Path Planning with Autonomous Obstacle Avoidance Using Reinforcement Learning for Six-axis Arms

Yinsen Jia
School of Electric and Automation Engineering
Nanjing Normal University, Nanjing, China
email: 1078325798@qq.com

Yichen Li
School of Management and Engineering
Nanjing University, Nanjing, China
email: liyichen7ban@163.com

Bo Xin
School of Management and Engineering
Nanjing University, Nanjing, China
email: xinbo@nju.edu.cn

Chunlin Chen
School of Management and Engineering
Nanjing University, Nanjing, China
email: clchen@nju.edu.cn

***Abstract*—In this paper, a strategy of path planning for autonomous obstacle avoidance using reinforcement learning for six-axis arms is proposed. This strategy gives priority to planning the obstacle avoidance path for the terminal of the mechanical arm, and then uses the calculated terminal path to plan the poses of the mechanical arm. For the points on the terminal path that the mechanical arm cannot avoid obstacles within the limit of the safe distance, this strategy will record these points as new obstacles and plan a new obstacle avoidance path for the terminal of mechanical arm. The above process is accelerated by the assisted learning strategies and looped until the correct path being calculated. The method proposed in this paper has been applied to a six-axis mechanical arm, and the simulation results show that this method can effectively plan an optimal path and poses for the mechanical arm.**

Keywords—Q-learning, obstacle avoidance, path planning, six-axis arms, assisted learning strategies.

I. INTRODUCTION

Wide applications of robots have greatly promoted the development of industrial production. However, with the increasing complexity of the working environment, pre-programmed fixed orbit manipulators cannot meet the needs of various applications, and the problem of autonomous path planning and obstacle avoidance has attracted more and more attention [1-4] in robotics communities. As shown in Fig.1, the black area are obstacles and the yellow ball is the target.

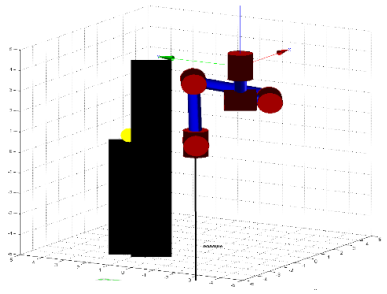

Fig. 1. A six-axis robotic arm in the environment with obstacles

The artificial potential field (APF)-based industrial robot station conversion obstacle avoidance system has been proposed [5] with external sensors to control the movement trajectory of the mechanical arm, which performs well in real-time planning. But when in cluttered environments, it is easy to get into oscillations and cannot reach the goal. In Ref. [6], obstacle-avoidance is implemented by adding a penalty function to the goal function, and using genetic algorithms to optimize the movement of the robot arm and reduce its energy consumption. Nevertheless, the main focus of this method is to reduce the energy redundancy of the manipulator, and it does not pay much attention to the obstacles avoidance in complex environment and the path planning of the mechanical arm. RBF (Radial basis function) neural networks and quadratic programming techniques have also been used in the robot obstacle avoidance problem [7]. However, it lacks of actual control examples and is difficult to modify the corresponding parameters of practical control problems. Recently, deep reinforcement learning has been investigated for collision avoidance of mobile robots using double deep Q-learning [8] and a cooperative collision avoidance strategy for multi-vehicle systems using reinforcement learning have been proposed [9], which only the terminal positions of the robots are considered for obstacle avoidance. To solve high-dimensional multi-constraint path planning problems, such as Probabilistic Roadmaps [10], it is necessary to keep re-planning when the initial position changes, which is time-consuming, and sometimes is impossible to find a feasible solution.

To provide a practical and efficient learning based obstacle avoiding method, in this paper, we propose a strategy of path planning with autonomous obstacle avoidance using a standard reinforcement learning algorithm (i.e., Q-learning) for six-axis arms. Simulation results show that the proposed approach can effectively make the mechanical arm avoid obstacles to reach the target point, and reduce the operation time with the assisted learning strategy.

II. BACKGROUND

A. Reinforcement Learning

In recent years, reinforcement learning (RL) such as Q-learning [11] has been widely used in operations research, control engineering, analysis and prediction. RL is based on the framework of Markov Decision Processes (MDP), which is a tuple of $< S, A, T, R, \gamma >$, where S is the set of states, A is the set of actions, T: $S\times A\times S\rightarrow[0, 1]$ is state transition probability, R: $S\times A\rightarrow R$ is reward function, and γ is the discounting factor. A policy is defined as a function π: $S\times A\rightarrow$ [0, 1], which represents a probability distribution regarding the state-action pairs and $\sum_{a\in A}\pi(a\,|\,s)=1, \forall s\in S$. The goal of RL is to find an optimal policy $\pi*$ to can get the best long-term return $J(\pi)$ as

$$J(\pi) = E_{\tau\sim\pi(\tau)}[r(\tau)] = E_{\tau\sim\pi(\tau)}[\sum_{t=0}^{\infty}\gamma^t r_t], \quad (1)$$

where $\tau = (s_0, a_0, s_1, a_1, ...)$ is the learning episode, $\pi(\tau) = p(s_0)\prod_{t=0}^{\infty}\pi(a_t \mid s_t)p(s_{t+1} \mid s_t, a_t)$, p is probability, r_t is the instant reward received when executing action a_t in state s_t.

