


Article

# Reinforcement Learning with Side Information for the Uncertainties

Janghoon Yang 

Department of A.I. Software Engineering, Seoul Media Institute of Technology, Seoul 07590, Republic of Korea; jhyang@smit.ac.kr; Tel.: +82-2-6393-3237

**Abstract:** Recently, there has been a growing interest in the consensus of a multi-agent system (MAS) with advances in artificial intelligence and distributed computing. Sliding mode control (SMC) is a well-known method that provides robust control in the presence of uncertainties. While our previous study introduced SMC to the reinforcement learning (RL) based on approximate dynamic programming in the context of optimal control, SMC is introduced to a conventional RL framework in this work. As a specific realization, the modified twin delayed deep deterministic policy gradient (DDPG) for consensus was exploited to develop sliding mode RL. Numerical experiments show that the sliding mode RL outperforms existing state-of-the-art RL methods and model-based methods in terms of the mean square error (MSE) performance.

**Keywords:** reinforcement learning; consensus; multi-agent system; sliding mode control



**Citation:** Yang, J. Reinforcement Learning with Side Information for the Uncertainties. *Sensors* **2022**, *22*, 9811. <https://doi.org/10.3390/s22249811>

Academic Editors: Gianmarco Romano, Giovanni Di Gennaro and Amedeo Buonanno

Received: 18 November 2022

Accepted: 13 December 2022

Published: 14 December 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

With the evolution of network technologies, previously standalone devices were getting connected to the network, thereby achieving the vision of the “internet of everything”. Accordingly, the network plays a critical role in a control system, often called a networked controlled system. Many networked controlled systems comprise multiple subsystems which can be modules for specific functions or agents to operate independently. Although decentralized control has been developed as an alternative to a centralized control to overcome the complexity issue, its performance can be degraded owing to limited information. Distributed control is known to provide a tradeoff between complexity and performance by utilizing information from a subset of agents in the system.

The objective of the distributed control usually depends on the goal of the system. A computational model for the distributed control of a system with sensing and actuation operating over wireless sensor networks was proposed to address issues intrinsic to the sensor network, such as communication jitter [1]. A distributed proportional-integral-derivative (PID) controller was derived by formulating primal-dual dynamics to maximize the sum network utility maximization with a link capacity constraint [2]. A model predictive distributed control for a system consisting of interacting subsystems was developed in the context of min-max optimization to derive robust distributed control [3]. Its significance has been shown to increase in association with cyber-physical systems (CPS) such as the cooperative control of manipulators and multi-agent system (MAS) such as microgrids [4]. The MAS is usually defined as a system comprising a set of agents for performing a task [5]. One of the most popular problems associated with it is the consensus problem in which each agent executes an action to achieve a common goal. Consensus control can be considered as a type of distributed control in which each agent shares its state information with neighbor agents, from which a consensus control is determined. Consensus control for the MAS has been studied in many practical problems such as a swarm of unmanned aerial vehicles (UAVs) [6], autonomous vehicle platoon [7], reactive power control in microgrids [8], and teleoperation of cyber-physical systems [9].

For simplicity of analysis, some studies assume that there is no delay associated with exchanging information. However, communication delays are unavoidable in practical systems, and various approaches have been proposed to address them. When the delays are the same for all communication links, the conventional consensus protocol achieves consensus as long as the delay is less than the threshold determined by the algebraic connectivity of the communication graph [10]. Similarly, an asynchronous event-triggered consensus protocol for MAS with second-order dynamics and the same delay was shown to achieve an average consensus when the delay is less than the event detection period [11]. An observer was introduced to predict the state of the MAS with both delay and disturbance [12] and cooperative containment control with delay [13]. The consensus conditions have been given a linear matrix inequality (LMI) from the Lyapunov stability condition for a homogenous MAS with a Markov delay [14], heterogeneous MAS with random link failures [15], and MAS with higher-order dynamics and multiple time-varying delays [16]. A bounded delay condition for the consensus of a heterogeneous MAS has also been developed using a frequency domain method [17].

Because delayed information is different from information without delay, it can be considered uncertain. A popular method for developing robust control involves exploiting sliding mode control (SMC). It is a nonlinear control in which the switching operation induces the state of the system into the sliding surface which can be defined independently from the original system dynamics and uncertainties [18]. SMC has been introduced to address different types of uncertainties and system configurations associated with the consensus of MAS such as an affine nonlinear MAS with disturbances and system uncertainties [19], heterogeneous second-order MAS with uncertain parameters [20], nonlinear MAS with communication delay [21], second-order MAS with constant same input delay and disturbance [22], and second-order MAS with un-known time-varying delays and disturbances [23]. Alternatively, reinforcement learning (RL) can be introduced to develop robust consensus control. RL learns to generate an action to maximize the expected return. Most existing RL-based consensus controls were developed from adaptive dynamic programming by solving coupled Hamilton–Jacobian–Bellman (HJB) equations [24–26]. They focused on the development of a consensus algorithm without explicit model knowledge, rather than dealing with uncertainties. In addition, most existing RL algorithms are sensitive to parameterizations and convergence problems [24].

