Deep reinforcement learning-based rehabilitation robot trajectory planning with optimized reward functions

Xusheng Wang1, Jiexin Xie1, Shijie Guo1, Yue Li2, Pengfei Sun3 and Zhongxue Gan1

Abstract
Deep reinforcement learning (DRL) provides a new solution for rehabilitation robot trajectory planning in the unstructured working environment, which can bring great convenience to patients. Previous researches mainly focused on optimization strategies but ignored the construction of reward functions, which leads to low efficiency. Different from traditional sparse reward function, this paper proposes two dense reward functions. First, azimuth reward function mainly provides a global guidance and reasonable constraints in the exploration. To further improve the efficiency, a process-oriented aspiration reward function is proposed, it is capable of accelerating the exploration process and avoid locally optimal solution. Experiments show that the proposed reward functions are able to accelerate the convergence rate by 38.4% on average with the mainstream DRL methods. The mean of convergence also increases by 9.5%, and the percentage of standard deviation decreases by 21.2%–23.3%. Results show that the proposed reward functions can significantly improve learning efficiency of DRL methods, and then provide practical possibility for automatic trajectory planning of rehabilitation robot.

Keywords
Rehabilitation robot, trajectory planning, deep reinforcement learning, reward function

Date received: 20 May 2021; accepted: 26 November 2021
Handling Editor: Chenhui Liang

Introduction
Trajectory planning is a fundamental problem for a rehabilitation robot. Conventional trajectory planning task of rehabilitation robots has always been completed by doctors. However, the disharmony of doctor-patient ratio and lacking of skilled doctors always cause contradictions and bring inconvenience to patients.1–3 Therefore, autonomous trajectory planning is highly expected. Nevertheless, the autonomous trajectory planning of a robot is a challenging task. Patients in rehabilitation training usually have movement restrictions, this requires the robot to avoid some points that the patient physically cannot reach (referred to ban point) during trajectory planning, otherwise, it will cause physical damage to the patient. Traditional trajectory planning methods are usually applicable to structured environment.4,5 However, the working environment of a rehabilitation robot is always changing with the patient’s physical condition, which is difficult to model in advance. In recent years, Deep

1Academy for Engineering and Technology, Fudan University, Shanghai, China
2Department of Computer Technology, Hebei College of Industry and Technology, Shijiazhuang, China
3Beijing Smartchip Microelectronics Technology Company Limited, Beijing, China

Corresponding author:
Zhongxue Gan, Academy for Engineering and Technology, Fudan University, 220 Handan Road, Yangpu District, Shanghai 200433, China.
Email: ganzhongxue@fudan.edu.cn

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
Reinforcement Learning (DRL) provides a new solution for trajectory planning tasks in such conditions.\textsuperscript{6–8} It enables the robot to learn autonomously and plan a feasible trajectory in unstructured environment. The structure of trajectory planning with DRL is shown as Figure 1. “Trial and Error” is the central mechanism of DRL method, the agent can explore the possible motions according to the current state of work environment and the robot by maximizing the cumulative reward with an optimization strategy. Through the interaction of agent, reward function, and work environment, the robot can accomplish the trajectory planning task in unstructured environment.\textsuperscript{9–11}