B. Reinforcement learning for robot manipulation

In the field of mechanical arm control, the application of reinforcement learning is more and more prevalent in recent years. A line-grasping control strategy for deicing robot based on Q-learning has been proposed to recognize obstacles by an image processing unit and accomplish the line grab control of the de-icing robot very well [12]. An improved Q-Learning method with back propagation (BP) neutral network has been proposed to replace the Q table of traditional Q-learning and adjusts the Q value and the weight of the network by action reward, which has a good result in robot football competition and greatly improves the learning efficiency [13]. A task-parametrized assistive strategy with RL for exoskeleton robots control has been proposed to reduce the sampling cost and improves the control performance by adopting an auxiliary strategy [14]. RL is also applied to the learning behavior of humanoid robots and a strategy of model-based Q-learning for intelligent humanoid robots has been proposed in [15].

III. PATH PLANNING WITH OBSTACLE AVOIDANCE

In this section, the framework of the proposed strategy is first introduced and then the detailed methods of path planning, obstacle avoidance are given.

A. Framework

The approach of path planning and obstacle avoidance using reinforcement learning can greatly improve the efficiency of the mechanical arm system, and the flow chart of the proposed strategy is shown as in Fig.2.

First, input initial obstacles and the target point and initialize the mechanical arm to start a learning process. Using the distance assisted learning strategy to encourage the terminal of the robot arm to move towards the target with collision detecting. If collision exists in this process, the terminal of the mechanical arm will move back to the previous position and the encouraging detour strategy will be triggered. After Q-value updating, it will start a new learning process. If no collision exists in this process and Q-value has not converged, the learning episodes η will plus step value patch, and after updating Q-value, it will start a new learning process.

Second, if the Q-values have converged and the path planning has been completed, the results will be input to the mechanical arm and drive the whole mechanical arm to move. If no collision exists in this process, the current set of the inverse solution angles will be recorded and judge whether the mechanical arm pose planning for the all points in place are completed. If it is not completed, the current mechanical arm pose will be updated. If it is completed, the final angle matrix pose will be output. On the other hand, if collision exists during the process, a new set of joint angular solutions or a new terminal pose will be selected to control the mechanical arm to move again until no collision exists.

Finally, if all the selective terminal positions and joint angular solutions for the current point in calculated path cannot make robotic arm pass, this point will be recorded as a new obstacle, and the mechanical arm will be reset for a new learning process.

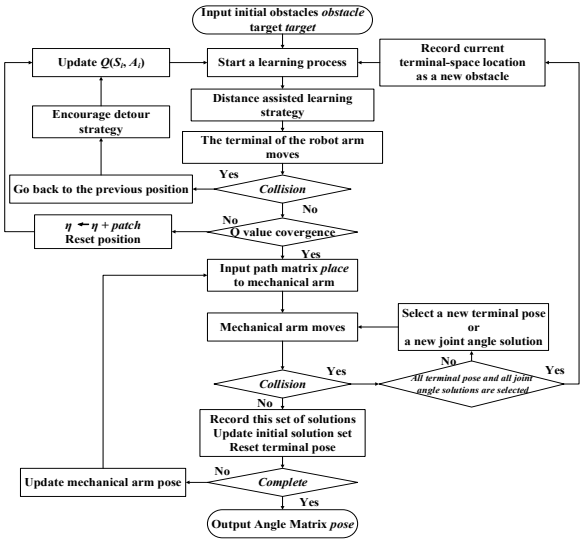

Fig. 2. Flow chart of the proposed strategy

B. Path planning

The path planning process for the terminal of the mechanical arm consists of two steps, i.e., the basic path planning strategy and the auxiliary learning strategy. The basic path planning strategy ensures that the end of the mechanical arm can successfully avoid obstacles and reach the target, and the auxiliary learning strategy accelerates this process and improves the computational efficiency.

