


Turbulence control for drag reduction through deep reinforcement learningTaehyuk Lee and Junhyuk Kim **Department of Mechanical Engineering, Yonsei University, Seoul 03722, Korea*Changhoon Lee †*Department of Mechanical Engineering, Yonsei University, Seoul 03722, Korea
and School of Mathematics and Computing, Yonsei University, Seoul 03722, Korea*

(Received 13 August 2022; accepted 20 January 2023; published 8 February 2023)

Deep reinforcement learning (DRL) was applied to turbulence control for drag reduction in direct numerical simulations of turbulent channel flow. Based on the wall shear stress information only, the DRL is capable of determining the optimal distribution of wall blowing and suction, which can reduce drag by 20%, comparable with previous wall-shear-based controls such as suboptimal control [Lee *et al.*, *J. Fluid Mech.* **358**, 245 (1998)] and neural-network-based control [Lee *et al.*, *Phys. Fluids* **9**, 1740 (1997)]. However, our DRL-based control can determine the optimal amplitude of the wall actuation, which was not possible in previous controls. More importantly, from an analysis of the optimal actuation fields, two distinct types of drag reduction mechanisms are identified: the first cancels the near-wall sweep and ejection events, whereas the second mechanism suppresses the streamwise vortices near the wall, which is similar to that of the suboptimal control. This study demonstrates the successful application of DRL to turbulence control and its physical interpretation.

DOI: [10.1103/PhysRevFluids.8.024604](https://doi.org/10.1103/PhysRevFluids.8.024604)**I. INTRODUCTION**

Flow control has been used in a wide range of engineering applications for its economic benefits [1–3]. In particular, turbulence control for skin-friction reduction using wall blowing and suction has been a hot issue, and thus it has been actively investigated both numerically and experimentally [1,4–11]. More than 25 years ago, a series of attempts were made to find an optimum distribution of the wall blowing and suction to reduce the skin-friction drag in the direct numerical simulation of turbulent channel flow [4,6,7]. Choi *et al.* [4] proposed the so-called “opposition control,” which applies wall blowing and suction in the opposite direction to the vertical velocity at $y^+ = 10$ with the same magnitude to cancel near-wall fluid motion caused by the streamwise vortices, resulting in 25% reduction in low-Reynolds-number channel flows. A more implementable control scheme based on the wall shear stress or pressure was sought using more systematic approaches. For example, a neural network [6] and suboptimal control theory [7] were applied to find the blowing and suction distributions based on the quantities sensed at the wall only, and both attempts yielded almost the same explicit control scheme represented by

$$\hat{\phi}(\kappa_x, \kappa_z) = C \frac{i\kappa_z}{\kappa} \frac{\partial \widehat{w}}{\partial y} \Big|_w = \frac{C}{\kappa} \frac{\partial}{\partial z} \left(\frac{\partial \widehat{w}}{\partial y} \right)_w, \quad (1)$$

*Present address: Korea Atomic Energy Research Institute, Daejeon 34057, Korea.

†clee@yonsei.ac.kr

where $\hat{\phi}$ is the Fourier coefficient of wall blowing and suction. $\kappa = \sqrt{\kappa_x^2 + \kappa_z^2}$ with κ_x and κ_z denoting the streamwise and spanwise wave numbers, respectively. $\frac{\partial w}{\partial y}|_w$ is the Fourier coefficient of spanwise wall shear stress. The derivation of Eq. (1) is provided in Appendix A. Using the Fourier convolution formula, the optimal wall actuation in physical space $\phi(x_i, z_k)$ can be expressed as the weighted sum of the local spanwise wall shear stress [7]. Here, C is undetermined, and the optimum value is found to maintain the given root-mean-square (rms) value of the actuation [6,9]. The physical implications of Eq. (1) is that the wall blowing and suction counteracts the near-wall wall-normal fluid motion associated with a streamwise vortex. Although these control methods reduce drag substantially, it is not certain that these are the most optimal control schemes based on the wall shear information guaranteeing maximum drag reduction because only the linear terms in the adjoint Navier-Stokes equation were considered in the suboptimal formulation, and the short-time response was considered in both the suboptimal formulation [7] and neural-network application [6].

Due to the recent recognition of machine learning as a potentially useful technique for representing nonlinear systems, various machine learning algorithms have been actively applied to a wide range of turbulence problems; see some examples from our group [12–14] and recent reviews [15–18]. In an attempt to apply the idea of opposition control [4], Han and Huang [19] and Park and Choi [20] applied supervised learning to predict the vertical velocity at $y^+ = 10$ using the wall-sensing quantities in an unmanipulated channel and then implemented the trained network for control in the test environment under the assumption that the strong correlation between the quantities obtained in an unmanipulated environment persisted in the test environment as well. Although this approach yielded up to 20% drag reduction, it does not guarantee successful control in general.

By contrast, deep reinforcement learning (DRL) is known to be capable of learning without prior assumptions. This algorithm enables the *agent* to determine the optimal *state* by recognizing and acting autonomously in a *manipulated* environment while interacting with the environment in a real-time trial-and-error process [21]. In other words, by incorporating feedback into machine learning, similar to closed-loop control, it is possible to overcome noise from the environment by compensating for deviations caused by perturbations and errors [22]. DRL has been applied to fluid mechanics in a variety of fields, including flow control [3,23,24], homogeneity optimization [25], shape optimization [26], heat transfer control [27], chaotic systems [28], and large-eddy simulation modeling [29,30]. The DRL algorithm performed well in both simulations and actual experiments [31].

