2021 IEEE 10th Data Driven Control and Learning Systems Conference
May 14-16, 2021, Suzhou, China

Reinforcement Learning Tracking Control for Unknown Continuous Dynamic Systems

Linqi Ye [1], Jiayi Li [2], Changliang Wang[3], Houde Liu[4], Bin Liang[5]

1. The Center of Intelligent Control and Telescience, Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China
E-mail: ye.linqi@sz.tsinghua.edu.cn

2.School of Mechanical Engineering and Automation, Harbin Institute of Technology, Shenzhen, Shenzhen 518055, China
E-mail: lijiayi@stu.hit.edu.cn

3.Shanghai Academy of Spaceflight Technology, Shanghai 201109, China
E-mail: sastty@163.com

4. The Center of Intelligent Control and Telescience, Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China
E-mail:liu.hd@sz.tsinghua.edu.cn

5. Navigation and Control Research Center, Department of Automation, Tsinghua University, Beijing 100084, China
E-mail:bliang@tsinghua.edu.cn

Abstract: Reinforcement learning tracking control (RLTC) is proposed to solve the traditional iterative learning control (ILC) problem. For unknown continuous dynamic systems, precision output tracking is achieved for a given trajectory after several times of trials. The controller is composed of a feedback part and a feedforward part, where the feedback part uses linear states feedback to stabilize the system and the feedforward part is a time-dependent signal to ensure precision tracking. The desired trajectory is defined in a finite time interval and the controller is applied repeatedly to the system with the feedforward signal updated by reinforcement learning in each trial. The tracking problem is treated as a black-box optimization problem where the integral quadratic tracking error is a cost to be minimized under input constraints. In order to apply reinforcement learning, the feedforward signal is approximated by spline interpolation of a few representative points distributed uniformly along the tracking time interval. A search strategy based on squeeze theorem is adopted to update the value of the representative points to achieve minimum tracking error cost. The cart pole example illustrates the effectiveness of the proposed method.

Key Words: Tracking control, iterative learning control, reinforcement learning, black-box optimization.

1 Introduction

Reinforcement learning, a promising method to achieve human-level or superman control, is getting more and more attention with the recent development in artificial intelligence. Inspired by learning behaviors of organism, reinforcement learning usually refers to an agent that interacts with the environment and modifies its actions based on rewards received in response to its actions. This agent has proven to be effective in handling complicated tasks in various fields.

Playing board game is one of the most successful applications of reinforcement learning. In 2015, Google's DeepMind created AlphaGo, a program that beat human Go Champion Fan Hui, Lee Sedol, and the highest-ranking Go player, Ke Jie from China [1]. AlphaGo is trained by supervised learning from human expert moves, and reinforcement learning from self-play. At the time of this paper, the program has presented a more powerful algorithm named MuZero, which achieves superhuman performance without any knowledge of the underlying dynamics aiming to solve real-world problems with complex and unknown dynamics [2]. Reinforcement learning also inspires some new methods applied in continuous action space. A typical example is deep Q-network (DQN), which was used to create a human-level agent that is virtually unbeatable across a set of 49 Atari video games [3]. The derived method, deep Q-Learning [4], is a model-free algorithm that can operate over continuous action spaces, solving more than 20 simulated physics tasks, including cartpole swing-up, dexterous manipulation, legged locomotion and car driving. Another technique, known as adaptive dynamic programming (ADP) [5-6], is developed to solve optimal control problems. Using actor and critic neural networks, ADP is able to solve the intractable Bellman equation to obtain the optimal control law.

However, it is noted that most of these works focus on setpoint control, i.e., keeping the control objective at a fixed target position. To the authors' knowledge, only a few papers talk about tracking control problem [7-10] and the results are still far from perfect. Therefore, the extension of reinforcement learning to tracking control for continuous nonlinear systems has remained a non-trivial open problem. In this paper, we intend to apply reinforcement learning to the tracking control problem for unknown continuous dynamic systems. Since it is based on reinforcement learning, we will call it reinforcement learning tracking control (RLTC). We imagine that RLTC may be divided into two levels:

Primary RLTC: This is the short-term goal. The tracking task is fixed. The system is allowed to repeat the same tracking task for several times, where the performance is improved gradually towards the optimal. For this level of

*This work is supported by National Natural Science Foundation (NNSF) of China under Grant 62003188, 61803221, 11702176 and U1813216.