The representative optimization strategies in DRL include Q-learning, DQN (Deep Q Network), SARSA (State Action Reward State Action), and the like.\textsuperscript{12–14} However, these methods cannot be directly used for trajectory planning task, as the spaces of output action generated by those methods are discrete, which cannot meet the need for trajectory planning task with continuous action spaces. To cope with this problem, Lillicrap et al.\textsuperscript{15} proposed an algorithm called DDPG (Deep Deterministic Strategy Gradient), by the nonlinear approximation, DDPG makes the output action space continuous. Furthermore, Tai et al.\textsuperscript{9} further improved the DDPG with a strategy of asynchronous execution. However, the performance of DDPG is restricted by the operation of experience replay. This shortcoming can be overcome by asynchronous update in A3C (Asynchronous Advantage Actors).\textsuperscript{16} The multi-threaded implementation in A3C also improves learning efficiency observably. However, A3C does not work well in complex environment due to the fixed learning rate, the performance of robustness is not satisfactory. To solve this problem, DPPO (Distributed Proximal Policy Optimization)\textsuperscript{17} was proposed, it introduces a penalty term, which can reduce the impact of unreasonable learning rate by providing a more reasonable update proportion. Above methods can solve the problem of trajectory planning to some extent. Nevertheless, randomness and blindness are still the major problems in DRL methods. When the agent faces an unstructured working environment with ban points, this problem will be more serious. Through previous work, we find that the kernel of this problem is reward function. Previous researches mainly focused on the innovation of optimization strategy but neglected the design of reward function. Most reward functions used in robot trajectory planning task are sparse reward functions. The value of sparse reward functions is zero everywhere, except for a few special places such as target or ban point.\textsuperscript{18} Sparse reward function always generates a good deal of ineffective explorations and tends to get trapped in locally optimal solution, which affects the efficiency of DRL method seriously.\textsuperscript{19–21} Therefore, this paper mainly focuses on construction of reward functions, the primary contributions are summarized as follows: (1) In consideration of the feature of trajectory planning task, this paper proposes two kinds of dense reward functions including azimuth reward function and aspiration reward function. Different from sparse reward function, dense reward function gives non-zero rewards most of the time. It provides much more feedback information after each action, thereby reducing the blindness of exploration in trajectory planning task. (2) Azimuth reward function is a results-oriented reward function, it prompts the agent to choose the action to get higher reward. Azimuth reward function mainly provides reasonable constraint in the exploration. According to characteristics of trajectory planning task, direction and distance are used to model azimuth reward function. Experiments proves that azimuth reward function can bring benefits to the convergence speed and robustness. (3) Aspiration reward function is process-oriented, it focuses more on the exploration process rather than the final result. In this paper, agent’s familiarity with the environment is defined as aspiration, which is the difference between predict the features and actual features. Aspiration reward function will stimulate the agent to explore unfamiliar areas, it is therefore capable of accelerating the exploration process and avoiding locally optimal solution. In order to predict the features felicitously, a novel feature extraction network SRU-HM is also proposed. With the help of SRU-HM, aspiration reward function can make better performance at a faster response.

Azimuth reward function

Target searching and ban point avoidance are the two goals in the trajectory planning task. Azimuth reward function, which is composed of direction reward function and distance reward function, is to provide
reasonable constricts for the agent from different perspectives. Section 2.1 and 2.2 explain the two reward functions respectively, section 2.3 introduces the implement of azimuth reward function based on the former.

**Direction reward function**

A challenge of trajectory planning in unstructured environment with ban points is to balance the target searching and ban point avoidance. Target searching aims to identify the shortest path while ban point avoidance requires security first. The two goals are even in opposition in some cases since the direction from rehabilitation robot to target and ban point are sometimes overlapped. As a consequence, it is necessary to propose a strategy for the agent to choose a reasonable direction. Direction reward function is proposed to take this duty. Inspired by Coulomb’s law, this paper regards the relative motion between target and end effector of rehabilitation robot as dissimilar charges attract each other. Similarly, when considering ban point avoidance, the relation between ban point and end effector can be seen as like charges repel to one another. Direction reward function is built as Figure 2, where \( \overrightarrow{RT} \) denotes the vector of attractive force from the target and \( \overrightarrow{RO} \) is the repellent vector to the ban point. \( \overrightarrow{RT} \) and \( \overrightarrow{RO} \) are described as formulated (1) and (2).

\[
\overrightarrow{RT} = \frac{Q_x Q_T}{D_{RT}} \frac{\overrightarrow{RT}}{|\overrightarrow{RT}|}
\]

(1)

\[
\overrightarrow{RO} = -\frac{Q_x Q_O}{D_{RO}} \frac{\overrightarrow{RO}}{|\overrightarrow{RO}|}
\]

(2)

Where \( D_{RT} \) is the relative distance from the end effector of rehabilitation robot to target and \( D_{RO} \) denotes the relative distance between the end effector of rehabilitation robot and ban point. In robot trajectory planning task, the attraction from target should be greater than the rejection of ban point in most instances. Otherwise, the robot may not be able to reach the target because of avoidance. In this paper, \( Q_x \) represents the amount of charge carried by different objects, the charge of target written as \( Q_T \), and the value of ban point is \( Q_O \), the ratio of target charge and ban point charge is recorded as \( e \). In the experiments, we find that if \( e \) is less than 1.4, the learning process may not be able to converge. When \( e \) is greater than 3, the avoidance may not work and the collision often occurs. Therefore, we set the ratio of two charges \( e \) to 2 to ensure the robot can complete the trajectory planning task while avoiding the ban points. \( \overrightarrow{RA} \) is the desired motion vector that can be computed by parallelogram law, \( \overrightarrow{RB} \) is the actual motion vector, the intersection angle of \( \overrightarrow{RA} \) and \( \overrightarrow{RB} \) is used to measure the similarity between actual motion vector and desired motion vector calculated by agent. In conclusion, direction reward function is calculated by formula (3), where \( K \) is the scaling factor, which is set to 0.74 to balance the value of direction reward function and distance reward function by aligning their extremums.

\[
R_{Direction} = K \frac{\overrightarrow{RA} \cdot \overrightarrow{RB}}{|| \overrightarrow{RA} || \times || \overrightarrow{RB} ||}
\]

(3)

When there are multiple ban points involved, each scene of ban point is calculated separately, and their results will be added to obtain the final direction reward function.