1) Basic path planning strategy

States and actions. For the path planning process using *Q*-learning, initially, the state set *S* is defined as the states of the coordinates of the positions for the terminal of the manipulator

$$S = \begin{cases} [x_1, y_1, z_1] \\ [x_2, y_2, z_2] \\ [x_3, y_3, z_3] \\ \vdots \\ [x_n, y_n, z_n] \end{cases}, \quad (2)$$

where *x*, *y* and *z* are the *X*-axis, *Y*-axis and *Z*-axis coordinates of a point in the working space, and *n* is the total number of points reached by the mechanical arm.

The action space is divided into six actions as

$$A = \{+\Delta x, -\Delta x, +\Delta y, -\Delta y, +\Delta z, -\Delta z\}, \quad (3)$$

where *A* represents the action set to be taken for the terminal of the manipulator in the next step, $+\Delta x$, $+\Delta y$, $+\Delta z$ are that *X, Y, Z* direction increase by Δx, Δy, Δz, *respectively.*

Reward functions. The reward function is set up to get feedback after the action is executed, and it is a key step to control the terminal of the mechanical arm to avoid obstacles and to plan a better path.

The distance between the position of the terminal of mechanical arm and the obstacle can be defined as

$$d_t = \sqrt{(x_i - x_{obs})^2 + (y_i - y_{obs})^2 + (z_i - z_{obs})^2}, \quad (4)$$

where d_t is the distance between the terminal of the mechanical arm and the center of the obstacle, (x_i, y_i, z_i) is the position coordinate of the terminal at a certain moment, (x_{obs}, y_{obs}, z_{obs}) is the central coordinate of an obstacle.

Set the specified safe distance as d_s, the reward function can be expressed as

$$R(S_i, A_i) = \begin{cases} r & d_p > d_s, (x_i, y_i, z_i) = (x_{tag}, y_{tag}, z_{tag}) \\ -p & d_p > d_s, (x_i, y_i, z_i) \neq (x_{tag}, y_{tag}, z_{tag}) \\ -P & d_p \leq d_s \end{cases}, \quad (5)$$

where r is the reward value for the terminal of the robotic arm reaching the target, d_p is the current distance between the obstacles with the terminal, $-p$ is the punishment when the terminal of the robotic arm neither touches the obstacles nor gets to the target, $-P$ is the punishment when the terminal of robotic arm touches the obstacles. (x_{tag},y_{tag},z_{tag}) is the central coordinate of the target.

In Q-learning, the one-step update of Q value is

$$Q(S_i, A_i) \longleftarrow Q(S_i, A_i) + \alpha[R(S_i, A_i) + \gamma MAX_{A_{i+1}} Q(S_{i+1}, A_{i+1}) - Q(S_i, A_i)], \quad (6)$$

where α is the learning rate, γ is the discount factor.

Action selection strategy. The action selection strategy is the basis for selecting the next action to be executed for the terminal of the mechanical arm. We choose the ε-greedy strategy, and the details are listed as follows

$$A_c = \begin{cases} random(A) & 0 \leq \varepsilon_s \leq \varepsilon \\ A_{\max} & \varepsilon < \varepsilon_s \leq 1 \end{cases}$$
$$Q(S_i, A_{\max}) = MAX_{A_i} Q(S_i, A_i), \quad (7)$$
$$\varepsilon = \begin{cases} 0.1\text{-}\dfrac{0.1k}{M} & \eta \leq M \\ 0 & \eta > M \end{cases}$$

where A_c is the selected action, $random(A)$ represents randomly selected actions in the action state set A, ε_s is a random number between [0,1], A_{max} is the action to obtain the maximum Q value in the current state S_i, η is the actual times of learning, M is specified times of convergence.

Through the above path planning using Q-learning, a path matrix can be obtained as

$$place = \begin{cases} [x_{f1}, y_{f1}, z_{f1}] \\ [x_{f2}, y_{f2}, z_{f2}] \\ [x_{f3}, y_{f3}, z_{f3}] \\ \vdots \\ [x_{fall}, y_{fall}, z_{fall}] \end{cases}, \quad (8)$$

where (x_{fi},y_{fi},z_{fi}) represents the spatial coordinates of a certain point in the final path, (x_{fall},y_{fall},z_{fall}) is the spatial coordinates of the last point in the final path.

2) Assisted learning strategy

Since the state space S tends to be very large in three-dimensional space, it will take a long time if only the basic path planning strategy is used, so the auxiliary learning strategy is designed to accelerate the path planning process.

Detour Encouraging. When the terminal of the mechanical arm encounters an obstacle, the case that mechanical arm continues to touch the obstacle at next time is not allowed, and the case that mechanical arm moves in the opposite direction to move away from the obstacle is not desirable. Therefore, utilizing extra punishment in the reward function will encourage the terminal to bypass obstacles and to avoid these two cases

$$R(S_i, A_i) = \begin{cases} R(S_i, A_i) - p_e & A_i = A_{obs} \\ R(S_i, A_i) - p_o & A_i = -A_{obs} \end{cases}, \quad (9)$$

where P_e is the extra punishment that if the last action hits an obstacle, the next action still hits an obstacle, P_o is the additional penalty when the last action hits an obstacle and the next action moves in the opposite direction. A_{obs} is the action of the terminal encountering an obstacle, $-A_{obs}$ is the opposite action of encountering an obstacle.