2021 IEEE 10th Data Driven Control and Learning Systems Conference (DDCLS) | 978-1-6654-2423-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/DDCLS52934.2021.9455473

control, the learning period is short, and the required information is relatively less. In fact, the task is the same as that in iterative learning control (ILC) [11-12]. However, ILC usually requires zero initial state error and has some limitations in dealing with systems with relative degree bigger than one and nonminimum phase systems.

Advanced RLTC: This is the long-term goal. The system can generalize its experience to new situations so that it can accomplish a new task without several times of trial. For this level of control, the learning period may be long so that adequate information about the system can be acquired, which may be represented by some nonlinear function approximator such as deep neural network.

In this paper, we focus on primary RLTC. Our goal is to achieve tracking for a given finite-time output trajectory by learning from previous trials. Unlike DQN or ADP, which focus on approximating the optimal feedback control law with neural networks, we simply use a linear feedback control law along with a feedforward signal, aiming to find an optimal feedforward signal that minimizes the integral quadratic tracking error. Then the tracking control problem is treated as an optimization problem under input constraints. When the system dynamics is exactly known, several trajectory optimization methods are available to solve this problem, such as pseudospectral methods [13-14], which use some specific polynomials to approximate the state and control input, and performs orthogonal collocation at a set of carefully selected quadrature points. However, these methods require exact knowledge of the system model, and the calculated input cannot be directly applied to real control.

Motivated by pseudospectral methods, we approximate the feedforward signal by a spline interpolation of some representative points so that the original problem is reduced to a finite-dimensional black-box optimization problem. We will solve this problem by using the idea of reinforcement learning, which can be summarized as a try-evaluate-improve process, i.e., we try a new policy, evaluate its performance, and adopt it if it is better than previous. A search strategy based on squeeze theorem is adopted to explore the possible locations of the representative points efficiently. Then the feedforward signal is going towards the optimal solution by continually adopting the better policy.

The main contribution of this paper is the development of RLTC method, which solves the traditional ILC problem from the perspective of trajectory optimization, releasing some inherit limitations of ILC. It may also be helpful for other black-box optimization problem. Compared to DQN and ADP, we focus on shaping the feedforward signal rather than the feedback signal. Moreover, RLTC method is generally direct, simple and powerful without using complicated neural networks, which does not require to know the system dynamics.

2 Main Results of RLTC

The control task of RLTC is the same as ILC, i.e., to track a given output trajectory which is defined in a finite-time interval by learning from previous trials. For RLTC, this problem is treated as an optimization problem to minimize the tracking error under input constraints. Like ILC, we assume that the initial conditions are set the same for each repetition and the system dynamics is invariant throughout the repetition. The control architecture of RLTC is shown in Fig. 1.

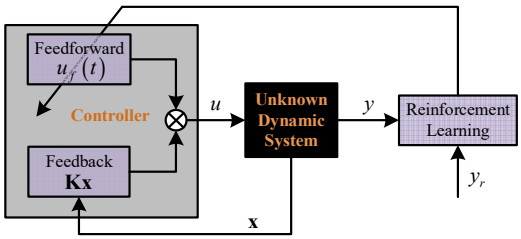

Fig. 1: RLTC architecture

The model of the dynamic system to be controlled is unknown. Only the states and the output are available for the controller. The output reference trajectory y_r is defined in a finite time interval $[0,T]$. The controller is composed of a feedback part and a feedforward part, where the feedback part uses linear states feedback to stabilize the system and the feedforward part is a time-dependent signal to ensure precision tracking. The controller is applied repeatedly to the system with the feedforward signal updated by reinforcement learning in each trial.

To begin with, it is assumed that there exists a linear control law $u = \mathbf{Kx}$ that can stabilize the system locally ($\mathbf{K}$ can be found by trial for simulated system). Then the controller is designed as follows