**Distance reward function**

Distance reward function is also constructed by both considering target searching and ban point avoidance. Therefore, distance reward function is made up of two parts. Ban point avoidance is a punitive element which is responsible for making the rehabilitation robot keep a safe distance from ban point. Target guidance provides positive incentives that navigates the rehabilitation robot to search the target.

**Ban point avoidance.** The characteristic of ban point avoidance is that the closer the robot moves to the ban point, the higher the negative reward value will be. However, if the relative distance is safe enough, ban point avoidance should not interfere with target guidance task. It’s clear that simple linear functions cannot
meet the demands. Gaussian function is used to model ban point avoidance as shown in formula (4), where \( D_{RO} \) denotes the relative distance between end effector of rehabilitation robot \( R \) and ban point \( O \) in Figure 2. The risk of collision increases if \( D_{RO} \) decreases, the agent will get more punishment in such condition. At the same time, the character of Gaussian function also ensures that ban point avoidance will not make too much impact when the relative distance is safe enough.

\[
R_{O-A}(D_{RO}) = -\frac{1}{\sqrt{2\pi}} e^{-\frac{D_{RO}^2}{2}} \tag{4}
\]

**Target searching.** Target searching is more of a positive motivation to motivate rehabilitation robot to arrive the target as quickly as possible. There are two cases that are shown in formula (5), where \( D_{RO} \) and \( D_{RT} \) are relative distances. When \( D_{RO} \) is less than \( D_{RT} \), a positive parameter compensation is required for \( R_{T-S} \) to ensure that agent can obtain a positive reward when the correct decision is made. In this paper, we set the value of \( \varphi \) to \( D_{RT}^2 - D_{RO}^2 \) at the initial position of the rehabilitation robot, which ensures that the result of \( R_{T-S} \) is 0 at the beginning. In another case, when \( D_{RO} \) is larger than \( D_{RT} \), the difference between \( D_{RO} \) and \( D_{RT} \) can describe the desired output directly, so no parameter compensation is required at this condition.

\[
R_{T-S}(D_{RO}, D_{RT}) = \begin{cases} D_{RO}^2 - D_{RT}^2 + \varphi, & D_{RO} \leq D_{RT} \\ D_{RO}^2 - D_{RT}^2, & D_{RO} > D_{RT} \end{cases} \tag{5}
\]

By combining ban point avoidance and target searching, we describe the position reward function as formula (6).

\[
R_{Distance}(D_{RO}, D_{RT}) = R_{O-A} + R_{T-S} \tag{6}
\]

**Implement of azimuth reward function**

In the actual trajectory planning task, distance and direction are both important factors should be considered comprehensively. However, the working environment of rehabilitation robot is intricate, which needs to consider the elements of both robot and the patient. As a consequence, the weights of the two items in azimuth reward function are always various in different scenarios. In this paper, we introduce a weight vector \( \lambda = [\lambda_{Distance}, \lambda_{Direction}] \) to build azimuth reward function. A ban point area in the rehabilitation robot’s workspace is divided into three parts, including safety, warning, and danger areas, as shown in Figure 3.

To improve learning efficiency as well as ensuring safety, \( \lambda \) is adjusted dynamically in different working areas. In safety area, distance reward function plays the leading role. In warning area, along with rehabilitation robot closing to the ban point, the weight of distance reward function decreases and direction reward function increases; at danger area, direction reward function takes charge. Specific adjustment strategy of \( \lambda \) is summarized in (7).

\[
\lambda = \begin{bmatrix} \lambda_{Distance} & \lambda_{Direction} \end{bmatrix}^T \begin{bmatrix} E \in \text{danger} \\ E \in \text{warning} \end{bmatrix} = \begin{bmatrix} D_{RO} - D_{RT} \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} E \in \text{safety} \tag{7} \]

where \( r_d \) and \( r_w \) are the radii of danger area and warning areas shown in Figure 3. Combining equations (5)–(7), the final expression of azimuth reward function is denoted as (8).