In this study, a reinforcement learning algorithm called twin delayed deep deterministic policy gradient (TD3) was utilized to determine the optimum distribution of wall blowing and suction to reduce drag based on the streamwise and spanwise wall shear stresses in a simulated turbulent channel flow environment. Successful training leads to substantial drag reduction comparable to opposition control [4] or suboptimal control [6], confirming that suboptimal control is indeed close to optimal among wall-information-based controls for drag reduction. Furthermore, an investigation of the optimal distribution of wall blowing and suction yields the discovery of two different mechanisms of drag reduction. One of them is similar to the mechanism of suboptimal control [6] or opposition control [4], while the other mechanism is to cancel the sweep and ejection events, which has not been reported before. Recently, Sonoda *et al.* [32] reported an application of reinforcement learning to turbulence control for drag reduction in channel turbulence, which is similar to our study. They attempted to extend the opposition control method by finding the more general control scheme of the wall blowing and suction as a nonlinear function of the streamwise and wall-normal fluctuation velocities at $y^+ = 15$ using reinforcement learning. They obtained up to 37% drag reduction in channel turbulence at $Re_\tau = 150$. However, their reinforcement-learning-based algorithm requires the detection of flow information inside the domain as input, restricting real implementation. Furthermore, the drag reduction mechanism of their successful control is quite different from ours, as will be discussed later.

TABLE I. DNS parameters for turbulent channel flow.

Re_τ	N_x	N_y	N_z	Δx^+	Δy^+	Δz^+	Δt^+
180	128	193	128	17.67	Min: 0.13, Max: 4.73	8.84	0.018
360	256	193	256	17.67	Min: 0.26, Max: 9.45	8.84	0.036

The remainder of this paper is organized as follows. Section II introduces the TD3 algorithm features and numerical method for the simulation environment. Subsequently, a network and deep reinforcement learning framework is proposed. In Sec. III, we discuss the convergence of DRL, the performance of each model, and the mechanism for the drag reduction of each model. A brief conclusion is provided in Sec. IV.

II. METHODOLOGY

A. Environment simulation for DRL

Turbulent channel flow is selected as the DRL environment for DRL. We solve the incompressible Navier-Stokes and continuity equations as follows:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\nabla p + \frac{1}{Re_\tau} \nabla^2 \mathbf{u}, \quad (2)$$

$$\nabla \cdot \mathbf{u} = 0, \quad (3)$$

where \mathbf{u} and p are the velocity vector and pressure, respectively. x , y , and z denote the streamwise, wall-normal, and spanwise directions, respectively. The equations are normalized by the channel half-gap δ and the wall shear velocity u_τ . The friction Reynolds number $Re_\tau = u_\tau \delta / \nu$, where ν denotes the kinematic viscosity. For the computational domain of $(4\pi \times 2 \times 2\pi)\delta$, the periodic condition is applied in the streamwise and spanwise directions, and the no-slip condition is enforced at the bottom and top walls. At the wall, a distribution of blowing and suction is prescribed for the boundary condition of the normal component of velocity. An optimum distribution of blowing and suction $\phi(x, z)$ was sought through deep reinforcement learning. For the spatial discretization, a spectral method is adopted in the streamwise and spanwise directions, and second-order finite differencing is applied in the wall-normal direction over the nonuniform grids given by $y_j = -\tanh[\alpha(1 - 2j/N_y)]/\tanh \alpha$ for $j = 0, 1, \dots, N_y$ with $\alpha = 2.5$. For the time advancement, Adams-Bashforth and Crank-Nicholson schemes are adopted for the convection terms and viscous terms, respectively. The number of grids, grid size, and integration time step in the wall unit are provided in Table I for two Reynolds numbers considered in this study.

B. DRL structure and learning framework

A reinforcement learning algorithm efficiently solves an optimization problem through dynamic programming, where the *agent* interacts with the *environment*. As illustrated in Fig. 1, in our application, the environment corresponds to direct numerical simulation of channel turbulence, and the agent attempts to provide optimal *action* to the environment based on the sensed *state* depending on the behavior of the feedback *reward* through repeated learning. The agent's learning is implemented by employing the Markov decision process (MDP) model.

For the action a_t for drag reduction in channel turbulence, we chose the wall blowing and suction $\phi(x, z)$, which is implemented in the channel environment as a boundary condition for the wall-normal component of velocity. For the state s_t , from which the optimal action is sought by the agent, the wall shear stresses ($\frac{\partial u}{\partial y}|_{\text{wall}}, \frac{\partial w}{\partial y}|_{\text{wall}}$) were selected. However, because the implementation of the action and the detection of the state are performed at the same place, numerical instability is frequently observed. To prevent this, the shear stresses were sensed at a plane slightly away from