$$u = u_f(t) + \mathbf{Kx} \tag{1}$$

where $\mathbf{Kx}$ is the feedback part with $\mathbf{K}$ being the feedback gain matrix, and $u_f(t)$ is the feedforward signal. In this controller, the feedback part is fixed while the feedforward signal is modified in each trial. To evaluate the tracking performance of each trial, we define the integral quadratic tracking error as a cost, i.e., $Q = \int_0^T e^2(\tau)d\tau$. Obviously, the cost Q depends on the feedforward signal $u_f(t)$. The goal of this paper is to find an optimal feedforward signal such that the tracking error cost is minimized. Since $u_f(t)$ is a continuous signal depends on time t, this is an infinite-dimensional optimization problem. To make it easier to handle, the feedforward signal can be approximated by a spline interpolation of a few representative points. For simplicity, the representative points are selected as n points distributed uniformly along the tracking time interval. Let $\mathbf{u}_\mathbf{p}$ be the value vector of the representative points, and $\mathbf{t}_\mathbf{p} = [t_0, t_1, t_2, ..., t_{n-1}]$ be the corresponding time vector where $t_0 = 0, t_1 = T/(n-1), ...,$ $t_{n-1} = T$. Then the feedforward signal $u_f(t)$ is expressed as follows

$$u_f(t) = \text{spline}(\mathbf{t}_\mathbf{p}, \mathbf{u}_\mathbf{p}, t) \tag{2}$$

Thus the original problem turns into a finite-dimensional optimization problem, i.e., finding the most proper value of $\mathbf{u}_\mathbf{p}$ to minimize the tracking error cost Q. The optimal feedforward value vector $\mathbf{u}_\mathbf{p}^*$ is formally defined as follows

$$\mathbf{u}_\mathrm{p}^* = \arg\min_{\mathbf{u}_\mathrm{p}} \int_0^T e^2(\tau)\, d\tau \tag{3}$$

Since the system dynamics is unknown, this is a black-box optimization problem. We solve this problem by using the idea of reinforcement learning, which can be summarized as a try-evaluate-improve process. This is the core of RLTC, forming the policy updating loop as shown in Fig. 2. In this loop, we try a new feedforward control signal each time as policy exploration, evaluate its performance, and adopt the new policy if it is better than previous, during which the policy is gradually improved.

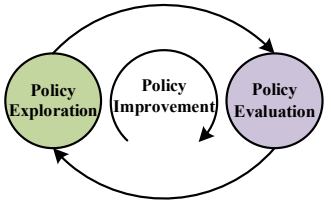

Fig. 2: Policy updating loop of RLTC

In the policy exploration, an efficient search strategy based on squeeze theorem is adopted. The search strategy has three main parts: up search, down search, and step adjust. Since the feedforward signal is represented by several points, we follow the coordinate descent idea, i.e., change the value of one point during each search. We do policy evaluation for each change to see if we get a smaller tracking error cost. It means to apply the controller (1) to the system with current feedforward signal. Then the overall flow chart of RLTC algorithm is shown in Fig. 3.

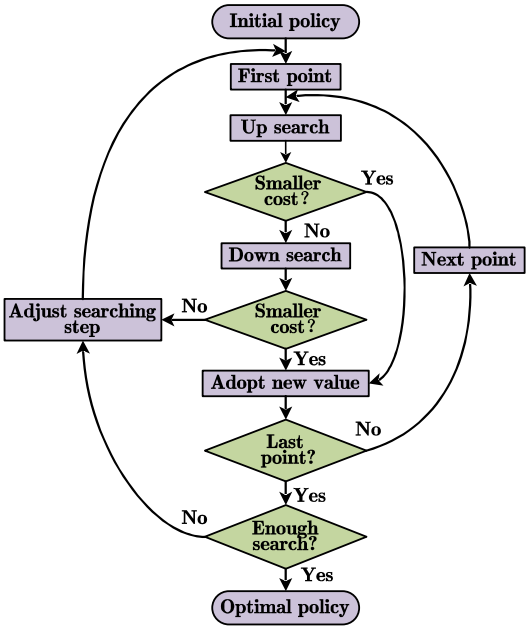

Fig. 3: Flow chart of RLTC algorithm

Initialization. An initial feedforward signal is set (e.g., select $\mathbf{u}_\mathrm{p}$ as a zero vector), and the cost is obtained in policy evaluation.