When multiple agents move together, their communication links become fragile. Thus, centralized control may not be feasible, necessitating distributed consensus control. In addition, there are uncertainties in the system model owing to uncertainties such as disturbances and delays. The recent success of RL in many control problems and the robust performance of SMC in the presence of uncertainties motivated this study. Many existing consensus control algorithms depend on model knowledge to derive a control signal from an assumed model. However, accurate system modeling is often limited, which results in degraded control performance. Alternatively, an RL approach that utilizes learning through experience without specific model knowledge may be adopted. However, most existing RL algorithms are sensitive to parameterizations and convergence problems [27].

Our previous work showed that, although the application of a twin-delayed deep deterministic policy gradient (DDPG) with articulated reward shaping provides a robust performance of the consensus, its performance is limited in comparison to that of the model-based algorithm [28]. Ref. [29] combined SMC with RL for consensus control in the presence of uncertainties, which was called the “slide RL”. The performance of the combined method is comparable to that of model-based control. However, slide RL works only when the parameters are initialized with values close to zero, and the update rate is very small. Thus, it is very sensitive to parameterizations. Because the slide RL is based on the RL developed from the coupled HJB equation, its development for various types of control problems necessitates mathematical development for approximate dynamic programming. This can be very difficult to derive for a system with particular dynamics. Thus, to develop a consensus algorithm whose performance is robust to uncertainties and parameterization,

and comparable to that of the model-based algorithm, we combine SMC with the state-of-the-art RL and twin delayed DDPG called TD3 [30]. This is called the “sliding mode RL” to distinguish it from the slide RL. The sliding mode RL is designed such that the switching control provides robustness to uncertainties, while the RL-based control improves the performance further by exploiting the SMC structure. The simulation result shows that the sliding mode RL achieved the best mean square error (MSE) performance in comparison to the model-based consensus algorithm and other RL-based algorithms. To further analyze the characteristics of the sliding mode RL, the performance of the switching control (SC), which is an SMC without a linear control part, was assessed. Interestingly, the performance of the SMC without linear control was comparable to that of other methods considered. This result suggests that sliding mode RL benefited considerably from nonlinear sliding control rather than RL. Transmission over wireless communication links incurs delays owing to distance, processing delays, and channel errors. In addition, disturbances, such as wind and rain, can occur in the moving path, and it is often difficult to obtain a precise mathematical model to derive a control signal. Thus, it is important to develop a robust control method without explicit knowledge of the system model. The remainder of this study is organized as follows: a system model for a multi-agent system and the objective of the consensus of MAS in mathematical form are presented in Section 2. The proposed sliding mode RL and its pseudo-code are described in Section 3. The numerical results which verify the performance of the sliding model RL are presented in Section 4. Section 5 contains concluding remarks and directions for future research.

## 2. System Model and Formulation of the Problem

In this study, we considered a second-order homogeneous MAS with a single leader and multiple followers in an environment with unknown time-varying delays and disturbances. Figure 1 depicts the four different realizations of the communication graphs, where node 0 is considered the leader node for all graphs. Each agent is assumed to transmit information to neighboring nodes that are connected through an edge in a communication graph. A leader–follower MAS aims to achieve a consensus on the position of the leader agent by sharing information with neighbors connected through communication links. The corresponding behavior of each agent can be expressed as

$$\ddot{x}_i(t) = u_i(t) + d_i(t), \quad (1)$$

where  $u_i(t)$ ,  $d_i(t)$ ,  $x_i(t)$ , and  $\ddot{x}_i(t)$  denote the control signal of agent  $i$ , an unknown bounded disturbance, the position of agent  $i$ , and second-order derivative of  $x_i(t)$ , respectively. Since  $u_i(t)$  is determined from the information available at agent  $i$ , it can be expressed as

$$u_i(t) = f(H(X_i(t), X_{i^-}(t))), \quad (2)$$

where  $H(\cdot)$  is the history of inputs from the initial time to time  $t$ ,  $X_i(t) = [x_i(t), \dot{x}_i(t)]$ ,  $X_{i^-}(t) = \cup_{j \in N_i} \{x_j(t - \tau_{i,j}(t)), \dot{x}_j(t - \tau_{i,j}(t))\}$ ,  $N_i$  is a set of neighbor agent indices of agent  $i$ ,  $\tau_{i,j}(t)$  is a communication delay at time  $t$  from agent  $j$  to agent  $i$ , and  $f(\cdot)$  is a mapping whose output is an action.

The objective of the control of a MAS considered in this research is to achieve positional consensus to the leader agent, which can be expressed as

$$\lim_{t \rightarrow \infty} |e_{d,i}(t)| = 0 \text{ for } \forall i, \quad (3)$$

where  $e_{d,i}(t) = x_i(t) - x_0(t)$ .  $e_{d,i}(t)$  is often referred to as a disagreement vector. To achieve (3), consensus control is performed independently for each agent using the available information. The convergence of the consensus of MAS with the knowledge of a perfect system model and an introductory explanation of the consensus are provided in [10]. However, the leader agent may send its state information to neighbor agents only. In addition, there may be a delay in the communication links. Thus, information on the

disagreement vector is often unavailable. Alternatively, the local position error  $e_{x,i}(t)$  and local velocity error  $e_{v,i}(t)$  are used to generate the control signal:

$$e_{x,i}(t) = \sum_{j \in N_i} x_i(t) - x_j(t - \tau_{i,j}(t)), \tag{4}$$

$$e_{v,i}(t) = \sum_{j \in N_i} \dot{x}_i(t) - \dot{x}_j(t - \tau_{i,j}(t)), \tag{5}$$

