let $C_k(s)$ denote the policy constraint of agent k. We divided the training process into three steps: (a) Before the beginning of each episode, select an agent for training. (b) Relevant agents share the episode's data and update model parameters. (c) Eliminate agents whose policy restrictions meet certain conditions. Each episode repeats the above three steps.

We now explain the selection rule of agent in the first step in detail. We denote Φ as the set of agents and the past average return of agent k as $\hat{\mu}_k$. The upper confidence bandit (UCB) method is used to choose agent, as shown in equation (2). We assume the confidence function $B(h, n)$ is of the form $\frac{z(h)}{n^\eta}$. The detail of confidence function is discussed in section 4.

$$k^* = \arg\max_{k \in \Phi}(\hat{\mu}_k + B(h, n)) \tag{2}$$

As Siddharth [14] pointed out, one-step update will affect human control due to the delay of task interface, so we updated model parameters after the end of each episode. After updating agents' parameters, two conditions should be met to determine the elimination of an agent k: (a) the model parameters of agent k are stable. (b) There is at least one agent that has better performance and a looser policy constraint than agent k. For RL algorithms based on Q value function, we can intuitively judge whether the value function network is stable through δ_τ^k, the average of the approximate temporal difference error under the observed trajectory τ. Formally,

$$\delta_\tau^k = \sum_\tau ((r_t + \max_{a' \in C_k} Q_t^k(s', a')) - Q_t^k(s, a)) \tag{3}$$

Besides, a looser policy constraint than agent k means that other actions can be performed in addition to those allowed by agent k. It's intuitive because fewer restrictions allow agents to explore more without sacrificing performance. As Tong proved, when the base learning method converges, this constraint sampling approach is guaranteed to converge even though it appears that the return of each arm of UCB is not subject to a static distribution.

Algorithm 1: Human-in-the-loop RL with Policy Constraint(HRLPC)

Initialize $\hat{\mu_k}$, $B(h, n)$, T_n, T_l ;
Initialize agents with different policy constraint ;
Initialize agents' parameters and hyperparameters ;
Initialize the change D_k = [ ], $\forall k \in \mathcal{A}$;
for *Episode i=1,2,...* **do**
  Select agent k to explore by using equation (2) ;
  Generate trajectory τ_i ;
  $R_{task} = \sum_{t=1}^{len(\tau_i)}$;
  $R_i = R_{task} + R_{feedback}$;
  $n_k = n_k + 1$;
  $\hat{\mu_k} = \frac{(n_k - 1)\hat{\mu_k} + R_i}{n_k}$;
  Update the relevant agent ;
  Calculate δ_i and $P_k = [D_k, \delta_i]$;
  if $\forall n \in \{n_k - T_n \leq 0\}$ *and* $D_{k(n)} \leq T_l$ **then**
    Eliminate agent k
  end
end

3.2 The Testing Process

Human Input Unlike the training process where the target is fixed, in the testing process machine needs to infer the new target based on human input. By concatenating agent's orginal state vector and human action vector, we can transmit human intentions to the agent. Obviously, if we know enough model information, we can also directly infer human's purpose from human's historical actions and concatenate agent's original state vector and target position vector.

$$s_t^* = \begin{bmatrix} s_t \\ a_h \end{bmatrix} \tag{4}$$

Shared Control We need an arbitration to determine the final action based on the input of the machine and the human, and it is obvious that the machine can choose the action that is closest to human input while ensuring that the action is not far worse than the best action. But obviously this method is easy to cause serious costs when human make mistakes at a certain moment. So we introduce a protection mechanism on the basis of the original arbitration, when the human behavior will lead to serious consequences, the system's action is dominated by the machine. The final output policy of the shared autonomous system is defined as $\pi(s)$. Formally,

$$\pi(s) = \begin{cases} \arg\max_{a \in \Omega_\alpha} \rho(a, a_h), & a_h \in \Omega_\beta \\ a^*, & \text{otherwise} \end{cases} \tag{5}$$

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on November 10,2022 at 06:25:00 UTC from IEEE Xplore. Restrictions apply.