\[
R_{Azimuth} = \begin{bmatrix} \lambda_{Distance} \\ \lambda_{Direction} \end{bmatrix}^T \begin{bmatrix} R_{Distance}(D_{RO}, D_{RT}) \\ R_{Direction} \end{bmatrix} \tag{8}\]

**Aspiration reward function**

**Structure of aspiration reward function**

Locally optimal solution is a common problem which always perplexes DRL method. The reason is that most DRL methods only adopt utility reward functions. In this pattern, positive rewards will be given if the actions meet expectations. On the contrary, if the actions deviate from expectations, agent will get negative bonus. Although the agent can complete the exploration task with utility reward function in some conditions, but the trap of local optimal solution is usually unavoidable and the learning efficiency is often not satisfactory. To solve the problems, aspiration reward function is proposed. The idea of aspiration reward function is to increase the agent’s desire to explore the unfamiliar environment. The agent’s familiarity with environment...
will affect its strategy adjustment. Compared with the traditional mode, the agent with aspiration reward function and utility reward function is more reasonable, since it has higher learning efficiency and is more consistent with the human learning habit. In this paper, the aspiration reward for the agent is negatively related to its familiarity with current working environment.24

The structure of aspiration reward function is shown in Figure 4. The core idea is to regard aspiration as the accuracy of the agent’s prediction of status feature changes. Aspiration reward function is composed of a feature extractor and an SRU-HM neural network. The former is used for extracting status features and the later is responsible for feature prediction. The difference between extracted feature and predicted the status feature is used to calculate aspiration reward. Considering that we are calculating the aspiration reward in time note $t$, $S_t$ and $S_{t+1}$ represent the status information at time $t$ and $t+1$ which including relative distance, manipulator state, and environment state. Relative distance is the distance between the robot end-effector and target or ban point. Manipulator state includes the torque and spatial position of each joint. Environment state refers to the environmental change caused by the robot action, which mainly includes whether any part of the robot collides with ban points, and whether the robot end effector contacts with the target point. The operation of normalization filters $S_t$ and $S_{t+1}$ then stitch them into one-dimensional vectors. The status features $\Phi(S_t)$ and $\Phi(S_{t+1})$ are got by feature extractor. The input of SRU-HM contains $\Phi(S_t)$ and $a_t$, where $a_t$ represents the action made by agent at time $t$. $\Phi(S_{t+1})$ is predicted status feature made by SRU-HM through $\Phi(S_t)$ and $a_t$. Finally, aspiration reward is calculated by comparing the difference between practical feature $\Phi(S_{t+1})$ and predicted feature $\Phi(S_{t+1})$. Aspiration reward function is summarized as (9).

$$R_{\text{Aspiration}}(t) = \eta \frac{\|\Phi(S_{t+1}) - \Phi(S_{t+1})\|^2}{2}$$  \hspace{1cm} (9)$$

Where $\eta$ is scaling factor, which is responsible for adjusting the proportion of aspiration reward function in learning process.

**Recurrent neural network with hierarchical memory (SRU-HM)**

In previous works, researchers usually use fully connected networks in feature prediction.25,26 Fully connected networks adopt stacked structure which is easy to implement, but in practice, the number of layers in the network is difficult to determine since shallow network cannot predict status feature accurately and the deep network is difficult to train and time-consuming. It is a challenging to make suitable status feature prediction by using a relatively simple network structure. To cope with this problem, this paper proposes a recurrent neural network with hierarchical memory (SRU-HM). Compared with traditional stacked structure, this build-in memory mechanism can retain long-term historical information. Inner and outer layers in hierarchical recurrent neural networks are connected so the inner and outer memory cell information can access each other. The structure of SRU-HM is shown in Figure 5.

In the hidden layer of SRU-HM, the inner unit of SRU is embedded in the outer unit to build a layered network. The input information sent to inner unit from outer unit and then returns back after processed. The internal process of SRU-HM is shown in Figure 6. Where $\sigma$ is activation function Sigmoid and $\tanh$ is Hyperbolic Tangent. 1 and $-1$ are both weights. The symbol with the upper wave indicates the inner layer parameters. $C_t$ and $C_{t-1}$ represent the information
stored in the outer memory unit at the time note $t$ and
the previous respectively. $C_{t-1}$ denotes the memory
information stored in the inner memory unit at time
$t - 1$ and $h_i$ represents the output information of
the hidden layer at the current moment. When the inner
SRU memory unit accesses the outer, it uses standard
SRU gating mechanisms to transfer outer information
to inner memory cells selectively. At the same time,
internal memory is regulated in the same way to further
screen valid information. Compared with traditional
fully connected network, SRU-HM can store and pro-
cess information more effectively and make more rea-
sonable predictions. Compared to traditional SRU,
SRU-HM modified it with hierarchical memory
mechanism. It enables the long-term past information
to provide more complete information relevant to
current prediction, which is effective for balancing the
variance disturbance caused by aspiration reward function.
In terms of time cost, SRU-HM is also more advantageous than traditional recurrent neural network
(RNN). In the experiments, SRU-HM can efficiently
complete the prediction task in aspiration reward function.