Assisted learning using Euclidean distance. By using the fixed-valued linear distance between the initial point and the target point, the real-time distance between the terminal of mechanical arm and the target is compared with the fixed-valued linear distance, and the reward was given by the difference to encourage the terminal to move towards the target point and improve the learning efficiency. We have

$$d_o = \sqrt{(x_{tag} - x_o)^2 + (y_{tag} - y_o)^2 + (z_{tag} - z_o)^2}$$
$$d_s = \sqrt{(x_{tag} - x_i)^2 + (y_{tag} - y_i)^2 + (z_{tag} - z_i)^2}, \quad (10)$$
$$R(S_{i+1}, A_{i+1}) = R(S_{i+1}, A_{i+1}) + (d_o - d_s) * r_e$$

where d_o is the Euclidean distance between the target point and the initial point of the terminal, (x_o, y_o, z_o) is the coordinates of the initial position of the terminal, d_s is the real-time distance between the target point and the terminal, r_e is the reward assigned by the distance-assisted learning strategy.

With this strategy, the closer between the terminal of the mechanical arm and the target, the more rewards it will get. When the initial distance is exceeded by the real-time distance, the value of rewards will become a penalty.

C. Obstacle avoiding

To make the mechanical arm avoid obstacles autonomously, it is far from enough to merely plan the position of the terminal of the mechanical arm. Even if the terminal can avoid obstacles, the joints of the mechanical arm may encounter obstacles when moving. Therefore, further posture planning of the mechanical arm is necessary to get an obstacle-free path for the whole mechanical arm.

A six-axis manipulator can be simplified as six joints and their connections. However, because of the large offset between 3th joint and 4th joint of the six-axis mechanical arm that used in this paper, an assisted joint was added between the 3th joint and 4th joint. Using the coordinates of seven joints and their connections, as shown in Fig.3, the shortest distance between the center of the obstacle and the joints of the manipulator and the connection between the center of the obstacle and the joints can be calculated by

$$w = \frac{\overrightarrow{KO} \cdot \overrightarrow{KL}}{\left|\overrightarrow{KL}\right|} \qquad d_{zv} = \begin{cases} \left|\overrightarrow{KO}\right| & w \leq 0 \\ \left|\overrightarrow{JO}\right| & 0 < w < 1 \\ \left|\overrightarrow{LO}\right| & w \geq 1 \end{cases}, \quad (11)$$

$$d_b = \sqrt{(x_{bi} - x_{obs})^2 + (y_{bi} - y_{obs})^2 + (z_{bi} - z_{obs})^2}$$

where w is the judgement of the shortest path, K and L are the joint coordinates of the two ends of a certain segment of the mechanical arm, O is the center of the obstacle, J is the foot point from O to AB, d_{zv} is the shortest distance from the obstacle to the v^{th} segment of the mechanical arm, d_b is the distance between the b-joint of the mechanical arm and the center of the obstacle, (x_{bi}, y_{bi}, z_{bi}) is the coordinates of the b-joint of the mechanical arm in space at a certain moment, $(x_{obs}, y_{obs}, z_{obs})$ is the central coordinates of an obstacle in space.

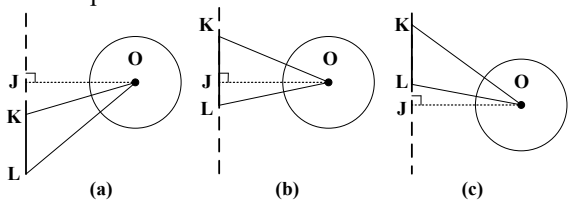

Fig. 3. The shortest distance between mechanical arm and obstacle

Algorithm 1 Autonomous obstacle avoidance

Input: Terminal path matrix *place*
Output: Final angle matrix *pose*
Initialize: Initial robot arm pose Angle set: β_0; $e = 1$
While $e \leqslant all$ **do**
  $Sqtraj \leftarrow inverse(rtx, rty, rtz, x_{fe}, y_{fe}, z_{fe})$
  Initialize $l = 1$
  While $l \leq 8$ **do**
    Choose β_l from *Sqtraj*
    Interpolate between β_0 and β_l
    Input to mechanical arm to move
    If *collision* **then**
      $l \leftarrow l + 1$
    else
      record β_l in *pose*
      $\beta_0 \leftarrow \beta_l$
      $e \leftarrow e + 1$
    end
  end
  If $l > 8$ **then**
    If *all terminal pose angle are selected* **then**
      Record (x_{fe}, y_{fe}, z_{fe}) as an new obstacle
      Reset mechanical arm
      Initialize $e = 1$
      Prepare for planning a new path
    else
      Change n_d to select a new terminal pose angle
    end
  end
end