where the function ρ calculates the similarity between human action and agent's action, $a' = \arg\min_{a \in \mathcal{A}} Q(s,a)$ and $a^* = \arg\max_{a \in \mathcal{A}} Q(s,a)$. We denote two constrained policy spaces as Ω_α and Ω_β respectively. Intuitively, if the human action is reasonable, i.e. $a_h \in \Omega_\beta$, agents will choose the most human-like action from Ω_α. On the contrary, if the human input action is wrong, the human input will be ignored. With the policy constraints, the security of man-machine system is guaranteed.

$$\begin{cases} \Omega_\alpha = \{a | Q(s,a) - Q(s,a') \geq \alpha(Q(s,a^*) - Q(s,a'))\} \\ \Omega_\beta = \{a | Q(s,a) - Q(s,a') \geq \beta(Q(s,a^*) - Q(s,a'))\} \end{cases} \tag{6}$$

Significantly, since the Q value function may be negative, we use the difference between the $Q(s,a)$ and its minimum value to measure the performance of the action a. The hyper-parameters $\Omega_\alpha \in [0,1]$ and $\Omega_\beta \in [0,1]$ control the tolerance of agents to human suboptimal behavior and abnormal input respectively.

4 Experiments

We use the Lunar Lander game from OpenAI Gym as simulation experiment platform. This game is used to simulate the landing of aircraft on the moon, its goal is to land safely at a certain target point. The motion space of the aircraft is four-dimensional and discrete, corresponding to different actuators. The original state space is an 8-dimensional vector that contains information about the lander's position, speed, and indicators of whether or not it has reached the ground. We modified the game setting to randomly generate the target landing site, which the lander doesn't know, on the ground below at the beginning of each episode. The machine needs human input to predict where to land because the target landing site is not available to the machine and appears randomly in each episode, but human know the target landing site between the two flags. So the final state space will be 9-dimensional, with additional information about the location of the destination.

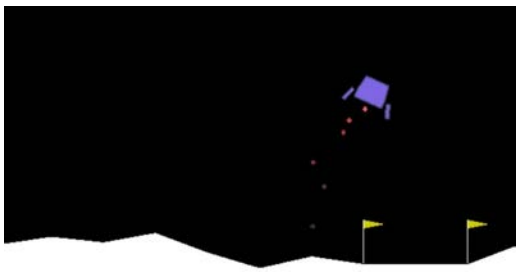


Fig. 2: the LunarLander scene.

Setting We generated 10 policies with different performance levels for different human pilots. The final policy constraint set is the combination of single or multiple human flight strategies. Of course, the policy constraint set can be empty, which means there is no restriction. Rainbow is the basic RL algorithm of agents and the Q value function is a neural network with two hidden layers of 128 units each. In the two parts of the reward function, the setting of R_{task} means that every dangerous action of the lander will be penalized. $R_{feedback}$ appears at the end of the episode. If the lander successfully arrives at the destination safely, human will give a large positive reward; otherwise, a large negative reward will be given. The learning rate of this experiment is 1e-4, the update interval of target network parameters is 4 time steps, and the training batch size is 64. The confidence bound $B(h,n) = c\frac{\sqrt{log(t)}}{s^{1/2}}$, and we set $T_n = 40$ and $T_l = 0.0125$.

4.1 The Training Phase

Results In order to reduce the training burden on human, the lander will be informed of the target location during training, but it needs to deduce the destination information according to human input in the testing phase. As shown in Fig. 3, all the rewards we plot are with 95% confidence intervals. Compared with Rainbow, our method improve the convergence rate and the stability of the returns.