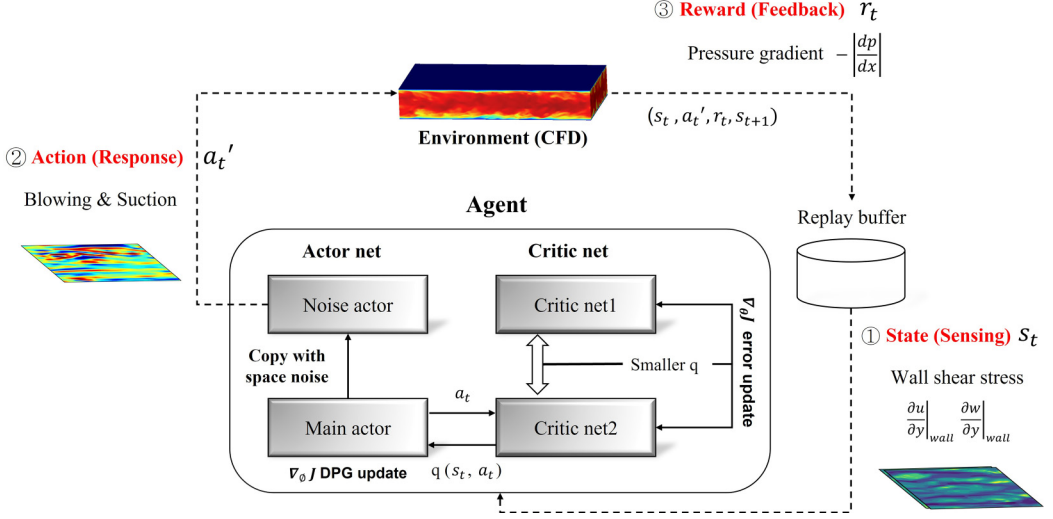


FIG. 1. Structure of TD3 algorithm with parameter space noise. Critic net consists of two identical networks, Critic net1 and Critic net2. By comparing the q predicted in Critic net1 and Critic net2, the two critic weights (θ) are updated using the smaller q of the two. For the efficient exploration, we form a noise actor (exploration policy) by adding Gaussian noise (exploration noise) to the weight (ψ) of the main actor (main policy). Here, $a_t' = a_t$ with space noise.

the wall [19], that is, at $y^+ \sim 2.1$, which is a little further than $y^+ = 0.55$ used by [19] because they considered only the spanwise shear stress, which is less sensitive than the streamwise shear stress. For the reward $r_t(s_t, a_t)$ to be maximized, the negative value of the absolute pressure gradient under the constant mass flow rate condition is chosen as $r_t = -|\frac{dp}{dx}|$. In a real implementation, however, the n -step reward $r_t^d = \sum_{j=1}^n \gamma^{j-1} r_{t+j}$ [33] was considered to reflect the gradual fading of the effect of the action over time. Here, one step means one state step, which is the sensing period of the state information, and γ (<1) is called the discounted factor. γ and n are the two most critical hyperparameters of a learning network. Finally, transition probability refers to an element of uncertainty in the environment that we have no control over [33].

As shown in Fig. 1, the agent, the most essential element of a reinforcement learning, observes the state and takes action in accordance with the agent's decision-making policy $\pi(s_t)$, which determines the probability of the agent taking the optimal action. The agent is rewarded by the environment prior to changing states. At this point, the value of the previous state is recalculated via the reward by the action, and the key step in reinforcement learning is to update the behavior policy based on the updated state value. There are numerous reinforcement learning models, and each model differs in terms of how the agent system learns the state and the reward received from the environment. In this study, we adopted an actor-critic policy gradient model called the twin-delayed deep deterministic policy gradient (TD3) model, which improves the stability of learning and performance by adding three elements to the deep deterministic policy gradient (DDPG) algorithm [34]. First, the action value function $q_\pi(s_t, a_t)$ is sought to satisfy the Bellman equation,

$$q_\pi(s_t, a_t) = \mathbb{E}[r_t^d + \gamma^n q_\pi(s_{t+n}, \pi_\phi(s_{t+n}) + \epsilon)], \quad (4)$$

with a delayed policy update $\pi_\phi(s_{t+n})$, which increases the stability of learning. Second, target policy smoothing regularization by adding clipped random noise ϵ to target q prevents overfitting by narrow peaks. Here, $\epsilon \sim \text{Clip}(\mathcal{N}(0, \sigma_\epsilon), -c, c)$, where \mathcal{N} is the Gaussian distribution, and c is the clipping range. Third, double q -learning is adopted for the critic network. It is generally known that the approximated q is larger than the true q ; therefore, the error occurrence corresponds to the

difference. To reduce the difference, two critical values were used, and the smaller value of the two was updated. This reduces the error caused by approximation [35]. The corresponding objective function is as follows:

$$J(\theta) = N^{-1} \sum [r_t^d + \gamma^n q_\pi(s_{t+n}, \pi_\phi(s_{t+n}) + \epsilon) - q(s_t, a_t)]^2, \quad (5)$$

where N is the minibatch size, and θ is the critic's weight parameter. The actor net serves as a policy function $\pi(s_t)$ for determining the optimal course of action. The policy objective function $J(\phi)$ is updated using

$$\nabla_\psi J(\psi) = \mathbb{E}[\nabla_a q_\pi(s_t, a_t) |_{a=\pi(s_t)} \nabla_\psi \pi_\psi(s_t)], \quad (6)$$

where ψ denotes the weight parameter of the actor.

Training was performed using four different initial fields, and a validation test was conducted in another field. One state step in the DRL corresponds to 50 simulation time steps to ensure that the effect of the action is sufficiently reflected in the environment before it is fed back to the state. Additionally, we employed a soft weight update using the $\theta' = \tau\theta + (1 - \tau)\theta'$ formula [34] with τ set to 0.001. Each episode is 1200 state steps long, and when utilizing the TD3 algorithm, the delayed update periods for the target and policy networks must be specified. It was updated once every five state steps in the study.

In the numerical implementation of a reinforcement learning, there are a couple of strategies to boost computational performance including the use of multi-environments [36]. In many cases, the CFD part consumes the most computation time. Therefore, we chose to use the CUDA parallelization technique on GPU for CFD, which speeds up simulation time by 20 times compared with that by the openMP-based CPU parallelization on 32 processors for $128 \times 192 \times 128$ grids. We observed that on the GPU server with Titan XP graphic cards, the whole reinforcement learning including CFD and DRL took 0.202 s per time step on average, among which the CFD part consumes around 0.180 s, which accounts for approximately 89% of the overall computation. A typical training of each model took two days.

Our codes, including the CFD part explained in the previous section and the DRL part, will be released as an open source at a public repository [37] upon publication of our paper.