For the obstacle avoidance of the mechanical arm, not only the position of the terminal should be specified, but also the pose angle of the terminal should be selected. The pose angle of the terminal can be discretized as

$$rt_{x,y,z} = \frac{2\pi}{n_d}, n_d \in N^*, \tag{12}$$

where (rt_x, rt_y, rt_z) represents the pose angles of the X, Y and Z axes of the terminal of the mechanical arm, respectively.

For the six-axis manipulator, there are at most eight sets of inverse solution angles for each accessible terminal position under the condition of specifying the pose and spatial position coordinates

$$\begin{aligned} inverse(rt_x, rt_y, rt_z, x_i, y_i, z_i) &= \{\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7, \beta_8\} \\ \beta_l &= \{\theta_{l1}, \theta_{l2}, \theta_{l3}, \theta_{l4}, \theta_{l5}, \theta_{l6}\} \end{aligned}, \tag{13}$$

where *inverse* is the inverse solution function, (x_i, y_i, z_i) represents the coordinates of the terminal, β_l is the solution set of joint angles of a set of mechanical arms, θ_{ln} is the n^{th} joint Angle.

All the coordinates are found to the feasible solution set of six-joint angular solution respectively β_f, and the set *pose* is the final solution set of the kinematic angle of the mechanical arm which is composed of all β_{fl}

$$pose_{(x_{f1}, y_{f1}, z_{f1}) \to (x_{fall}, y_{fall}, z_{fall})} = \{\beta_{f1}, \beta_{f2}, \cdots \quad , \tag{14}$$

where (x_{f1}, y_{f1}, z_{f1}) is the first group coordinate, (x_{f1}, y_{f1}, z_{f1}) is the final group coordinate, β_{fl} is the feasible solution set of six-joint angular solution.

The algorithm of the autonomous obstacle avoidance is shown as in ***Algorithm 1***.

D. Integrated algorithm

The integrated algorithm is given in ***Algorithm 2*** for the path planning with obstacle avoidance.

Algorithm 2 Integrated algorithm

Input: Target: *target*; Initial obstacles: *obstacle*;
Output: Angle matrix *pose*
Initialize: Learning policy ε-greedy;
While *Not complete* **do**
  While *Not converged* **do**
    Initialize $i = 1$, state S_i
    While S_i *is not target* **do**
      Select A_i using ε-greedy
      Observe $R(S_i, A_i)$ and update $R(S_i, A_i)$ with assisted learning strategy
      $$Q(S_i, A_i) \longleftarrow Q(S_i, A_i) + \alpha[R(S_i, A_i) + \gamma MAX_{A_{i+1}} Q(S_{i+1}, A_{i+1}) - Q(S_i, A_i)]$$
      $i \leftarrow i + 1$
    end
  end
  Output terminal path matrix *place*
  pose ← Final angle matrix using ***Algorithm 1***
end

As shown in ***Algorithm 2***, after the Q-learning for path planning converges, the terminal path matrix will be output to the mechanical arm, and ***Algorithm 1*** will be used to plan the pose of the mechanical arm and output the final pose of the angle matrix.

IV. EXPERIMENTS

In this section, several groups of simulated experiments are carried out to test the proposed approach.

A. Cases with simple obstacles

As shown in Fig.4, the starting point of the terminal of the mechanical arm is the red point, the black area is the superimposed cube obstacles, and the yellow point is the target point. Simulation parameters are shown as in Table 1. The D-H parameters used by the six-axis mechanical arm are shown in Table 2. Because there is the rod length and link offset between 3rd and 4th mechanical arm joints, a point is added between the two joints. It is used to calculate

the space coordinates between the 3rd and 4th mechanical arm joints, ensuring the accuracy of collision detection.