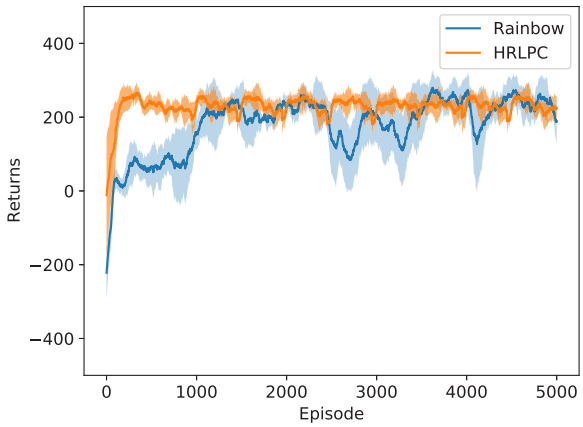


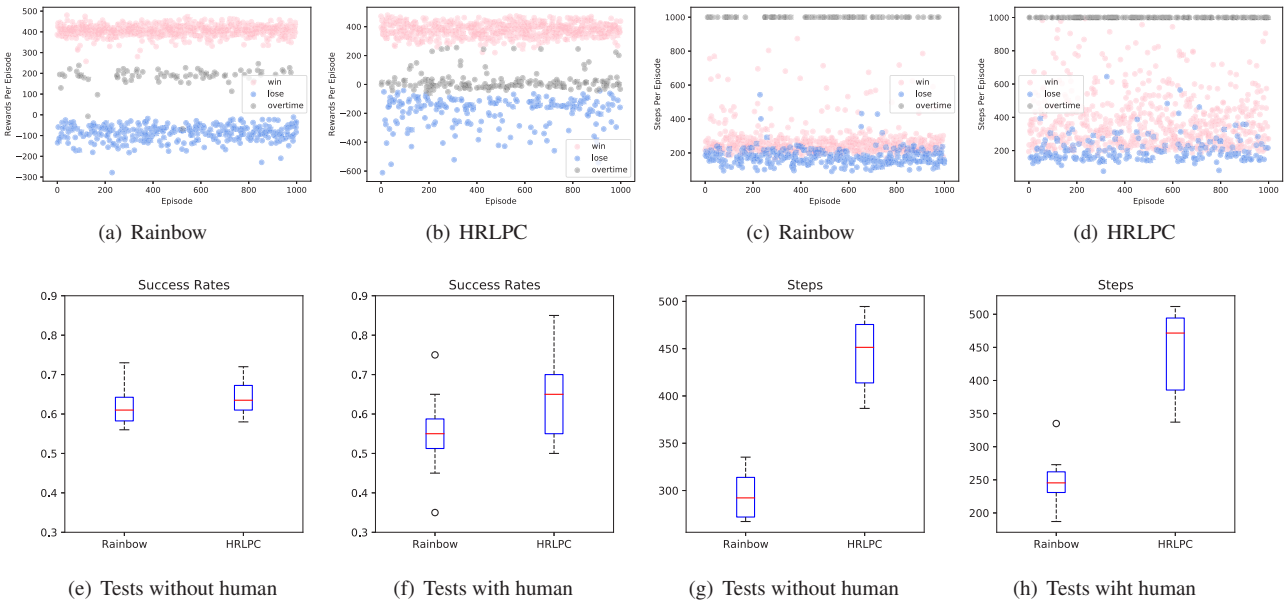
Fig. 3: Returns with 95% confidence intervals

Discussion The results are exactly in line with our expectations. With policy constraints, agents can quickly find the optimal behavior and avoid costs caused by explorations in the training process. It is obvious that if there is a good policy constraint, agents will quickly learn the optimal policy. However, experimental results show that the final reward obtained by using our method is not much higher than other methods. Because in the Lunar Lander, which has a low dimensional action space, the exploration is not difficult for agents and the difference between the suboptimal solution and the optimal solution is small. Even if our method has no great advantage in the return of the episode, agents can still explore more efficiently and steadily with policy constraints.

4.2 The Testing Phase

Intuitively, the environment becomes unstable due to the random distribution of destinations, and it should be difficult for agents to obtain stable strategies based on the single agent RL methods. However, the training results show that stable strategies can still be obtained by using Rainbow if we have encoded destination information into the state vector. But in shared autonomous, it's difficult for agents to understand the target. So we make agents no longer have extra information and keep human in the loop of testing. In the testing process, we invited 12 volunteers(5 women and 7 men) to participate. In order to avoid the result of the ex-

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on November 10,2022 at 06:25:00 UTC from IEEE Xplore. Restrictions apply.



(a) Rainbow (b) HRLPC (c) Rainbow (d) HRLPC

(e) Tests without human (f) Tests with human (g) Tests without human (h) Tests wiht human

Fig. 4: (a-d): In the top row, we plot the scatter plots of returns and steps of agents for 1000 episodes, where agents were tested without human participation since they were told the target for each episode. (e-h): On the bottom row, we plot the box plots of agents' win rate and average steps in the testing process both with and without human participation.

periment not being fair because someone did not participate seriously, we offered a bonus for each participant, with an additional bonus for the person with the highest success rate.