**Implementation of reward functions in DRL**

In this part, we explain how to implement the presented
reward functions to the major DRL methods. In the
previous work, it can be found that DRL methods with
actor network and critic network (A–C frame) has
much better performance than the one using actor net-
work (A frame) or critic network (C frame) alone.
Therefore, this paper largely discusses the implementa-
tion and comparisons of reward function on methods
with A–C frame.

The learning process of DRL method with the pro-
posed reward functions is shown as Figure 7, it is com-
prised of four stages including initialization, action
selection, reward calculation, and network training. At
the first stage initialization, actor network $\mu(S|\theta^\mu)$,
critic network $\sigma(S,a|\theta^\sigma)$, and SRU-HM network
$\rho(S,a|\theta^\rho)$ are initialized randomly. $S$ represents the sta-
tus information and $a$ indicates the action. Actor net-
work is responsible for predicting the output action and
critic network takes charge of evaluating the quality of
the action. SRU-HM is used to predict the status informa-
tion. $\theta^\mu$, $\theta^\sigma$, and $\theta^\rho$ are the weights of actor network,
critic network, and SRU-HM respectively. In the stage
of action selection, by considering the evaluation from
critic network and status $S$, actor network predicts an
action and put it into effect. Reward calculation follows
the previous definition, which is the most important
part because it directly affects the judgment of critic
network, the training of network will also depend upon

![Figure 7. Diagram of DRL in A–C frame with proposed reward functions.](image-url)
the reward. The last stage is network training, which is the process of updating the parameters of actor network, critic network, and SRU-HM. In this stage, reward $R$, status $S$, and action $a$ need to be considered comprehensively. The goal of actor network is to plan actions with higher rewards, critic is to make appropriate evaluations, and SRU-HM is to make more accurate predictions based on the current environmental status. The training of the three networks is carried out simultaneously with the exploration process. Action selection, reward calculation, and network training will iterate until the networks converge. The whole process is summarized as Algorithm 1, where $M$ is the maximum episode and $T$ is the maximal training steps in each episode.

### Experiments and discussion

In this section, three sets of experiments are conducted to verify the performance of the proposed reward functions. Convergence rate, mean value, and standard deviation are selected as evaluation indicators. Convergence rate and mean value are chosen to test the learning efficiency, and standard deviation is for stability and robustness. In the experiments, the proposed reward functions are implemented to the state-of-the-art DRL methods Asynchronous Advantage Actor-Critic (A3C)\(^\text{16}\) and Distributed Proximal Policy Optimization (DPPO).\(^\text{17}\) Basic reward function is used for comparison. Basic reward function is a sparse reward function, it gives 0 in most cases, except for the robot reaching the target or ban point. In the first two experiments, azimuth reward function and aspiration reward function are put in use respectively, and the last set of experiments are conducted with the both reward functions. Simulation experiments are conducted in V-REP.\(^\text{27,28}\) In this paper, we simulated two working environments as shown in Figure 8, rehabilitation robot needs to reach the target point without touching any ban point to complete the trajectory planning task.

Every experiment will conduct for five times, the results are averaged in order to eliminate contingency. In the experiments the maximal reward for DRL method is set to 2000. If the accumulated reward value of the agent reaches 90% of the upper limit stably, trajectory planning is considered to be completed. The configuration used in the experiments is summarized in Table 1.