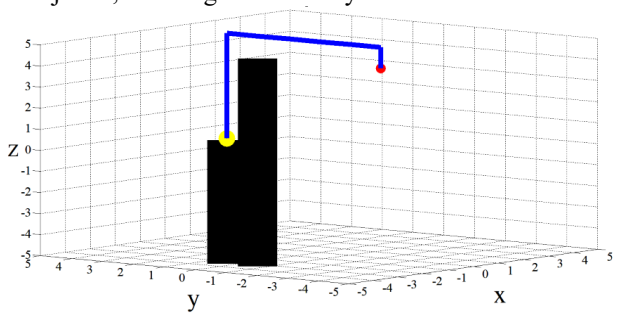

Fig. 4. The environment with simple obstacles

Table 1 Simulation parameter 1

Simulation parameter			
Parameter name	***Value***	***Parameter name***	***Value***
γ	0.9	d_p	0.76
α	0.05	d_s	0.96
η_{set}	10000	P_0	10000
ε	0.1	P_e	230000
Δx	1	(x_0,y_0,z_0)	(0,-2,4)
Δy	1	r	20000
Δz	1	r_e	40
$(x_{tag}, y_{tag}, z_{tag})$	(0,3,0)	β_0°	(57.9948,-10.1070,-132.9377, 53.0502,90.0002,32.0054)
p	15000	t	50
P	8000	M	9900
$rt^{\circ}_{x,y,z}$	{0,45,90,135,180,225,270,315}		
$(x_{obs}, y_{obs}, z_{obs})$	[(0,1,0);(0,3,0);(1,1,0); (-1,1,0);(-1,2,1);(-1,3,1);(-1,3,0); (0,1,1);(0,3,1);(1,1,1);(-1,1,1);(0,1,-1);(0,3,-1);(1,1,-1); (1,2,-1);(1,3,-1);(-1,1,-1);(-1,2,-1);(-1,3,-1);(0,2,-1)]		

Table 2 Six-axis Robotic Arm D-H parameter

Six-axis Robotic Arm D-H parameter				
Axis	***Rod rotation angle***	***Rod length***	***Connecting rod offset***	***Axis rotation angle***
1	***0***	***0***	***0***	***0***
2	***90°***	***0***	***0***	***0***
3	***0***	***-300***	***0***	***0***
4	***0***	***-300***	***106***	***0***
5	***90°***	***0***	***114***	***0***
6	***-90°***	***0***	***167***	***0***
***Auxiliary*(3.5)**	***0***	***-300***	***0***	***0***

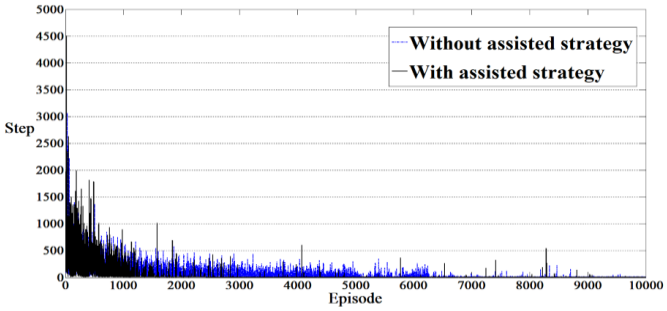

Fig. 5. Learning performance with simple obstacles.

As shown in Fig.4, the blue path is the initial path for the terminal of mechanical arm to avoid obstacles. The steps of all the episodes in the learning process are shown as in Fig.5. The steps of learning episodes without assisted learning strategies, which are blue line, are generally higher than those with assisted learning strategies, which are black line. The time of learning without assisted learning strategies is 2151.219s, and that of the learning with assisted learning strategies is 1268.14s. These results show that the assisted learning strategies efficiently speedup the learning process.

The movement process of the mechanical arm is shown as in Fig.6. It is clear that the mechanical arm, which is limited by the set safe distance, can avoid obstacles successfully and reach the target point safely.

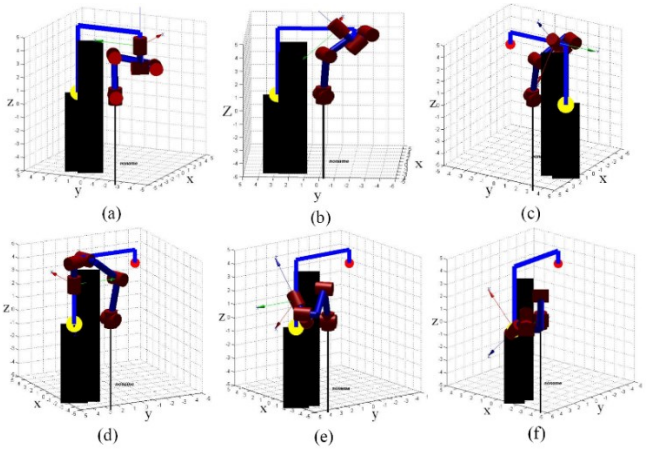

Fig. 6. The action of the 6-axies robot arm with simple obstacles

B. *Cases with complex obstacles*

The environment with complexed obstacles are shown as in Fig.7.