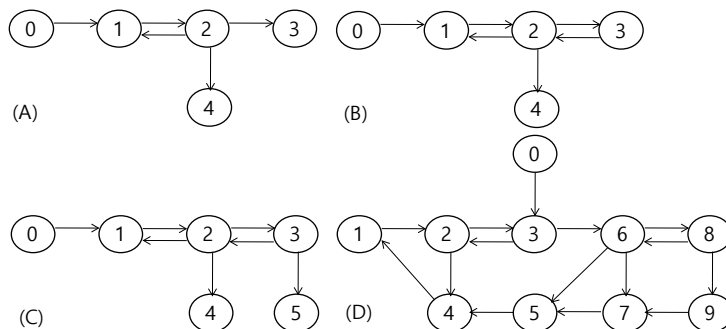


Figure 1. (A–D) Communication graphs for simulations.

### 3. A Sliding Mode RL for the Consensus of a MAS

In this section, a framework for developing a sliding mode RL is explained first so that it can be applied to most existing RLs independently from the specific algorithm structure in RL. Then, the sliding mode RL based on TD3 is presented with a pseudo-code as a specific realization. The time index is omitted for simplicity.

RL is a model-free algorithm for determining the policy of an agent through interactions with the environment which follows the Markov decision process (MDP). When the model information, which is transition probability, is given, it can be solved using the Bellman equation satisfied by the optimal policy  $\pi^*$  [30]:

$$Q^{\pi^*}(s, a) = r(s, a) + \gamma E_{s', a'} \{ Q^{\pi^*}(s', a') \}, \tag{6}$$

where a tuple  $(s, a, r, s', a')$  which consists of the current state  $s$ , current action  $a$ , reward  $r$ , next state  $s'$ , and next action  $a'$  defines an experience used to update the RL.  $Q^\pi(s, a)$  is called a state–action value function, which is determined by a policy  $\pi$ .

The control signal with SMC can be expressed as

$$u_{SMC} = f(x_0, \dots, x_K) - k_u \text{sign}(q), \tag{7}$$

where  $f(\cdot)$  is a function that is determined from the definition of the sliding variable and dynamicity of the agents,  $K$  is the number of follower agents,  $k_u$  is a nonlinear control gain that determines the degree of robustness to uncertainties, and  $q$  is the sliding variable. The proposed method combines RL and SMC such that  $f(\cdot)$  can be determined through RL. Nonlinear switching control in the second part of Equation (7) is kept the same as in SMC. The corresponding sliding mode RL is expressed as (8)

$$u_{sliding\ mode\ RL} = A_\phi(s) - k_u \text{sign}(q) + \varepsilon, \tag{8}$$

where  $s$  is the state information used for generating an action,  $A_\phi(\cdot)$  is the output of a neural network parameterized by  $\phi$ , and  $\varepsilon$  is the exploration noise of which distribution can be defined in an implementation-specific way.

TD3 [30] is considered as a specific method to be used for developing the proposed algorithm. The structure of the proposed sliding mode RL is shown in Figure 2. RL generates an action for refining the switching control, while the sliding variable is updated with the position and velocity errors at each time step. The control signal  $a$  is generated

by Equation (8) and applied to the agent. A resulting tuple of experience  $(s, a, r, s', q)$  is stored in the replay buffer from which the batch training data are sampled to train the neural networks for RL. TD3 was designed to reduce the overestimation bias by using a pair of independent critics, considering the target action–value function with clipped double Q learning and delaying policy updates. The pseudo-code for the sliding mode TD3 is presented in Figure 3 through modifying the pseudo-code of TD3 in [30]. It starts with initializing hyper-parameters and replaying buffer. At each time step, it generates an action which can be decomposed into SMC and exploration noise. The SMC is further decomposed into the refinement control, which is generated by the actor network, and the switching control, which is driven by local errors. Then, the tuple of the experience at the current time step is saved at the replay buffer. Now,  $N$  samples ( $N$  is the batch size) are sampled from the replay buffer to train the network for RL. After generating the action for the next state from the actor network and clipped exploration noise, the target Q value is calculated from the double Q values from target critic networks. The critic networks are then updated. If the current time step  $t$  is the integer multiple of the target update period, the actor networks are updated by the deterministic policy gradient. The target critic and actor networks are also updated with the parameters of the main critic and actor networks, respectively. The modifications from the original TD3 are  $a = A_\phi(s) - k_u \text{sign}(q) + \epsilon$  in the generation of the action and  $(s, a, r, s', q)$  in the sample buffer.  $q$  is included in the sample buffer since the actor network learns to generate the part of the control signal except for switching control.