C. Exploration noise

Since our model is based on the deterministic policy gradient model [34,38], an exploration noise strategy is necessary to avoid becoming stuck in the local optimum. This critical technique can be categorized into two methods: action-space noise and parameter-space noise. The former adds noise to the policy itself. Adding uncorrelated noise (white noise) to the policy (actor) can increase the agent's exploration diversity, whereas adding temporally correlated noise such as the Ornstein-Uhlenbeck (OU) process-based noise [39,40] can create a policy that can be explored naturally under physical constraints. The latter method injects Gaussian noise into the weight parameters of the actor network. This enables the agent to acquire sparse rewards by exploring an environment that is difficult to access [41]. It is known that parameter space noise outperforms traditional action noise and performs better in high-dimensional actions and continuous environments. Therefore, we adopted space parameter noise as the exploration noise.

For the appropriate intensity of noise to vary to reflect the changing environment, adaptive scaling for the standard deviation of noise σ_t was implemented as follows:

$$\sigma_{t+1} = \begin{cases} \alpha \sigma_t & \text{if } d(\pi, \tilde{\pi}) < \lambda, \\ \frac{1}{\alpha} \sigma_t & \text{otherwise,} \end{cases} \quad (7)$$

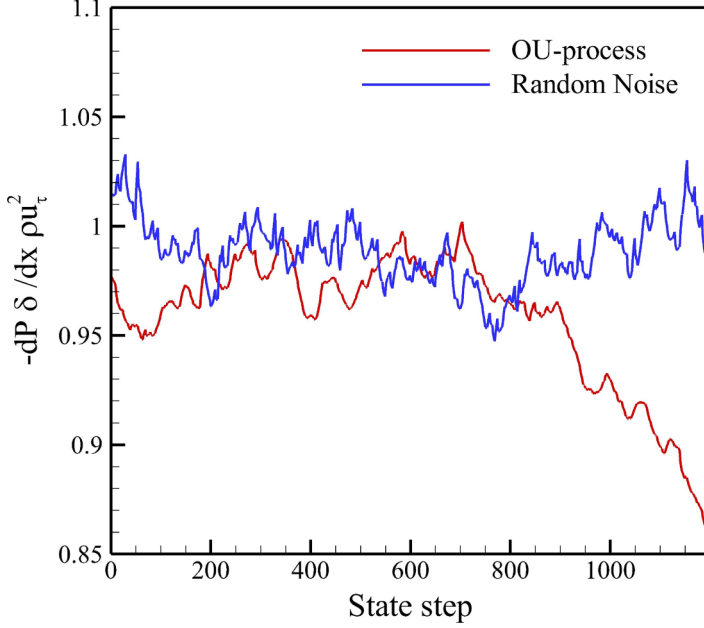


FIG. 2. Comparison of the learning behavior with respect to the noise introduced by exploration. The blue line depicts the pressure gradient changes when random noise is used as the exploration noise, whereas the red line indicates the pressure gradient when the OU-process is used as the exploration noise.

with

$$d(\pi, \tilde{\pi}) = \sqrt{\frac{1}{N} \sum_{i=1}^N \mathbb{E}[\pi(s_t)_i - \tilde{\pi}(s_t)_i]^2}, \quad (8)$$

where α is the scaling factor. $\tilde{\pi}$ and λ are the perturbed policy (noise actor) and threshold value, respectively. N is the minibatch size, and $d(\pi, \tilde{\pi})$ denotes the distance between the perturbed policy and the main policy. Therefore, the noise intensity varies automatically in proportion to the difference between the perturbed and main policies.

Two types of noise were tested: random-walk noise, which is not correlated in time, and temporally correlated OU-process noise, generated by

$$x_t^{\text{RW}} = \mathcal{N}(\mu, \sigma_t), \quad (9)$$

$$dx_t^{\text{OU}} = \xi(\mu - x_t^{\text{OU}})dt + \sigma_t dW_t, \quad (10)$$

where μ and ξ are the mean noise and the mean reversion constant, respectively, and W_t is the Wiener process characterized by an increment $dW_t = \mathcal{N}(0, \sqrt{dt})$. Because random-walk noise is not correlated in time, it explores more diverse experiences at every state step, and higher rewards are likely to be obtained. Therefore, finding learning conditions is relatively straightforward. However, as illustrated in Fig. 2, the reward sometimes exhibits nonphysical phenomena, such as a highly oscillatory behavior in the mean pressure gradient. Furthermore, learning convergence is not guaranteed.

However, because the OU-process noise is correlated in time, it explores less diverse experiences than random walk noise. The nonphysical behavior of the reward observed for the random-walk noise, however, was not found in the mean pressure gradient, as shown in Fig. 2. Most importantly,

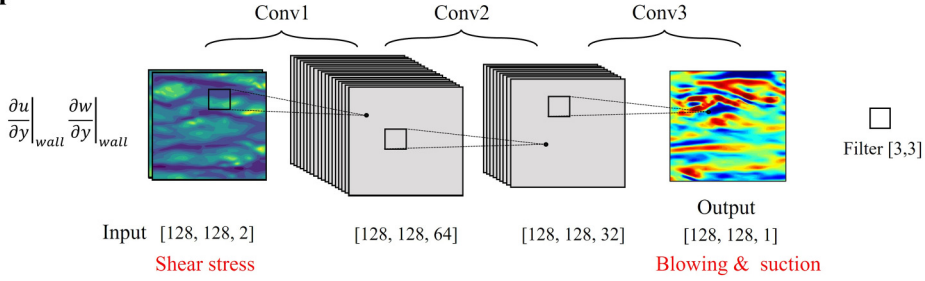
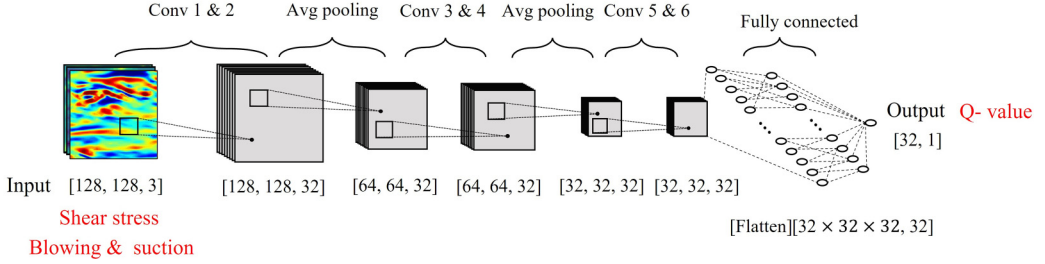
Actor