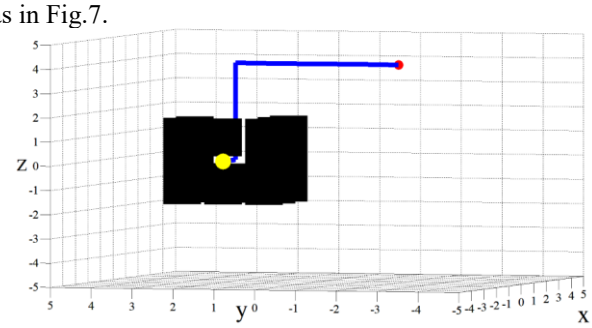

Fig. 7. The environment with complex obstacles

Table 3 Simulation parameters 2

Simulation parameter	
Parameter name	***Value***
d_p	0.85
d_s	0.95
$(x_{obs}, y_{obs}, z_{obs})$	[(0,1,0);(0,3,0);(1,1,0);(-1,1,0);(-1,2,1);(-1,3,1);(-1,3,0); (0,1,1);(0,3,1);(1,1,1);(-1,1,1);(0,1,-1);(0,3,-1);(1,1,-1); (1,2,-1);(1,3-1);(-1,1,-1);(-1,2,-1);(-1,3,-1);(0,2,-1)]

The parameter settings are shown as in Table3. Initial simulation result of terminal path is shown in Fig.7. The learning performance is shown as in Fig.8. The time of learning without assisted learning strategies is 1661.722s, and that with assisted learning strategies is 725.996s.

However, during the process of the pose planning of the mechanical arm, it shows that a point in the calculated path could not meet the requirements of safe distance for mechanical arm, so the path planning was re-carried out. The results are shown as in Fig.9. The movement process of the mechanical arm is shown as in Fig.10 and the initial pose is shown in Fig.10(a). In the whole process, it can be seen that the mechanical arm, which limited by the safe distance, can avoid obstacles successfully and reach the target point safely.

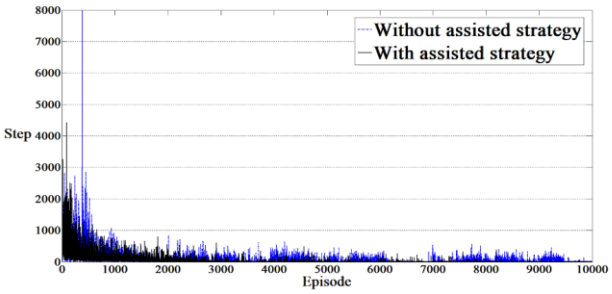

Fig. 8. Learning performance with complexed obstacles.

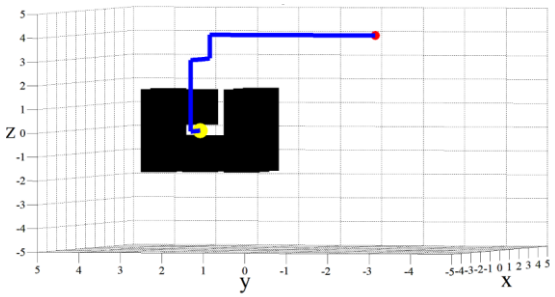

Fig. 9. Redesigned results of the terminal path of the mechanical arm.

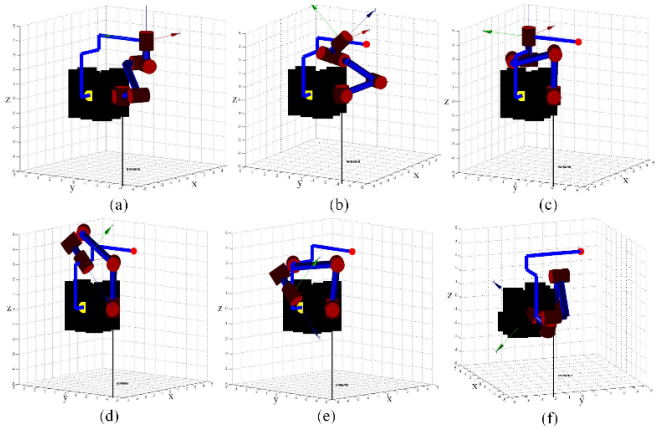

Fig. 10. The action of the 6-axies robot arm with complexed obstacles

V. Conclusion

In this paper, a strategy of path planning with autonomous obstacle avoidance using reinforcement learning for six-axis arms is proposed. By adopting two assisted learning strategies, the learning process is effectively speeded up. In addition, using the finiteness of the inverse solution sets of the mechanical arm, the pose angle solution set can be calculated accurately. The experimental results of demonstrate the effectiveness and efficiency of the proposed approach. Our future work will focus on other reinforcement learning algorithms [16-18] for more complex tasks.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (No.71732003), in part by the National Key Research and Development Program of China under Grant 2016YFD0702100, and in part by the Fundamental Research Funds for the Central Universities under Grant 011814380035.