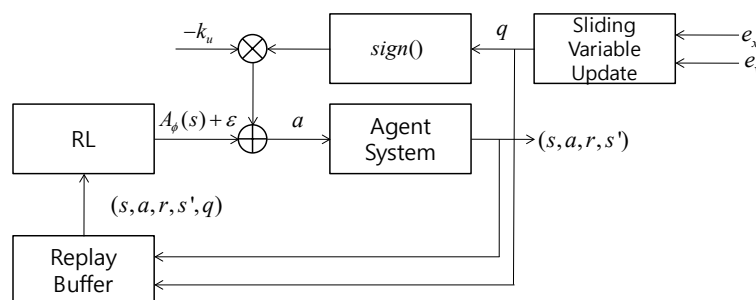


Figure 2. Structure of sliding mode RL.

```

Initialization :
Initialize hyper parameters :  $\theta_1, \theta_2, \phi, \theta'_1 = \theta_1, \theta'_2 = \theta_2, \phi' = \phi$ 
Initialize replay Buffer B
for  $t = 1 : T$ 
    generate exploration noise :  $\epsilon \sim N(0, \sigma)$ 
    generate action :  $a = A_\phi(s) - k_u \text{sign}(q) + \epsilon$ 
    calculate sliding variable:  $q = ce_x + e_v$ 
    store transition tuple  $(s, a, r, s', q)$  in B

    sample mini-batch of  $N$  transitions  $(s, a, r, s')$  from B
     $a' = A_{\phi'}(s') + \epsilon', -k_u \text{sign}(q), \epsilon' \sim \text{clip}(N(0, \sigma'), -c, c)$ 
     $y = r + \gamma \min(Q_{\theta'_1}(s', a'), Q_{\theta'_2}(s', a'))$ 
    update  $\theta_i$  with the gradient of  $N^{-1} \sum (y - Q_{\theta_i}(s, a))^2$  for  $i = 1, 2$ 
    if  $t \bmod d == 0$ 
         $\nabla_\phi J(\phi) = N^{-1} \sum \nabla_a Q_{\theta_i}(s, a)|_{a=A_\phi(s) - k_u \text{sign}(s, \text{slide})} \nabla_\phi A_\phi(s)$ 
        update  $\phi$  with  $\nabla_\phi J(\phi)$ 
    update target networks :
         $\theta'_i = \tau \theta_i + (1 - \tau) \theta'_i$ 
         $\phi' = \tau \phi + (1 - \tau) \phi'$ 
    
```

Figure 3. Pseudo-code for the sliding mode TD3.

#### 4. Numerical Simulations

In this section, the performance of sliding mode RL is assessed via numerical experiments. For better assessment of the characteristics of the sliding mode RL, we considered the several different system configurations [24] which are summarized in Table 1. Each system configuration is set with different accelerations of the leader agent, disturbances, communication graphs, and delays. The considered maximum delay was 0.5 while the delay differed over each communication link. The disturbances were different for each agent, whereas the envelope of the disturbance was the same with some delay. Figure 1 shows the four communication graphs used for the simulations. Node 0 in all graphs was set as the leader node. While the graphs in the Figure 1A,B possess the same number of nodes, the graph in Figure 1B has an additional edge from node 3 to node 2. The graph in Figure 1C was constructed by adding one more node to the graph in Figure 1B. Finally, the graph in Figure 1D was included to evaluate the performance of the proposed algorithm for a relatively large communication graph. Although these graphs were considered for the effect of the graph on the proposed algorithm, the simulation results in the subsequent section indicate that the graph itself did not significantly affect the performance. The MAS was simulated at a sampling rate of 0.01 s.

**Table 1.** Simulation system configurations with the acceleration of the leader agent, disturbance, communication graph, and delay.  $k$  and  $l$  are the indices of two connected nodes.

Case	Acceleration of Leader Agent	Disturbance	Graph	Delay
1	$\cos(7t) + \cos(3t)$	$\sin(11t) + \cos(13t)$	A	$0.25(1 + \cos(t + (k+l)\pi/7))$
2	$[\cos(7t) + \cos(3t)](2 - e^{-t})$	$\sin(11t)[3 - e^{-t}]$	A	$0.25(1 + \cos(111t + (k+l)\pi/7))$
3	$[\cos(7t) + \cos(3t)](2 - e^{-t})$	$\sin(11t)[3 - e^{-t}]$	B	$0.5(1 + e^{-0.1(k+l)t})^{-1}$
4	$[\cos(7t) + \cos(3t)](2 - e^{-t})$	$\sin(11t)[3 - e^{-t}]$	C	$0.5(1 - (1 + e^{-0.1(k+l)t})^{-1})$
5	$[\cos(7t) + \cos(3t)](2 - e^{-t})$	$\sin(11t)[3 - e^{-t}]$	D	$0.5(1 + \cos(t + (k+l)\pi/7))(2 + e^{-0.1(i+j)t})^{-1}$
6	$\cos(17t)(3 - e^{-t})(2 + \cos(13t))^{-1}$	$\cos(23t)(e^{-0.1} + 1) - e^{-t}$	D	$0.5(1 - 0.5(1 + \cos(t + (k+l)\pi/7))e^{-0.1(k+l)t})$

The actor network consisted of a linear hidden layer with 256 nodes and an output layer with a hyper-tangent activation function. The critic network first concatenated the sub-network for the state and one for the action where the first one comprised two hidden layers with 16 and 32 nodes, and ReLU activation, and the second one consisted of one hidden layer with 32 nodes and ReLU activation. The concatenated sub-network outputs were passed to two hidden layers with 256 nodes of which output was passed to a linear output layer. The Adam optimizer was exploited to update the weight while the learning rate was 0.01 for both networks, and the batch size was 1024. The output of the actor network was clipped between  $-10$  and  $10$ . The update rate of the target network was set to 0.0001. The values of  $\sigma$ ,  $\sigma'$ ,  $c$ ,  $d$ , and  $\gamma$  were set 2.0, 2.0, 3.0, 2, and 0.99, respectively. The state for RL was defined as  $e_{x,i}(t)$  and  $e_{v,i}(t)$  over the most recent five time steps, which resulted in a state dimension of 10. Defining the states in terms of the local error also has the advantage of enabling the use of the same network regardless of the number of neighbor agents.