Critic


FIG. 3. Structures of the actor and critic networks. The actor network consists of only convolutional layers, and the critic network consists of convolutional layers and fully connected layers. Here, Conv denotes convolutional layers, and Avg pooling denotes average pooling.

with OU-process noise, the learning converges very well. Therefore, OU-process noise was adopted in this study.

Because the noise actor injects Gaussian noise into the main actor’s weight, the noise actor must be updated whenever the main actor is updated. Therefore, the update frequency of a noise actor must be specified. We set the noise actor’s update frequency to 5 to match the target update frequency. The optimum values of the network hyperparameters are listed for the three types of DRLs considered in this study, which will be explained in the following section.

D. Actor-critic network structure

Figure 3 shows the actor and critic network architectures within the TD3 structure. The actor net consists of three convolutional layers, and each layer retains the same spatial dimension of 128×128 as the input data $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$, resulting in an output (blowing and suction) in the same dimension, which is applied as an action at the wall. The choice to use convolutional layers was made based on the feasible assumption that the optimal actuation at a point is determined by nearby wall shear stress distribution and the observation that the optimal policy would be translationally invariant in the fully developed channel flow. (This is similar to the method “M2” of Ref. [42].) However, due to this translational invariance of the policy, the separate consideration of local rewards in learning such as the method “M3” of [42] was not necessary. Here, the input fields and output actuation are normalized by the mean streamwise shear stress $\langle \frac{\partial u}{\partial y} \rangle$ and u_τ of the unmanipulated flow, respectively. The first, second, and last layer filters consist of 64 kernels in $3 \times 3 \times 2$, 32 kernels in $3 \times 3 \times 64$, and a kernel in $3 \times 3 \times 32$. Three consecutive applications of a convolution layer with a filter size of 3×3 to input data for the output actuation immediately indicates that the blowing and suction at a point are determined by nearby 7×7 data of the input shear stresses. Utilizing more hidden convolution layers with a larger input domain did not improve the learning performance, indicating that the three convolution layers are optimal. The corresponding size of the input data is 124×62 wall unit lengths in the streamwise and spanwise directions.

TABLE II. Learning parameters for each model. Notably, all dimensional parameters are nondimensionalized by u_τ and δ .

Learning parameter	DRL- $(\frac{\partial u}{\partial y})$	DRL- $(\frac{\partial w}{\partial y})$	DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$
n of n -step reward	5	5	5
Discounted factor, γ	0.95	0.95	0.99
Replay buffer memory	30 000	30 000	30 000
Batch size, N	64	64	64
Disturbed scaling distance, λ	0.03	0.03	0.04
Critic smoothing clip, c	0.3	0.3	0.3
Standard deviation of smooth target q /action rms, σ_q	0.2	0.2	0.2
Starting standard deviation for exploration noise, σ_0	0.1	0.1	0.1
One state step/simulation time step	50	50	50
One episode state steps	1200	1200	1200
Reciprocal of mean reverting constant in wall unit, $(1/\xi)^+$	4.5	4.5	1.8
Action output coefficient	0.5	0.1	0.5
Disturbed distance adaptive scaling factor, α	1.01	1.01	1.01
Soft update critic, τ	0.001	0.001	0.001
Actor network learning rate	0.0001	0.0001	0.0001
Critic network learning rate	0.0002	0.0002	0.0002
Critic network regularization	0.0001	0.0001	0.0001
Actor update frequency	5	5	5
Target update frequency	5	5	5
Noise actor update frequency	5	5	5

The critic network is composed of six convolutional layers and three fully connected layers. The q value is predicted in this network using state and action information as input data, and for every two convolutional layers, an average pooling layer is added. The input data were provided in $128 \times 128 \times 3$ (state and action). The first-layer filter contains 32 kernels in $3 \times 3 \times 3$, and the remaining convolutional layers contain 32 kernels in $3 \times 3 \times 32$. For both the actor and critic networks, the stride of all layers was one pixel. All the fully connected layers are composed of 32 neurons, and the final output has a q value size of 1. In both networks, the ReLU activation function is applied to all layers.

E. Optimization of the learning process

In this study, we consider three DRL models depending on the choice of input data: DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ was trained using both the streamwise and spanwise wall shear stresses as inputs, whereas DRL- $(\frac{\partial u}{\partial y})$ and DRL- $(\frac{\partial w}{\partial y})$ were trained based on the streamwise and spanwise wall shear stresses, respectively. To compare the performance and analysis, two control models developed in the past were considered: the OPP model for opposition control [4] and the SUB model for suboptimal control based on spanwise wall shear stress [6,7].