### Azimuth reward function

In this section, we apply the azimuth reward function to DPPO and A3C, and experimental results are summarized in Table 2. It can be seen that A3C and DPPO with azimuth reward function both perform better in convergence and robustness compared to traditional reward function basic. For A3C, convergence speed of A3C is accelerated by 18.6%--19.9%, and the promotion for DPPO is 24.5%--35.5%. In the aspect of mean

---

**Algorithm 1**: Trajectory planning algorithm with azimuth reward function and aspiration reward function.

**Require:**
- Status $S$.

**Ensure:**
- Action $a$.

1: Initialize Actor Network $\mu(S|\theta^a)$, Critic Network $\sigma(S,a|\theta^\pi)$, and SRU-HM Network $\rho(S,a|\theta^\rho)$
2: for episode = 1 to $M$ do
3: for $t$ = 1 to $T$ do
4: $a_t := \mu(S|\theta^a)$
5: Compute $R_{\text{azimuth}}$ 
6: Compute $R_{\text{aspiration}}$
7: $R = R_{\text{azimuth}} + R_{\text{aspiration}}$
8: Update weigh of Actor Network $\theta^a$
9: Update weigh of Critic Network $\theta^\pi$
10: Update weigh of SRU-HM Network $\theta^\rho$
11: end for
12: end for

---

**Table 1.** Configuration used in the experiments.

| Category | Details |
|----------|---------|
| System   | Ubuntu 18.04 |
| CPU      | Intel i9 9900 K |
| GPU      | NVIDIA GeForce RTX 1080 Ti 11G |
| RAM      | 64 GB |
| Environment | V-REP |
| Language | Python |

**Figure 8.** Simulation of rehabilitation robot trajectory planning task: Scene A has one ban point and Scene B with two. Scene A and B represent tasks of different complexity.
value, the two methods also have some advance by 5.2%–6.1%. The improvement in robustness is more significant, standard deviation of A3C and DPPO decrease by 32.5% on average. It can be seen that azimuth reward function not only speeds up the learning efficiency, but also increases the convergence stability of DRL method greatly. In the exploration, the role of azimuth reward function is to provide a global guidance and reasonable constraints for the agent, therefore, it can effectively reduce invalid exploration and improve efficiency. The reward curves of A3C and DPPO is visualized in Figure 9. In the process, the reward stays at 0 or even negative for some episodes at early stage of exploration. The reason for this is that rehabilitation robot may touch ban point during random exploration. By contrast, it shows that azimuth reward function can greatly shorten this stage to improve exploration efficiency. In the convergence phase, the curves of azimuth reward function are more stable as well.

### Aspiration reward function

In DRL methods, most reward function are result-oriented, but aspiration reward is quite different, it mainly focuses on the process rather result. Different from general reward functions that give external evaluation, aspiration reward function is more like the personality of the agent, which shows more interest to unfamiliar things. This determines that the improvement brought by aspiration reward function is mainly in the convergence speed. Table 2 and Figure 8 shown the results, convergence speed is improved by up to

### Table 2. Result of different reward functions R-F is short for Reward Function, A-A denotes Azimuth + Aspiration.

| Method | Scene | R-F        | Reward | Episode |
|--------|-------|------------|--------|---------|
| A3C    | A     | Basic      | 1822.7 ± 66.4 | 338     |
|        |       | Azimuth    | 1933.5 ± 46.1 | 275     |
|        |       | Aspiration | 1924.5 ± 75.7 | 279     |
|        |       | A-A        | 1962.2 ± 52.6 | 217     |
|        | B     | Basic      | 1835.7 ± 60.4 | 472     |
|        |       | Azimuth    | 1941.4 ± 39.4 | 378     |
|        |       | Aspiration | 1933.4 ± 74.5 | 389     |
|        |       | A-A        | 1968.4 ± 55.7 | 296     |
| DPPO   | A     | Basic      | 1837.7 ± 65.2 | 433     |
|        |       | Azimuth    | 1941.2 ± 44.1 | 279     |
|        |       | Aspiration | 1928.5 ± 73.2 | 268     |
|        |       | A-A        | 1959.4 ± 56.4 | 247     |
|        | B     | Basic      | 1851.5 ± 60.9 | 426     |
|        |       | Azimuth    | 1955.2 ± 39.2 | 322     |
|        |       | Aspiration | 1940.1 ± 69.8 | 326     |
|        |       | A-A        | 1970.6 ± 53.2 | 258     |

### Figure 9. Convergence curves of Different Reward Functions: The curve in blue is Basic, yellow is Azimuth, green is Aspiration, and red is A-A.
38.1%, and convergence in mean improves 5.5% as well. However, aspiration reward function has some negative influences on standard deviation, this phenomenon is consistent with the essence and original intention of aspiration reward function. It can be found in the Figure 8 that the benefit brought by aspiration is pretty obvious at the early stage of exploration. For curves where the reward is zero or negative, this part is significantly reduced in the method with aspiration reward function. In addition to accelerating the exploration, avoiding local optimal solution is also an advantage of aspiration reward function. Agent with basic reward function sometimes falls into a local optimal solution. Specifically, the reward does not increase within a certain period of episodes, but obviously the method hasn’t converged yet. The case in Figure 8-A is much more obvious. Aspiration reward function can avoid this problem in most cases.