The gain for switching control  $k_u$  is an important parameter for implementing the sliding mode RL. A simulation was performed to determine proper  $k_u$ . The position and velocity of each agent were initialized using a standard normal variable. The follower agent was set to drift with disturbance without applying consensus control for a second, while the leader agent was set to move with the defined dynamics in Table 1 to realize the delay over communication links. Consensus control was then applied for 100 s, which was observed to be sufficient time for the convergence of a stable consensus algorithm. For the last 50 s, the mean squared local error (MSE) and mean squared disagreement error (MSD) were measured with 20 different initializations for each case. Table 2 shows the characteristic of the sliding mode RL with the  $k_u$  values for cases 5 and 6. Although the performance of



the sliding mode RL was assessed with  $k_u$  values of 20 and 30, their performances were not included, since several divergences occurred for cases 5 and 6 while no divergence was observed for other cases. A larger  $k_u$  in SMC typically increases the robustness of the control to uncertainties. However, large values of  $k_u$  often incur excessive control at each time instance, which increases the MSE and MSD, although it is not significant. In other words,  $k_u$  incurs a tradeoff between stability and performance at convergence. Considering that a  $k_u$  of 40 did not incur divergence and it resulted in the smallest MSE and MSD,  $k_u$  was set to 40.

**Table 2.** MSE and MSD with different  $k_u$  s for cases 5 and 6.

Metrics	Case\ $k_u$	40	50	100
MSE	5	0.00011	0.00016	0.00073
	6	0.00013	0.00017	0.00088
MSD	5	0.01475	0.01521	0.01971
	6	0.00965	0.00994	0.01616

The proposed sliding mode RL was evaluated for six cases listed in Table 1 and compared with the SMC [23], SC, modified TD3(p) [28], and slide RL [29]. The modified TD3 with a pre-trained model in [28], denoted as modified TD3(p), is TD3 with reward shaping, which uses the parameters of the trained model from a specific environment for initialization. The sliding mode RL was configured to have the same network structures and reward shaping as the modified TD3(p). Table 3 shows the MSE and MSD of the sliding mode RL and existing algorithms. The sliding mode RL provides the best MSE performance for all cases. The comparison with the modified TD3(p) on which the sliding mode RL is based clearly shows the benefit of the sliding mode, which is gained by simply structuring the control signal. Furthermore, SC outperforms SMC. It is conjectured that, while the non-switching control part may contribute to accelerating the convergence, it may introduce additional uncertainties when the state remains on the sliding surface. The SC can be considered a model-free control since the control signal is generated without explicit model knowledge. Although it required a proper definition of a sliding variable, the sliding mode RL provided better MSE than the SMC, which is based on model knowledge. This suggests that the actor network of sliding mode RL learned how to reduce the error more efficiently.

**Table 3.** Performance of the sliding mode RL for the various system configurations.

Metrics	Case\ $k_u$	1	2	3	4	5	6
MSE	SMC	0.00043	0.00041	0.00060	0.00050	0.00033	0.00069
	SC	0.00027	0.00028	0.00029	0.00027	0.00024	0.00031
	modified TD3(p)	0.04103	0.05078	0.18825	0.13413	0.02477	0.02609
	slide RL	0.00028	0.00021	0.00024	0.00022	0.00022	0.00030
	sliding mode RL	0.00014	0.00011	0.00010	0.00011	0.00011	0.00013
MSD	SMC	0.00269	0.00228	0.00802	0.00939	0.00359	0.00370
	SC	0.00069	0.00084	0.00257	0.00332	0.00217	0.00151
	modified TD3(p)	0.30884	0.36008	5.59826	4.76610	0.80414	0.99561
	slide RL	0.00361	0.00443	0.01388	0.01190	0.01442	0.01004
	sliding mode RL	0.00381	0.00397	0.01269	0.01236	0.01475	0.00965

While the sliding mode RL provides the best MSE performance, the SC is found to provide the best MSD performance for all cases. The sliding mode RL and the slide RL had similar MSD performance for all cases. Their MSDs were more than five times

larger than those of the SC for all cases except case 4. The superior performance of SC is conjectured to be attributed to the nature of switching control without regard to any other information. Further theoretical analysis on the role of switching control remains a topic for future research. Figures 4 and 5 show the box plots of the MSE and MSD, respectively. The sliding mode RL is observed to provide consistent MSE performance for various system configurations similar to the SMC, whereas the modified TD3(p) shows significant dependency on the initialization. SC and slide RL also perform consistently, although the consistency seems to depend on the system configuration. Figure 5 shows that all the consensus algorithms have a larger variance in MSD than in MSE. For the same algorithm, a small variance in MSE does not imply a small one in the MSD. Furthermore, the consistency in the MSD performance depends on the system configuration for all the algorithms considered.