Policy Updating. The policy updating follows the coordinate descent idea. Each searching loop will start from the first point until the last point. First, do up search to replace its value by $\mathbf{u}_\mathrm{p}(i) = \mathbf{u}_\mathrm{p}(i) + \delta$, where δ is the searching step. Obtain the new cost in policy evaluation. If the new cost is smaller and the resulted input maintains in the constraints, then adopt the new value and go to next point, else recover the old value and do down search to replace its value by $\mathbf{u}_\mathrm{p}(i) = \mathbf{u}_\mathrm{p}(i) - \delta$. Obtain the new cost in policy evaluation. If the new cost is smaller and the resulted input maintains in the constraints, then adopt the new value and go to next point, else recover the old value and go to next point. If no improvement is made during the loop, then adjust the step by $\delta = \delta / 2$ and go to up search from the first point.

End. If enough search loop is made, output the optimal policy.

Fig. 4 shows an example of the feedforward signal updating process, where u_f is the final optimal feedforward signal, u_f^0, u_f^1, u_f^2 represent the initial feedforward signal, the feedforward signal after one searching loop, and the feedforward signal after two searching loops, respectively. In this example, the task duration is 10 seconds and 11 representative points are selected, which locate at the time instants 0, 1, 2, …, 10, respectively. It can be seen that starting from zero, the feedforward signal gradually goes towards the optimal solution.

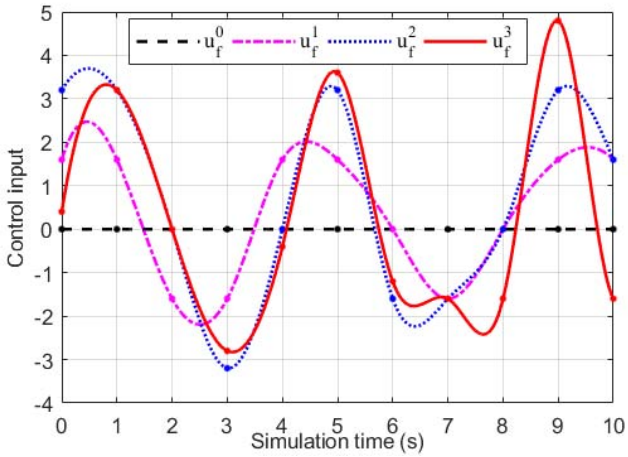

Fig. 4: Feedforward signal updating process

For the representative points, some go directly to the optimal location, such as the one at t = 5s, while others may first go in an opposite direction of the optimal location and then turn back, such as the one at t = 2s. This is because the RLTC algorithm looks for a better global rather than local solution when updating each representative point. In addition, a relatively big step in the beginning of the updating process may cause an excessive adjustment, which will be corrected in later iterations of the policy updating loop.

3 Simulation and Discussions

In this section, the RLTC method is applied to a simulated physical system, the cart-pole system, which is a classical nonlinear, unstable, nonminimum phase system that has been widely used in control education and research to demonstrate the effectiveness of various control methods. The cart-pole system is shown in Fig. 5, which consists of a cart and a planar inverted pendulum on it.

In this system, u is the external force that moves the cart in the horizontal plane, x is the cart position, θ is the pole angle, M is the mass of the cart, m is the mass of the pole, l is the half length of the pole. The equations of motion are

$$(M+m)\ddot{x} - ml\cos\theta\ddot{\theta} + ml\sin\theta\dot{\theta}^2 = u$$
$$I\ddot{\theta} - ml\cos\theta\ddot{x} - mgl\sin\theta = 0 \quad (4)$$

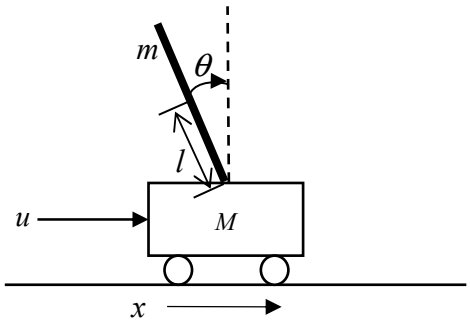

Fig. 5: Cart-pole system

where $I = 4ml^2/3$ is the moment of inertia of the pendulum with respect to the pivot, $g = 9.8m/s^2$ is the acceleration due to gravity. Denote $v = \dot{x}, \omega = \dot{\theta}$ as the cart velocity and the pole angular velocity, and denote $\mathbf{x} = [x, v, \theta, \omega]^T$ as the system states. Take the external force u as the control input and the cart position x as the output. The objective is to let the cart position track a given trajectory and keep the pole from falling down. In the simulation, the system parameters are selected as $m = 1kg, M = 10kg, l = 1m$, and the feedback control gain is designed as $\mathbf{K} = [5, 15, -200, -80]^T$. The input constraint is set as $-30 \le u \le 30$. The initial condition is set as $\mathbf{x}(0) = [0, 0, 0, 0]^T$. The simulation time is set as 10 seconds. The representative points number is set as 11. The searching loop number is set as 40. Two cases are considered with a step trajectory and a sinusoidal trajectory, respectively.