Results As Fig. 4 shows, our approach(HRLPC) has a significant lead over Rainbow. In the scatter plots, HRLPC has a lower crash rate but a higher average elapsed time, whereas Rainbow had a lower average elapsed time, regardless of win or loss, but a higher crash rate. As we can see from the box charts, when considering the more realistic shared autonomy scenario, where the agent cannot acquire the target on their own, human participation does increase HRLPC's win rate but Rainbow's win rate drops significantly. At the same time, human involvement reduced Rainbow's average elapsed time but increased the average steps of HRLPC slightly.

Discussion Although the training results shows that our method is not much different from Rainbow's ultimate returns, and it just has an advantage in convergence speed. Actual the testing results also shows that Rainbow's success rate is only slightly worse than our approach when agents know the target in advance. Because the action space is only 4-dimensional, the exploration scope of agent is not particularly large, and the optimal solution can always be found eventually after long-term exploration. But this is the biggest drawback of this approach, the long period of unconstraint exploration leads to a lot of risky behavior, which is the reason why Rainbow is aggressive in training and testing process, with not only a high win rate but also a high crash rate. It can be easily found from the scatter diagram that agents without policy constraints either crash or land quickly during the test.

When we do not allow the agents to have additional information, i.e. the target location, it is theoretically possible that the performance of the entire system will decline. First of all, because of the high landing speed and freedom of the aircraft, it is difficult to make it land smoothly with only four movements. Even if agents trained by the RL algorithm can assist human, they need to predict the target location based on human input. Due to the error of the LSTM network itself, there is a gap between the predicted target and the actual target. But actually the results show that HRLPC's success rate actually increased, while Rainbow's success rate dropped significantly. In our analysis, the reason is that in the testing phase the policy constraints of agents still exist, agents will take the rationality of human input actions into extra consideration. As long as human don't always deliver wrong target location information, the system can complete the task. Moreover, agents will ignore the input of unsafe human behavior and make the system act as safely as possible, which is the reason why human participation increases the average time of our method.

5 Conclusion

For shared autonomy, the system requires humans and agents to perform tasks together. We propose human-in-the-loop reinforcement learning with policy constraints that leverage human priori knowledge to design human feedback rewards and policy constraints. On the one hand, by constraining agents' exploration, they can quickly and efficiently learn the optimal strategy and reduce the cost in the training process. On the other hand, in the testing process, the influence brought by human dangerous actions is limited, which ensures the security and stability of system performance in the testing process.Finally, The experimental results confirm that our method greatly improves the sampling efficiency and safety in the training process as well as the stability and success rate in the testing process.

Our method still needs to be improved in the future, such as improving the accuracy of intention inference, designing better policy constraints to further improve sampling efficiency and system performance, and designing better arbitra-

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on November 10,2022 at 06:25:00 UTC from IEEE Xplore. Restrictions apply.

tion mechanism. What is worth further thinking is what kind of policy constraints can make the best use of human priori knowledge, which is still an open question to be solved.

References

[1] Zha D, Xie J, Ma W, et al, Douzero: Mastering doudizhu with self-play deep reinforcement learning, *International Conference on Machine Learning*, 2021: 12333-12344.

[2] Liu X Y, Yang H, Chen Q, et al, Finrl: A deep reinforcement learning library for automated stock trading in quantitative finance, arXiv preprint arXiv:2011.09607, 2020.

[3] Liao X, Li W, Xu Q, et al, Iteratively-refined interactive 3D medical image segmentation with multi-agent reinforcement learning, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020: 9394-9402.

[4] Degrave J, Felici F, Buchli J, et al, Magnetic control of tokamak plasmas through deep reinforcement learning, *Nature*, 602(7897): 414-419, 2022

[5] Abbink D A, Carlson T, Mulder M, et al, A topology of shared control systems—finding common ground in diversity, *IEEE Transactions on Human-Machine Systems*, 2018, 48(5): 509-525.

[6] Lin Z, Harrison B, Keech A, et al, Explore, exploit or listen: Combining human feedback and policy model to speed up deep reinforcement learning in 3d worlds, arXiv preprint arXiv:1709.03969, 2017.