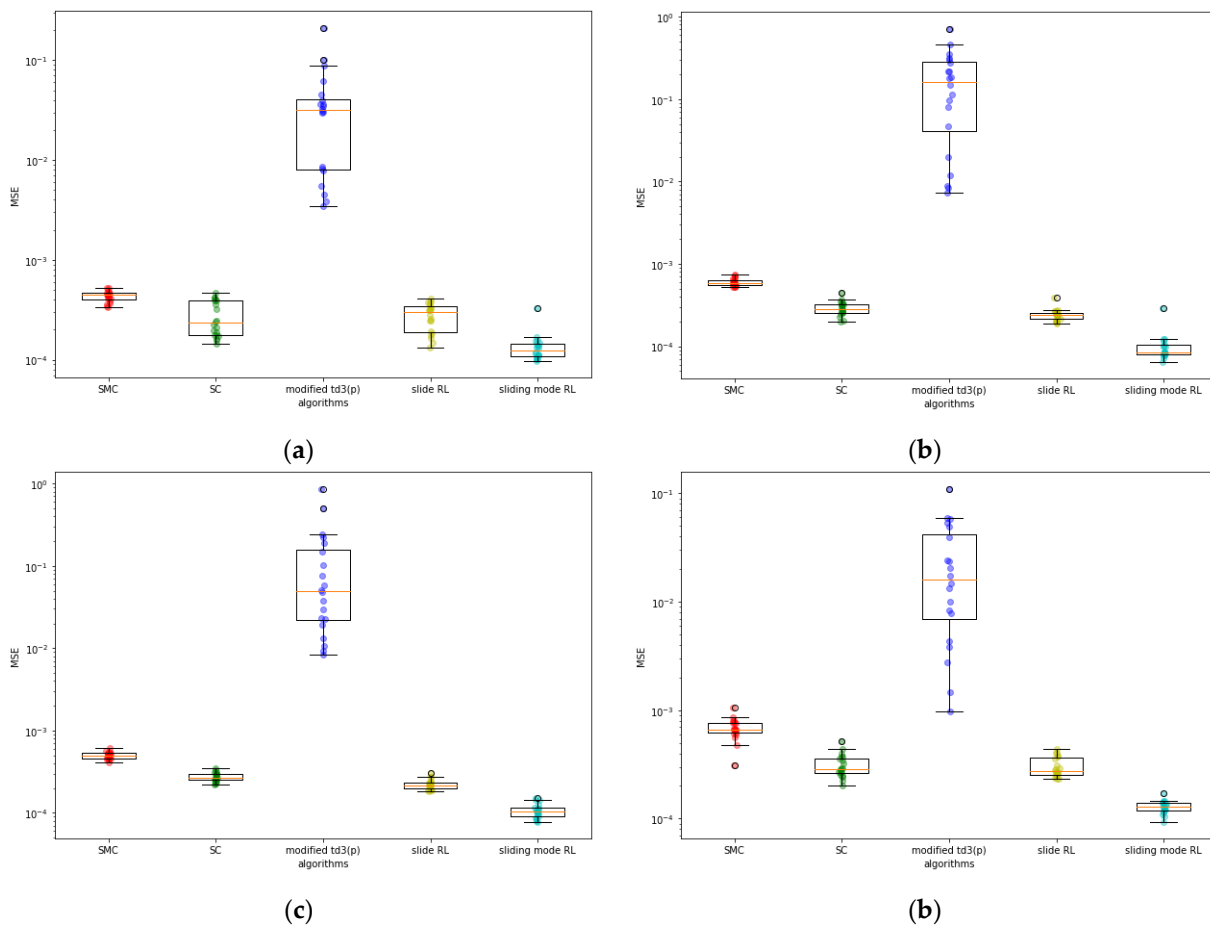
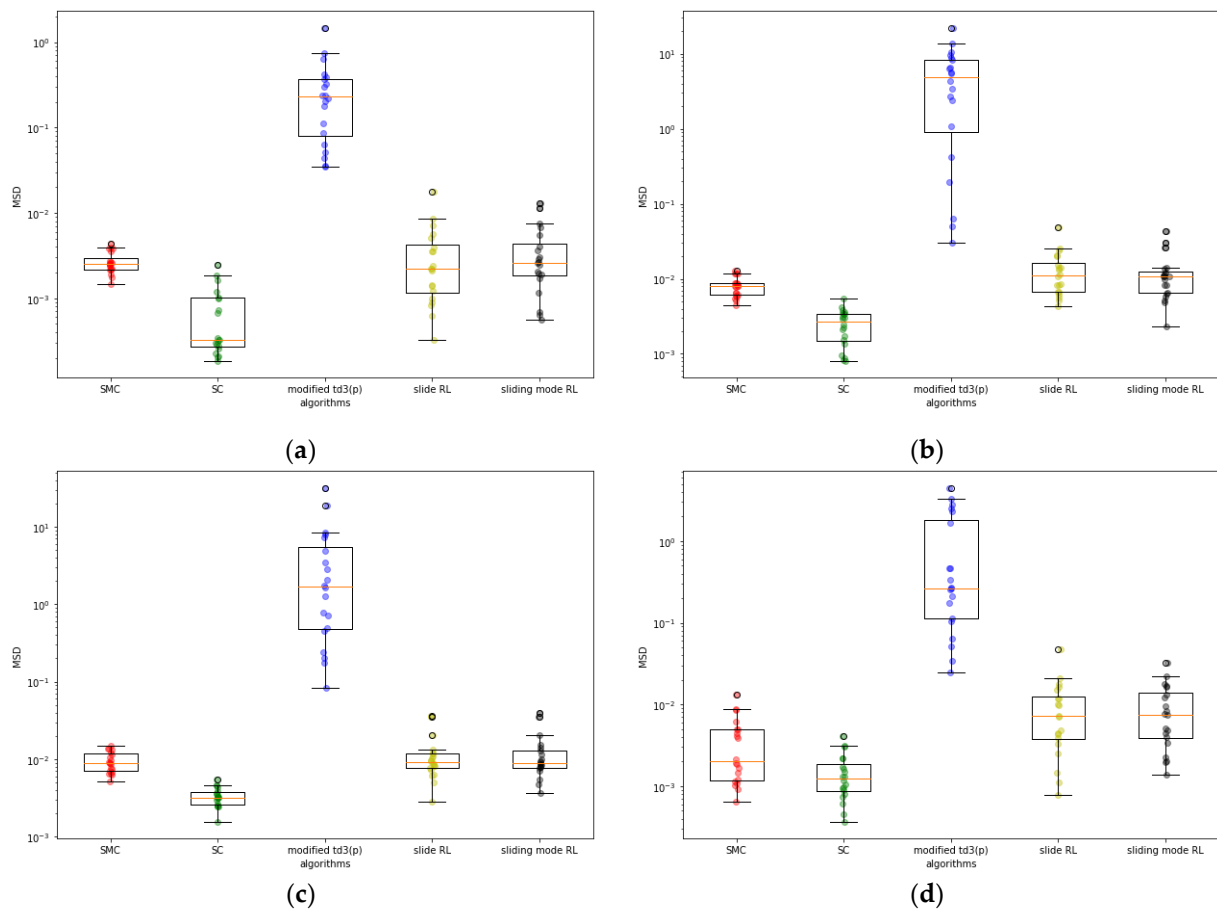