In the first case, a step trajectory $x_r = 1$ is considered. The simulation results are shown in Figs. 6 and Fig. 7.

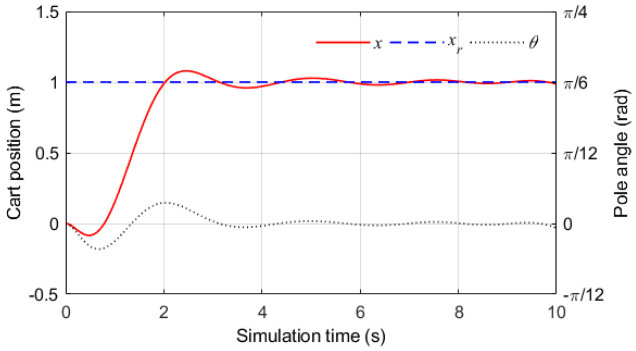

Fig. 6: Cart position and pole angle for step trajectory

As shown in Fig. 6, the cart position reaches the setpoint in about 2 seconds, then overshoot slightly, and finally achieves nearly steady-state. The final cost is 1.274. Meanwhile, the pole angle keeps in a small neighborhood around zero. From Fig. 7, it can be seen that the control input maintains in the input constraint range, which starts from the maximum negative value to eliminate the initial error and then converges to steady-state.

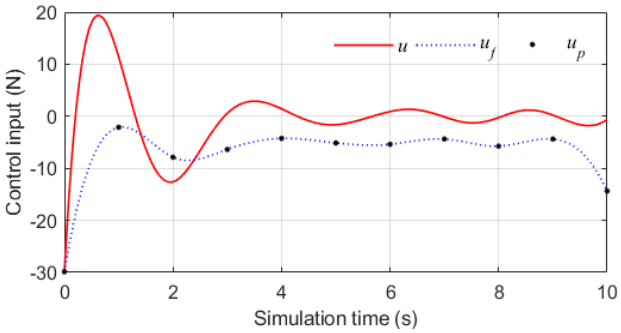

Fig. 7: Control input for step trajectory

As a comparison, we assume the system dynamics is known and solve this optimization problem by GPOPS, a program based on hp-adaptive pseudospectral methods [15]. The result is shown in Fig. 8. It can be seen that the optimal control input exhibits a "bang-bang" feature. The cost is 1.127. RLTC cannot lead to such result since the spline interpolation smooths the input signal. Nevertheless, comparing the two costs, RLTC still yields a nice result without knowing the system dynamics.

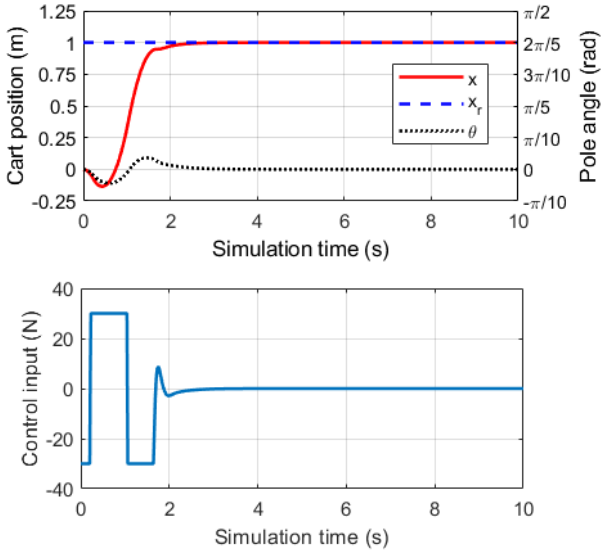

Fig. 8: Optimization result for step trajectory by GPOPS

In the second case, a sinusoidal trajectory $x_r = \sin(0.5\pi t)$ is considered. The simulation results are shown in Fig. 9 and Fig. 10.