Among the many learning parameters listed in Table II, the number of state steps n and discount factor γ of the n -step reward are the most critical factors affecting the learning performance. When n is equal to 1, that is, when a single-step reward is used, learning does not occur in the proper direction to minimize the drag. As n increases, the amount of drag reduced by the DRL increases, but an excessively large n deteriorates performance. The optimum n was found to be five, indicating that five state steps or equivalently 4.5 wall time units are appropriate for reflecting the response of the reward to the action. More importantly, we set the update time of the aforementioned target critic and actor network to the reward accumulation time, five state steps, based on the optimal interval to reflect the behavior of reward to the action. Therefore, there exist two different timescales; the

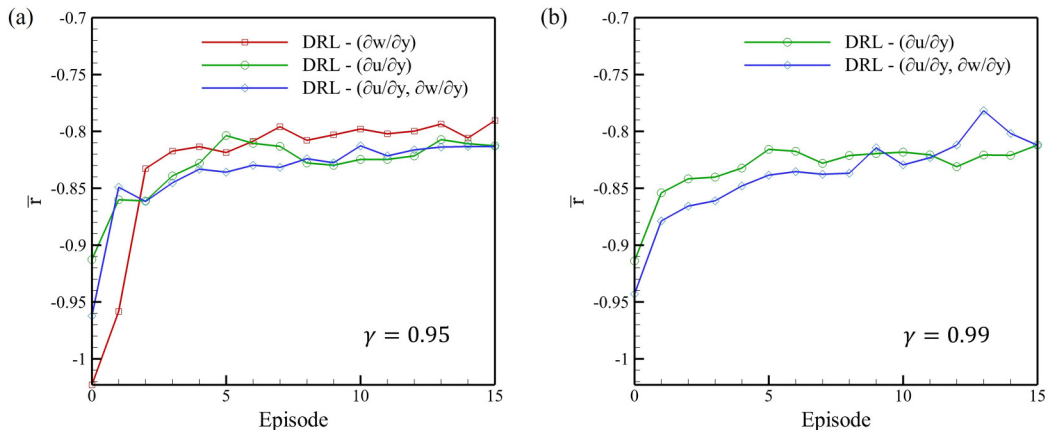


FIG. 4. Learning performance of the normalized reward in terms of episode for (a) $\gamma = 0.95$ and (b) $\gamma = 0.99$.

first one is a fast timescale $\Delta t^+ (=0.018)$, which is a simulation time step for CFD, and the second one is a slow timescale, five state steps ($=250\Delta t^+ = 4.5$), for the action update [36]. Furthermore, the discount factor γ determines the relative weight of the reflection among those five step rewards. The optimum range of γ was found to be 0.95–0.99 depending on the choice of input data.

Figure 4 shows the learning performance in terms of the normalized reward for $\gamma = 0.95$ and 0.99. The normalized reward is defined as $\bar{r} = \sum_{j=1}^n \gamma^{j-1} r_{t+j} / \sum_{j=1}^n \gamma^{j-1}$. When learning was performed with $\gamma = 0.95$, the DRL- $(\frac{\partial w}{\partial y})$ model outperformed the DRL- $(\frac{\partial u}{\partial y})$ and DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ models during the learning procedure. However, when $\gamma = 0.99$, the DRL- $(\frac{\partial w}{\partial y})$ model failed to yield stable results, whereas both the DRL- $(\frac{\partial u}{\partial y})$ and DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ models converged well. Although the training DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ model was slightly unstable in terms of learning convergence, it yielded the highest reward among the tested models. Therefore, γ is fixed at 0.99 for the DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ model, and γ is chosen to be 0.95 for DRL- $(\frac{\partial u}{\partial y})$ and DRL- $(\frac{\partial w}{\partial y})$ models.

In all cases, we trained each model with 16 episodes because 16 episodes turned out to be the optimal metric via trial and error. A consideration of further episodes yielded a model with no significant improvement in performance for some cases or a ruined model for other cases. In the reinforcement learning, there is no assurance that longer learning would result in improvement because the model sometimes tends to worsen due to the catastrophic forgetting [43] or overfitting. Therefore, it is critical to execute an early stopping on the optimal metric.

In terms of time and data usages, our DRL model spent 10.1 s on the GPU server with Titan XP graphic cards and used one set of data per state step, which consists of 50 time steps of the CFD simulation. Here, one set of data includes the field data for the state, action, the next state, and reward with one field data consisting of the two-dimensional field data on 128×128 grids.

III. RESULTS

A. Test performance of the DRL models

Three DRL models and two comparative models, the OPP and SUB models, were tested in two different environments from the one in which the DRL models were trained at $Re_\tau = 180$: one in a different instance at the same Reynolds number, and another at different Reynolds numbers, $Re_\tau = 360$. The drag reduction performance of all models is shown in Fig. 5 in terms of the required pressure gradient to maintain the same flow rate. The amount of averaged drag reduction along with