Figure 4. Box plot of MSEs: (a) case1; (b) case3; (c) case4; (d) case6.





**Figure 5.** Box plot of MSDs: (a) case1; (b) case3; (c) case4; (d) case6.

## 5. Conclusions

In this study, sliding mode RL was developed by introducing a sliding variable and structured control into the modified TD3. Despite the very minor change in the modified TD3, the sliding mode RL performed significantly better than the modified TD3(p), which was the modified TD3 with pre-training. It also outperformed the SMC which used explicit model knowledge for MSE performance while it provided the best MSE among the considered consensus algorithms. It also showed that the MSD performance was comparable to that of the SMC with model knowledge.

Many interesting future research directions can be identified from the numerical experiments. The SC which may be considered as a model-free algorithm provided the best MSD performance. In addition, the sliding mode RL and slide RL significantly improve the baseline algorithm by simply introducing a sliding variable and the corresponding switching control structure. From these results, it is conjectured that one may improve the performance of RL by formulating the problem such that a proper sliding variable can be defined. In other words, the phantom of the sliding mode in RL needs to be elucidated in future research. Beyond the scope of RL, the superior performance of the SC over the SMC requires further investigation. It is conjectured that, while the non-switching control part may aid in accelerating convergence, it may work as additional uncertainties when the state remains on the sliding surface. However, a more detailed theoretical explanation may result in an opportunity to develop better nonlinear control. MAS has been proven to achieve consensus over a class of switching graphs [31]. The convergence of MAS is known to depend on the second-smallest eigenvalue of the Laplacian matrix [10]. Uncertainties are likely to have an effect on convergence speed of the consensus of MAS over graphs with time-varying connectivity. Thus, developing an RL-based consensus algorithm to

accelerate the convergence of MAS over a graph with time-varying connectivity in the presence of uncertainties needs to be considered in future research.