References

[1] J. Chen and K. Song, "Collision-Free Motion Planning for Human-Robot Collaborative Safety Under Cartesian Constraint," *2018 IEEE International Conference on Robotics and Automation*, Brisbane, QLD, 2018, pp. 4348-4354.

[2] X. Zhao, Z. Cao, Q. Jia, L. Pang, Y. Yu and M. Tan, "A Vision-Based Robotic Grasping Approach under the Disturbance of Obstacles," *2018 IEEE International Conference on Mechatronics and Automation*, Changchun, 2018, pp. 2175-2179.

[3] G. Nirmala, S. Geetha and S. Selvakumar, "Intellectual method of guiding mobile robot navigation using reinforcement learning algorithm," *2015 IEEE International Conference on Engineering and Technology*, Coimbatore, 2015, pp. 1-4.

[4] Z. Wang, C. Chen, H. H. Li, D. Dong and T. J. Tarn, "Incremental reinforcement learning with prioritized sweeping for dynamic environments," *IEEE/ASME Transactions on Mechatronics*, vol. 24, pp. 621-632, 2019.

[5] S. N. Gai, R. Sun, S. J. Chen and S. Ji, "6-DOF Robotic Obstacle Avoidance Path Planning Based on Artificial Potential Field Method," *2019 16th International Conference on Ubiquitous Robots*, Jeju, Korea (South), 2019, pp. 165-168.

[6] M. Mahdavian, M. Shariat-Panahi, A. Yousefi-Koma and A. Ghasemi-Toudeshki, "Optimal trajectory generation for energy consumption minimization and moving obstacle avoidance of a 4DOF robot arm," *3rd RSI International Conference on Robotics and Mechatronics*, Tehran, 2015, pp. 353-358.

[7] Yun Chao, Liu Gang, Wang Gang et al., "Obstacle avoidance for redundant manipulators using RBF neural networks and quadratic programming", *Journal of Mechanical & Electrical Engineering*, vol. 33, no. 1, pp. 1-7, 2016.

[8] X. Xue, Z. Li, D. Zhang and Y. Yan, "A Deep Reinforcement Learning Method for Mobile Robot Collision Avoidance based on Double DQN," *IEEE 28th International Symposium on Industrial Electronics*, Vancouver, BC, Canada, 2019, pp. 2131-2136.

[9] Q. Wang and C. Phillips, "Cooperative collision avoidance for multi-vehicle systems using reinforcement learning," *18th International Conference on Methods & Models in Automation & Robotics*, Miedzyzdroje, 2013, pp. 98-102.

[10] L. E. Kavraki, P. Svestka, J. -. Latombe and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566-580, Aug. 1996.

[11] C. J. Watkins, P. Dayan, "Q-learning", *Machine learning*, vol. 8, no. 3-4, pp. 279-292, 1992.

[12] S. Wei, Y. Wang, Y. Yang, F. Yin, W. Cao and Y. Tang, "Applying Q-Learning Algorithm to Study Line-Grasping Control Policy for Transmission Line Deicing Robot," *International Conference on Intelligent System Design and Engineering Application*, Changsha, 2010, pp. 382-387.

[13] S. Wang, Z. Song, H. Ding and H. Shi, "An Improved Reinforcement Q-Learning Method with BP Neural Networks in Robot Soccer," *2011 Fourth International Symposium on Computational Intelligence and Design*, Hangzhou, 2011, pp. 177-180.

[14] M. Hamaya, T. Matsubara, T. Noda, T. Teramae and J. Morimoto, "Learning task-parametrized assistive strategies for exoskeleton robots by multi-task reinforcement learning," *2017 IEEE International Conference on Robotics and Automation*, Singapore, 2017, pp. 5907-5912.

[15] T. D. Le, A. T. Le and D. T. Nguyen, "Model-based Q-learning for humanoid robots," *2017 18th International Conference on Advanced Robotics*, Hong Kong, 2017, pp. 608-613.

[16] C. Chen, D. Dong, H. X. Li, J. Chu and T. J. Tarn, "Fidelity-based probabilistic Q-learning for control of quantum systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, pp. 920-933, 2014.

[17] J. Li, D. Dong, Z. Wei, Y. Liu, Y. Pan, F. Nori and X. Zhang, "Quantum reinforcement learning during human decision-making," *Nature Human Behaviour*, vol.4, pp. 294-307, 2020.

[18] Z. Ren, D. Dong, C. Chen and H. Li, "Self-paced prioritized curriculum learning with coverage penalty in deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, pp. 2216-2226, 2018.