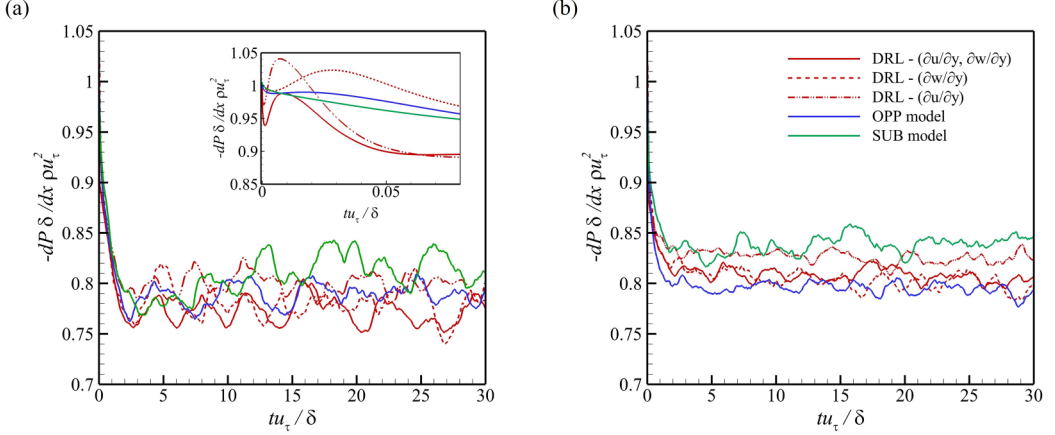


FIG. 5. Test results of performance of the trained DRL models in terms of the mean pressure gradient to maintain a constant flow rate: (a) test results at $Re_\tau = 180$ of the DRL models trained at $Re_\tau = 180$ and (b) test results at $Re_\tau = 360$ of the DRL models trained at $Re_\tau = 180$. u_τ used for nondimensionalization of the pressure gradient and time is that of the unmanipulated flow. The inset shows early behavior of the pressure gradient by all models.

the rms value of the wall actuation is listed in Table III. At the same Re_τ as that of the trained environment, DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ and DRL- $(\frac{\partial w}{\partial y})$ perform the best by reducing the drag by 23%, which is better than the OPP and SUB models, as shown in Fig. 5(a). When the models trained at $Re_\tau = 180$ were tested at $Re_\tau = 360$, the performance of DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ and DRL- $(\frac{\partial w}{\partial y})$ deteriorated slightly. The oscillatory behavior of the pressure gradient over time for $Re_\tau = 180$ is due to the finite size of the tested channel; thus, it is suppressed for the cases with $Re_\tau = 360$, the horizontal domain of which is four times larger in the wall unit than in the cases with $Re_\tau = 180$. The initial behavior of the pressure gradient, shown in the inset of Fig. 5(a), shows an interesting difference between one group of DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ and DRL- $(\frac{\partial w}{\partial y})$ and another group of DRL- $(\frac{\partial u}{\partial y})$, OPP model, and SUB model. In controls based on information including the streamwise wall shear stress [DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ and DRL- $(\frac{\partial w}{\partial y})$], the pressure gradient suddenly drops at the beginning of the control and then shows a wild oscillation before eventually decreasing, whereas in other controls, the pressure gradient decreases relatively monotonically. This suggests that the mechanism of drag reduction may be different between the two groups of control.

An advantage of DRL-based control for drag reduction is that the optimum value of the magnitude of wall actuation is determined through the process of learning. As listed in Table III, the rms actuation for DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$, $0.124u_\tau$ for $Re_\tau = 180$ is smaller than that for the SUB model, $0.15u_\tau$, which was recommended in Ref. [7]. For a fair comparison, the rms actuation for the SUB model for

TABLE III. Averaged drag reduction % based on mean pressure gradient and the actuation intensity averaged over $t u_\tau / \delta = 10-30$ except for the SUB model, for which the intensity is fixed by the given value.

	Re_τ (trained)	Re_τ (tested)	DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$	DRL- $(\frac{\partial w}{\partial y})$	DRL- $(\frac{\partial u}{\partial y})$	OPP model	SUB model
Drag reduction (%)	180	180	23.1	22.5	20.4	21.3	18.9
	180	360	19.0	19.7	17.2	20.2	15.6
ϕ_{rms}/u_τ	180	180	0.124	0.153	0.115	0.149	0.150
	180	360	0.132	0.179	0.124	0.192	0.179

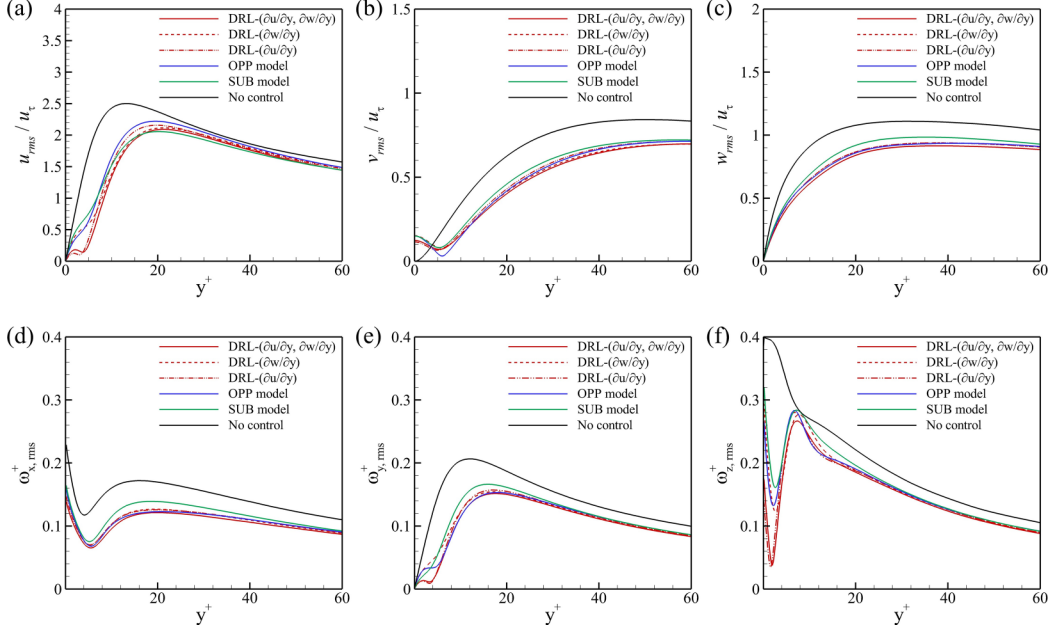


FIG. 6. rms velocity fluctuations normalized by the wall shear velocity of the unmanipulated flow at $Re_\tau = 180$. Parts (a), (b), and (c) illustrate the rms values in the streamwise, wall-normal, and spanwise directions, respectively. Parts (d), (e), and (f) provide the rms values of the fluctuation vorticity.