**Funding:** This research was funded by the Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Science and ICT under Grant NRF-2017R1A2B4007398, and the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2020S1A5A2A03045921).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Sinopoli, B.; Sharp, C.; Schenato, L.; Schaffert, S.; Sastry, S.S. Distributed Control Applications within Sensor Networks. *Proc. IEEE* **2003**, *91*, 1235–1246. [[CrossRef](#)]
2. Zhang, X.; Papachristodoulou, A. A distributed PID controller for network congestion control problems. In Proceedings of the American Control Conference, Portland, OR, USA, 4–6 June 2014.
3. Jia, D.; Krogh, B. Min-Max Feedback Model Predictive Control for Distributed Control with Communication. In Proceedings of the American Control Conference, Anchorage, AK, USA, 8–10 May 2002.
4. Ding, D.; Han, Q.; Wang, Z.; Ge, X. A Survey on Model-Based Distributed Control and Filtering for Industrial Cyber-Physical Systems. *IEEE Trans. Ind. Inform.* **2019**, *15*, 2483–2499. [[CrossRef](#)]
5. Dorri, A.; Kanhere, S.S.; Jurdak, R. Multi-Agent Systems: A Survey. *IEEE Access* **2018**, *6*, 28573–28593. [[CrossRef](#)]
6. Jaimes, A.; Jamshidi, M.M. Consensus-based and network control of UAVs. In Proceedings of the 5th International Conference on System of Systems Engineering, Loughborough, UK, 1–6 June 2010.
7. Liu, Y.; Zhai, C.; Gao, H.; Chen, L. Consensus of autonomous vehicle platoon with time delays. In Proceedings of the Chinese Automation Congress (CAC), Jinan, China, 20–22 October 2017; pp. 4598–4603.
8. Contzen, M.P.; Raisch, J. Reactive power consensus in microgrids. In Proceedings of the European Control Conference (ECC), Aalborg, Denmark, 29 June–1 July 2016; pp. 334–339.
9. Yan, J.; Yang, X.; Luo, X.; Chen, C.; Guan, X. Consensus of Teleoperating Cyber-Physical System via Centralized and Decentralized Controllers. *IEEE Access* **2017**, *5*, 17271–17287. [[CrossRef](#)]
10. Olfati-Saber, R.; Murray, R.M. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. Autom. Control* **2004**, *49*, 1520–1533. [[CrossRef](#)]
11. Duan, G.; Xiao, F.; Wang, L. Asynchronous Periodic Edge-Event Triggered Control for Double-Integrator Networks with Communication Time Delays. *IEEE Trans. Cybern.* **2018**, *48*, 675–688. [[CrossRef](#)]
12. Wang, C.; Zuo, Z.; Qi, Z.; Ding, Z. Predictor-Based Extended-State-Observer Design for Consensus of MASs with Delays and Disturbances. *IEEE Trans. Cybern.* **2019**, *49*, 1259–1269. [[CrossRef](#)]
13. Wang, D.; Zhang, N.; Wang, J.; Wang, W. Cooperative Containment Control of Multiagent Systems Based on Follower Observers with Time Delay. *IEEE Trans. Syst. Man Cybern. Syst.* **2017**, *47*, 13–23. [[CrossRef](#)]
14. Kim, J.M.; Park, J.B.; Choi, Y.H. Leaderless and leader-following consensus for heterogeneous multi-agent systems with random link failures. *IET Control Theory Appl.* **2014**, *8*, 51–60. [[CrossRef](#)]
15. Zhang, Q.; Niu, Y.; Wang, L.; Shen, L.; Zhu, H. Average consensus seeking of high-order continuous-time multi-agent systems with multiple time-varying communication delays. *Int. J. Control Autom. Syst.* **2011**, *9*, 1209–1218. [[CrossRef](#)]
16. Wu, J.; Shi, Y. Consensus in multi-agent systems with random delays governed by a Markov chain. *Syst. Control Lett.* **2011**, *60*, 863–870. [[CrossRef](#)]
17. Sun, Y.-J.; Zhang, G.-L.; Zeng, J. Consensus Analysis for a Class of Heterogeneous Multiagent Systems with Time Delay Based on Frequency Domain Method. *Math. Probl. Eng.* **2014**, *2014*, 248684. [[CrossRef](#)]
18. Pisano, A.; Usai, E. Sliding mode control: A survey with applications in math. *Math. Comput. Simul.* **2011**, *81*, 954–979. [[CrossRef](#)]
19. Ghayoomi, P.; Ghasemi, R. Observer based sliding mode consensus controller design for nonlinear multi-agent systems. In Proceedings of the International Conference on Inventive Systems and Control (ICISC), Coimbatore, India, 19–20 January 2017.
20. Zhao, N.; Zhu, J.-D. Robust Consensus Problem of Heterogeneous Uncertain Second-Order Multi-Agent Systems Based on Sliding Mode Control. *Front. Control Eng.* **2021**, *2*, 744027. [[CrossRef](#)]
21. Yuan, L.; Li, J. Consensus of Discrete-Time Nonlinear Multiagent Systems Using Sliding Mode Control Based on Optimal Control. *IEEE Access* **2022**, *10*, 47275–47283. [[CrossRef](#)]
22. Wang, C.; Wen, G.; Peng, Z.; Zhang, X. Integral Sliding-Mode Fixed-Time Consensus Tracking for Second-Order Non-Linear and Time Delay Multi-Agent Systems. *J. Frankl. Inst.* **2019**, *356*, 3692–3710. [[CrossRef](#)]
23. Yang, J. A Consensus Control for a Multi-Agent System with Unknown Time-Varying Communication Delays. *IEEE Access* **2021**, *9*, 55844–55852. [[CrossRef](#)]

24. Zhang, H.; Jiang, H.; Luo, Y.; Xiao, G. Data-Driven Optimal Consensus Control for Discrete-Time Multi-Agent Systems with Unknown Dynamics Using Reinforcement Learning Method. *IEEE Trans. Ind. Electron.* **2017**, *64*, 4091–4100. [[CrossRef](#)]
25. Wang, X.; Su, H. Completely model-free RL-based consensus of continuous-time multi-agent systems. *Appl. Math. Comput.* **2020**, *382*, 125312. [[CrossRef](#)]
26. Li, J.; Ji, L.; Li, H. Optimal consensus control for unknown second-order multi-agent systems: Using model-free reinforcement learning method. *Appl. Math. Comput.* **2021**, *410*, 126451. [[CrossRef](#)]
27. Yang, J. Deep Learning-Based Consensus Control of a Multi-Agents System with Unknown Time-varying Delay. *Electronics* **2022**, *11*, 1176. [[CrossRef](#)]
28. Yang, J. Reinforcement Learning for the Consensus of Multi-agents with Unknown Time Varying Delays. *J. Digit. Contents Soc.* **2022**, *23*, 1277–1287. [[CrossRef](#)]
29. Yang, J. A slide reinforcement learning for the consensus of a multi-agents system. *J. Adv. Navig. Technol.* **2022**, *26*, 226–234.
30. Fujimoto, S.; Hoof, H.; Meger, D. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceeding of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018*.
31. Chowdhury, N.R.; Sukumar, S.; Chatterjee, D. A new condition for asymptotic consensus over switching graphs. *Automatica* **2018**, *97*, 18–26. [[CrossRef](#)]