As shown in Fig. 9, the cart position reaches the desired trajectory in about 2 seconds and then nearly coincides with the desired trajectory, meanwhile the pole keeps upright. The final cost is 0.6795. The control input maintains in the constraint range as shown in Fig. 10.

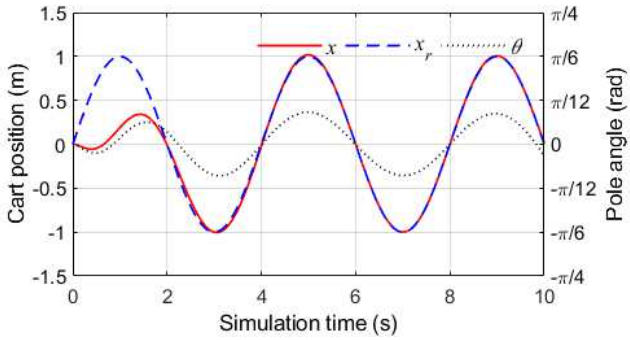

Fig. 9: Cart position and pole angle for sinusoidal trajectory

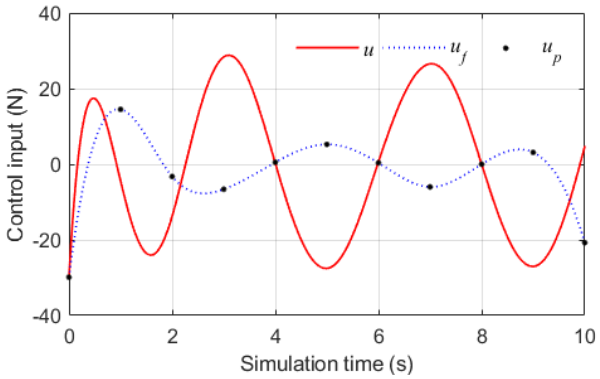

Fig. 10: Control input for sinusoidal trajectory

The optimization result by GPOPS is shown in Fig. 11, with a cost of 0.5267. Still a "bang-bang" feature is observed in the beginning. Then after t = 3s the optimal control input is nearly the same as the input obtained by RLTC.

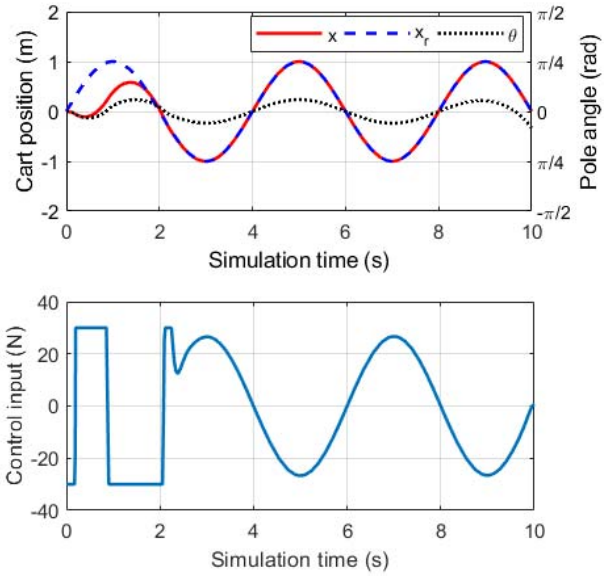

Fig. 11: Optimization result for sinusoidal trajectory by GPOPS

In this case, it is noted that there is an initial tracking error though the cart position is in the right place at the beginning. This is due to the initial condition mismatch. Since the desired cart position is $x_r = \sin(0.5\pi t)$, the desired cart velocity should be $v_r = 0.5\pi\cos(0.5\pi t)$. Therefore, the initial values for the cart position and velocity should be $x_r(0) = 0$ and $v_r(0) = 0.5\pi$. As for the pole angle and the angle velocity, their desired trajectories and initial values are determined by ideal internal dynamics [16]. Based on the previous simulation, we can find out the values for the pole angle and angular velocity at time t = 8s, which are about $\theta = 0$ and $\omega = 0.3$. Therefore, a matched initial condition should be $\mathbf{x}(0) = [0, 1.57, 0, 0.3]^T$. The simulation result of RLTC is shown in Fig. 12 with the matched initial condition. It can be observed that the initial tracking error is eliminated and the proposed RLTC method achieves nearly perfect tracking.