[7] Zhang Q, Kang Y, Zhao Y B, et al, Traded Control of Human-Machine Systems for Sequential Decision-Making Based on Reinforcement Learning, *IEEE Transactions on Artificial Intelligence*, 2021.

[8] Henderson P, Islam R, Bachman P, et al, Deep reinforcement learning that matters,*Proceedings of the AAAI conference on artificial intelligence*, 2018, 32(1).

[9] Goertz R C, Manipulators used for handling radioactive materials, *Human factors in technology*, 1963: 425-443.

[10] Javdani S, Srinivasa S S, Bagnell J A, Shared autonomy via hindsight optimization, *Robotics science and systems: online proceedings, 2015*, 2015.

[11] Peng Z, Li Q, Liu C, et al, Safe Driving via Expert Guided Policy Optimization, *Conference on Robot Learning*, PMLR, 2022: 1554-1563.

[12] Buchli J, Stulp F, Theodorou E, et al, Learning variable impedance control, *The International Journal of Robotics Research*, 2011, 30(7): 820-833.

[13] Wu Y H, Charoenphakdee N, Bao H, et al, Imitation learning from imperfect demonstration, *International Conference on Machine Learning*, PMLR, 2019: 6818-6827.

[14] Reddy S, Dragan A D, Levine S, Shared autonomy via deep reinforcement learning[J]. arXiv preprint arXiv:1802.01744, 2018.

[15] Mu T, Theocharous G, Arbour D, et al, Constraint Sampling Reinforcement Learning: Incorporating Expertise For Faster Learning, arXiv preprint arXiv:2112.15221, 2021.

[16] Ziebart B D, Maas A L, Bagnell J A, et al, Maximum entropy inverse reinforcement learning, *Aaai. 2008*, 8: 1433-1438.

[17] Guan L, Verma M, Guo S, et al, Explanation augmented feedback in human-in-the-loop reinforcement learning, arXiv preprint arXiv:2006.14804, 2020.

[18] Goecks V G, Human-in-the-loop methods for data-driven and reinforcement learning systems, arXiv preprint arXiv:2008.13221, 2020.

[19] Kelly M, Sidrane C, Driggs-Campbell K, et al, Hg-dagger: Interactive imitation learning with human experts, *2019 International Conference on Robotics and Automation (ICRA). IEEE*, 2019: 8077-8083.

[20] Garcıa J, Fernández F, A comprehensive survey on safe reinforcement learning, *Journal of Machine Learning Research*, 2015, 16(1): 1437-1480.

[21] Achiam J, Held D, Tamar A, et al, Constrained policy optimization, *International conference on machine learning*, PMLR, 2017: 22-31.

[22] Srinivasan K, Eysenbach B, Ha S, et al, Learning to be safe: Deep rl with a safety critic, arXiv preprint arXiv:2010.14603, 2020.

[23] Saunders W, Sastry G, Stuhlmueller A, et al, Trial without error: Towards safe reinforcement learning via human intervention, arXiv preprint arXiv:1707.05173, 2017.

[24] Peng Z, Li Q, Liu C, et al. Safe Driving via Expert Guided Policy Optimization, *Conference on Robot Learning*, PMLR, 2022: 1554-1563.

[25] Altman E, *Constrained Markov decision processes: stochastic modeling*, Routledge, 1999.

[26] Efroni Y, Mannor S, Pirotta M, Exploration-exploitation in constrained mdps, arXiv preprint arXiv:2003.02189, 2020.

[27] Zheng L, Ratliff L, Constrained upper confidence reinforcement learning, *Learning for Dynamics and Control*, PMLR, 2020: 620-629.

[28] Bhatnagar S, Lakshmanan K, An online actor–critic algorithm with function approximation for constrained markov decision processes, *Journal of Optimization Theory and Applications*, 2012, 153(3): 688-708.

[29] Achiam J, Held D, Tamar A, et al, Constrained policy optimization, *International conference on machine learning*, PMLR, 2017: 22-31.

[30] Hessel M, Modayil J, Van Hasselt H, et al, Rainbow: Combining improvements in deep reinforcement learning, *Thirty-second AAAI conference on artificial intelligence*, 2018.

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on November 10,2022 at 06:25:00 UTC from IEEE Xplore. Restrictions apply.