Azimuth and aspiration reward function

In this section, azimuth and aspiration reward function (referred as A-A reward function for abbreviation hereinafter) work together. Results are plotted in Figure 8, as can be seen both A3C and DPPO with A-A reward function are superior to others in all cases. The convergence rate of A3C is increased by up to 37.2% compared to basic reward function, and this promotion is 39.4%–42.9% for DPPO in different Scene. Convergent mean value went up by an average of 171.3, this improvement is more distinct in the Scene with two ban points, this also shows that the proposed A-A reward function can effectively cope with complex scenarios. For standard deviation, the performance of robustness is also improved but slightly less than azimuth reward function due to aspiration. Considering convergence rate, convergent mean, and robustness comprehensively, the efficiency improvement brought by A-A reward function is significant. From the results of visualization, at the beginning of exploration, the curve of aspiration reward function is better than others, the desire to explore the unknown environment plays an important role at this stage. As the exploration progresses, agent becomes familiar with the working environment, by this time, azimuth reward function gradually showed its advantages, the curve of azimuth gradually overtakes aspiration. In addition, azimuth reward function also takes the duty of safety guarantee, ensuring the rehabilitation robot does not touch ban point. At this stage, aspiration reward function is mainly responsible for preventing the agent from falling into a local optimal solution. In convergence stage, aspiration reward function may cause little interference to standard deviation, but compared with the improvement in convergence performance it brings, it is completely negligible. Finally, taking a look at the performance of A3C and DPPO, DPPO performed better than A3C when using basic reward function in Scene B. The reason is the learning rate of A3C is fixed, and DPPO introduces a penalty mechanism for optimization, so it performs better in complex environments. However, when using the proposed A-A reward function, the performance of the two methods is equivalent in simple environment Scene A, DPPO performs slightly better in complex environment Scene B. This shows that the reward function proposed can also make up for some of the flaws in the optimization method.

Conclusion

To cope with the inefficiency and blindness of rehabilitation robot trajectory planning task with DRL methods, this paper puts forward two new dense reward functions including azimuth reward function and aspiration reward function. The former can provide rational constraints for the agent during exploration, while the latter is capable to accelerate exploration and avoid locally optimal solution. To improve the efficiency of aspiration reward function, a new feature prediction network SRU-HM is also proposed. Experimental results demonstrate that major methods with the proposed reward functions can improve the convergence rate and trajectory planning quality dramatically with respect to the accuracy and robustness. In future studies, we will try to conduct the multi-agent exploration experiment on the actual rehabilitation robots. The further research of SRU-HM is also a major work, in addition to reward calculation, we are going to make SRU-HM be a part of the brain of the agent and play a more important role.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103).

ORCID iDs

Xusheng Wang https://orcid.org/0000-0003-3145-3830
Jiexin Xie https://orcid.org/0000-0002-3095-3086

References

1. Carrera I, Moreno HA, Saltarén R, et al. Road: domestic assistant and rehabilitation robot. Med Biol Eng Comput 2011; 49: 1201–1211.
2. Yubo Z, Zixi W, Linhong J, et al. The clinical application of the upper extremity compound movements rehabilitation training robot. In: 9th international conference on rehabilitation robotics (ICORR), Chicago, IL, USA, 28 June–1 July 2005, pp.91–94. New York, NY: IEEE.

3. Lu EC, Wang RH, Hebert D, et al. The development of an upper limb stroke rehabilitation robot: identification of clinical practices and design requirements through a survey of therapists. Disabil Rehabil Assist Technol 2011; 6: 420–431.

4. Yang C, Li Z and Li J. Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum vehicle models. IEEE Trans Cybern 2013; 43: 24–36.

5. Menasri R, Nakib A, Daachi B, et al. A trajectory planning of redundant manipulators based on bilevel optimization. Appl Math Comput 2015; 250: 934–947.

6. Katyal K, Wang I, Burlina P, et al. Leveraging deep reinforcement learning for reaching robotic tasks. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops (CVPR), Honolulu, Hawaii, USA, 21–26 July 2017, pp.18–19. New York, NY: IEEE.

7. Lei X, Zhang Z and Dong P. Dynamic path planning of unknown environment based on deep reinforcement learning. J Robot 2018; 2018: 1–10.