In our algorithm, the searching loop number and the representative points number play important roles in the final tracking performance. To evaluate their impacts, more simulations will be taken in the next.

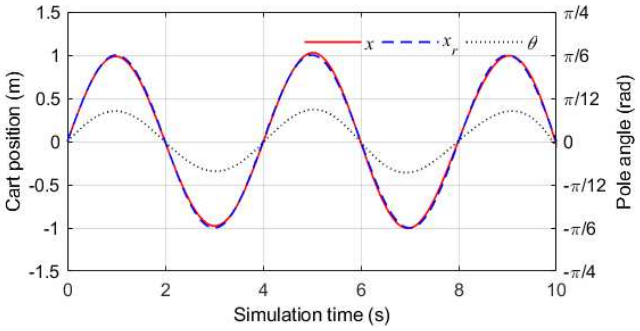

Fig. 12: Cart position for sinusoidal trajectory with matched initial condition

- Impact of searching loop number

To explore the impact of searching loop number, a comparison is made with different searching loop number (N=2, N=5, and N=10) for the sinusoidal reference as shown in Fig. 13.

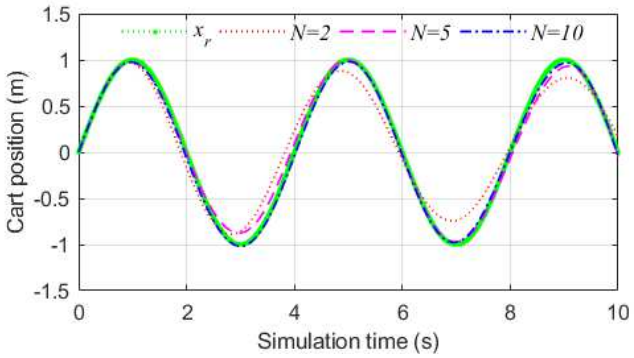

Fig. 13: Cart position for sinusoidal trajectory with different searching loop number

From Fig. 13, it can be seen that the cart position approaches the desired trajectory as the searching loop number increases and nearly coincides with the desired trajectory after ten searching loops of trial and improvement. To show this impact more clearly, the tracking error cost-searching loop number curve is shown in Fig. 14, from which it can be seen that the tracking error cost converges to zero quickly with the increase of the searching loop number and approaches nearly zero when N=10.

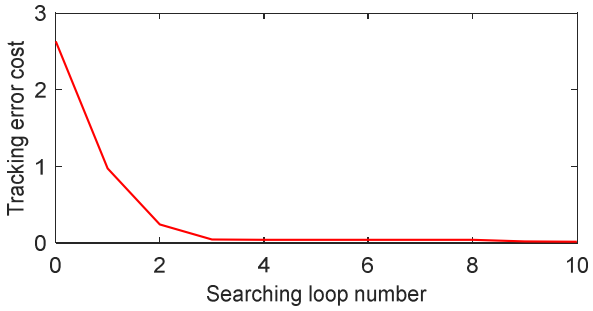

Fig. 14: Tracking error cost-searching loop number curve

- Impact of representative points number

Another factor, the representative points number, will also affect the tracking performance as it determines the maximum approximation precision for the optimal feedforward input that can be achieved. To evaluate this impact, a comparison is made with different representative points number (n=6, n=11, and n=21) for the sinusoidal

reference. The tracking error cost-iteration number curve is shown in Fig. 15.

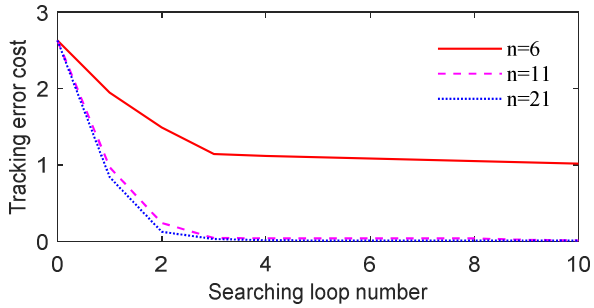

Fig. 15: Tracking error cost-searching loop number curve with different number of representative points

As can be seen from Fig. 15, when the representative points are too sparse, say n = 6, the tracking error will remain large even with enough searching loops. That's because the sampling points are too less so that the spline interpolation cannot approximate the optimal feedforward signal properly. Comparing the results for n = 11 and n = 21, it can be seen that the tracking error converges a little faster when n = 21 but it is also more time-consuming since it has more points to adjust. As a trade-off, a proper number of representative points should be chosen to ensure a good tracking (enough points should be chosen) as well as efficiency (points not too much).

4 Conclusions

In this paper, reinforcement learning tracking control (RLTC) is proposed to accomplish a repeated tracking control task for unknown continuous dynamic systems. RLTC solves the traditional ILC problem from a new way, releasing some inherit limitations of ILC, it can therefore deal with nonzero initial state errors, systems with relative degree bigger than one and nonminimum phase systems. In the RLTC framework, a feedback control is designed to stabilize the system and a feedforward signal is updated by reinforcement learning to minimize the tracking error cost. The feedforward signal is approximated by spline interpolation of a few representative points to convert the problem to a finite-dimensional black-box optimization problem. The core of RLTC is the policy updating loop, where the policy exploration based on squeeze theorem can explore the possible locations of the representative points efficiently and the policy evaluation can tell whether the tracking performance is improved. Then the feedforward signal is going towards the optimal by continually adopting the better policy. RLTC makes a successful attempt to apply reinforcement learning to solve the tracking control problem. The method is generally direct and simple, without using complicated neural networks, as well as very powerful which does not require to know the system dynamics. However, this paper only deals with the primary RLTC problem with repeated tracking task. In the future, advanced RLTC will be investigated to possibly accomplish new tracking task by learning from the experience.

References

[1] D. Silver, et al., Mastering the game of Go with deep neural networks and tree search, *Nature* 529(7587): 484-489, 2016.

[2] D. Silver, et al., Mastering Atari, Go, chess and shogi by planning with a learned model, *Nature* 588(7839): 604-609, 2020.

[3] V. Mnih, et al., Human-level control through deep reinforcement learning, *Nature* 518(7540): 529-533, 2015.

[4] T. P. Lillicrap, et al., Continuous control with deep reinforcement learning, *arXiv preprint* arXiv: 1509.02971, 2015.

[5] F. L. Lewis and D. Vrabie, Reinforcement learning and adaptive dynamic programming for feedback control, *IEEE circuits and systems magazine* 9(3):32-50, 2009.

[6] F. Wang, H. Zhang, and D. Liu, Adaptive dynamic programming: An introduction, *IEEE computational intelligence magazine* 4(2): 39-47 ,2009.

[7] H. Zhang, et al., Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method, *IEEE Transactions on Neural Networks* 22 (12): 2226-2236, 2011.

[8] R. Kamalapurkar, et al., Approximate optimal trajectory tracking for continuous-time nonlinear systems, *Automatica* 51: 40-48, 2015.

[9] C. Mu, et al., Data-driven tracking control with adaptive dynamic programming for a class of continuous-time nonlinear systems, *IEEE transactions on cybernetics* 47(6): 1460-1470, 2017.

[10] B. Luo, D. Liu, T. Huang, and J. Liu., Output Tracking Control Based on Adaptive Dynamic Programming with Multistep Policy Evaluation, *IEEE Transactions on Systems, Man, and Cybernetics* 49(10):2155-2165, 2019.

[11] D. A. Bristow, M. Tharayil, and A. G. Alleyne, A survey of iterative learning control, *IEEE Control Systems* 26(3): 96-114, 2006.

[12] H. Ahn, Y. Chen, and K. L. Moore, Iterative learning control: Brief survey and categorization, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37(6): 1099-1121, 2007.

[13] F. Fahroo, and I. M. Ross, Direct trajectory optimization by a Chebyshev pseudospectral method, *Journal of Guidance, Control, and Dynamics* 25(1): 160-166, 2002.

[14] D. Benson, *A Gauss Pseudospectral Transcription for Optimal Control*, Ph.D. Thesis, Dept. of Aeronautics and Astronautics, MIT, November 2004

[15] M. A. Patterson, and A. V. Rao, GPOPS-II: A MATLAB software for solving multiple-phase optimal control problems using hp-adaptive Gaussian quadrature collocation methods and sparse nonlinear programming, *ACM Transactions on Mathematical Software (TOMS)* 41(1), 2014.

[16] S. Gopalswamy and J. K. Hedrick, Tracking nonlinear non-minimum phase systems using sliding control, *International Journal of Control*, 57(5):1141-1158, 1993.