8. Kahn G, Villafior A, Ding B, et al. Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In: 2018 IEEE international conference on robotics and automation (ICRA), Brisbane, QLD, Australia, 21–26 May 2018, pp.5129–5136. New York, NY: IEEE.

9. Tai L, Paolo G and Liu M. Virtual-to-real deep reinforcement learning: continuous control of mobile robots for mapless navigation. In: 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS), Vancouver, BC, Canada, 24–28 September 2017, pp.31–36. New York, NY: IEEE.

10. Chen X, Ghadirzadeh A, Folkesson J, et al. Deep reinforcement learning to acquire navigation skills for wheel-legged robots in complex environments. In: 2018 IEEE/RSJ international conference on intelligent robots and systems (IROS), Madrid, Spain, 1–5 October 2018, pp.3110–3116. New York, NY: IEEE.

11. Wu YH, Yu ZC, Li CY, et al. Reinforcement learning in clinical practices and design requirements through a survey of therapists. Disabil Rehabil Assist Technol 2011; 6: 420–431.

12. Mnih V, Badia AP, Mirza M, et al. Asynchronous methods for deep reinforcement learning. In: 2016 International conference on machine learning (ICML), Anaheim, CA, USA, 18–20 December 2016, pp.1928–1937. Princeton, NJ: IMLS.

13. Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.

14. Smart WD and Kaelbling LP. Effective reinforcement learning for mobile robots. In: 2002 IEEE international conference on robotics and automation (ICRA), Washington, DC, USA, 11–15 May 2002, pp.3404–3410. New York, NY: IEEE.

15. Lillicrap TP, Hunt JJ, Pritzel A, et al. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971, 2015.

16. Mnih V, Badia AP, Mirza M, et al. Asynchronous methods for deep reinforcement learning. In: 2016 International conference on machine learning (ICML), Anaheim, CA, USA, 18–20 December 2016, pp.1928–1937. Princeton, NJ: IMLS.

17. Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.

18. Smart WD and Kaelbling LP. Effective reinforcement learning for mobile robots. In: 2002 IEEE international conference on robotics and automation (ICRA), Washington, DC, USA, 11–15 May 2002, pp.3404–3410. New York, NY: IEEE.

19. Badnava B and Mozayani N. A new potential-based reward shaping for reinforcement learning agent. arXiv preprint arXiv:1902.06239, 2019.

20. Nair A, McGrew B, Andrychowicz M, et al. Overcoming exploration in reinforcement learning with demonstrations. In: 2018 IEEE international conference on robotics and automation (ICRA), Brisbane, QLD, Australia, 21–26 May 2018, pp.6292–6299. New York, NY: IEEE.

21. Marti B. Automatic shaping and decomposition of reward functions. In: Proceedings of the 24th international conference on machine learning (ICML), Corvallis, OR, USA, 20–24 June 2007, pp.601–608. Princeton, NJ: IMLS.

22. Transeth AA, Leine RI, Glocker C, et al. Snake robot obstacle-aided locomotion: modeling, simulations, and experiments. IEEE Trans Robot 2008; 24: 88–104.

23. Raja P and Pugazhenthi S. Optimal path planning of mobile robots: a review. Int J Phys Sci 2012; 7: 1314–1320.

24. Pathak D, Agrawal P, Efros AA, et al. Curiosity-driven exploration by self-supervised prediction. In: 2017 international conference on machine learning (ICML), Vienna, Austria, 18–24 July 2017, pp.2778–2787. Princeton, NJ: IMLS.

25. Burda Y, Edwards H, Pathak D, et al. Large-scale study of curiosity-driven learning. arXiv preprint arXiv:1808.04355, 2018.

26. Houthooft R, Chen X, Duan Y, et al. Curiosity-driven exploration in deep reinforcement learning via Bayesian neural networks. arXiv preprint arXiv:1605.09674v 1, 2016.

27. Rohmer E, Singh SP and Freese M. V-rep: A versatile and scalable robot simulation framework. In 2013 IEEE/RSJ international conference on intelligent robots and systems (IROS), Tokyo, Japan, 3–8 November 2013, pp.1321–1326. New York, NY: IEEE.

28. Freese M, Singh S, Ozaki F, et al. Virtual robot experimentation platform V-REP: a versatile 3D robot simulator. In: 2010 international conference on simulation, modeling, and programming for autonomous robots (ICSMPAR), Berlin, Heidelberg, Germany, 15 November–18 November 2010, pp.51–62. Berlin: Springer.