$Re_\tau = 360$ was set to the same value as that for the $DRL-(\frac{\partial w}{\partial y})$. Controls based on the streamwise wall shear stress appear to be more energy-efficient than the control based on the spanwise wall shear stress for both Re_τ . When $Re_\tau = 360$, the rms actuation for all DRL models increased slightly, although that for the OPP model exhibited the largest increase.

Figure 6 presents the distribution of the rms velocity and vorticity fluctuations of the manipulated channel for all models compared with those of the unmanipulated channel flow at $Re_\tau = 180$. All the components of the velocity and vorticity fluctuations were significantly suppressed by the wall actuation of all the control models. The near-wall distribution of the wall-normal component indicates that the wall actuation by all models clearly counteracts the wall-normal motion of turbulence, with the OPP model being the most pronounced. On the other hand, the distributions of the streamwise rms velocity, wall-normal rms vorticity, and spanwise rms vorticity show two distinct behaviors very near the wall between the controls based on the streamwise wall shear stress and others. Given that the spanwise rms velocity is suppressed to the same level in all models, the $DRL-(\frac{\partial u}{\partial y})$ and $DRL-(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ models suppress the streamwise velocity fluctuations in the region with $y^+ < 8$ more substantially than the other models, strongly suggesting that DRL models based on the streamwise shear stress weaken the streaky structures near the wall. This indicates that the drag reduction mechanism may be different between the two control groups. In the following section, we investigate these two drag reduction mechanisms in detail.

B. Identification of two drag reduction mechanisms

As demonstrated in the previous section, the DRL based on the wall shear stresses successfully determined the optimal wall actuation to reduce drag. However, it is not clear how the wall actuation proposed by the DRL models reduces drag. Furthermore, the amount of drag reduction achieved by the developed DRL models was comparable to that achieved by the OPP or SUB models. Therefore,

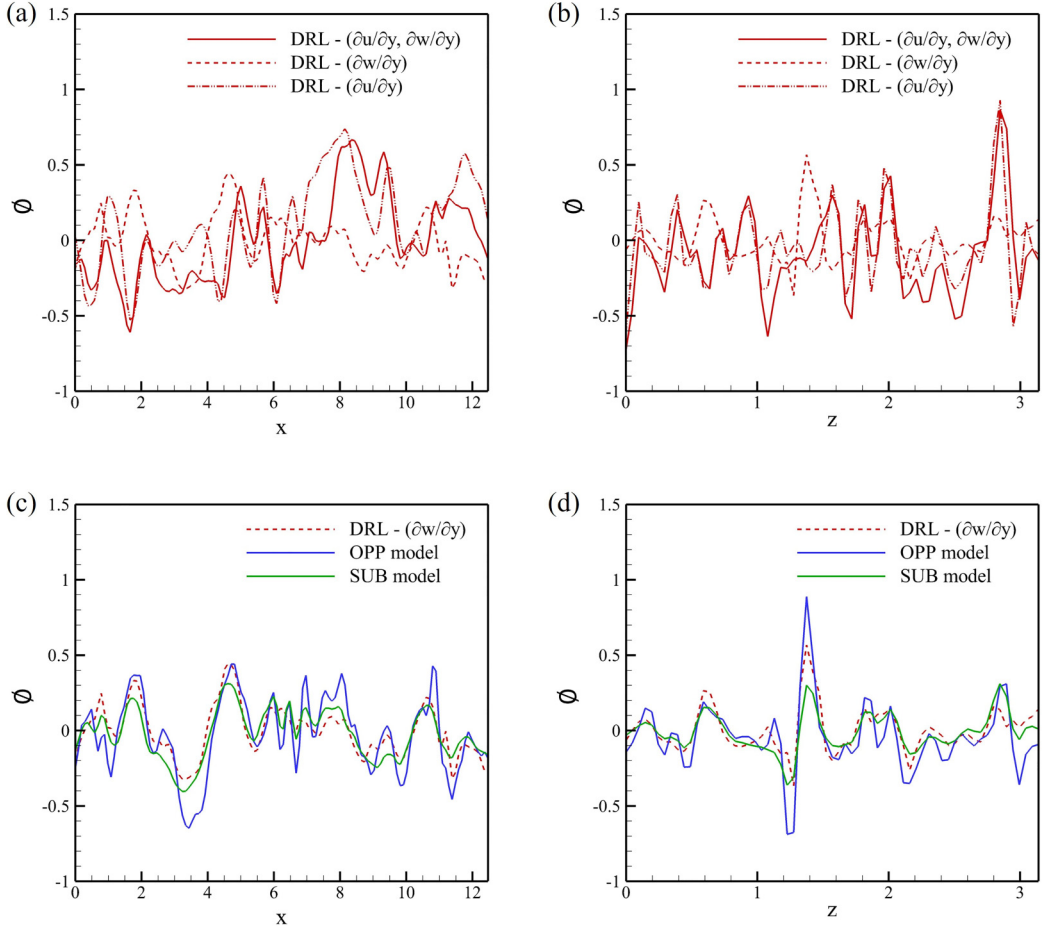


FIG. 7. Comparison of the wall actuation profile of the tested models for the same flow field (a), (c) along the streamwise direction and (b),(d) along the spanwise direction.

in this section we investigate the drag reduction mechanism of the DRL models compared with that of the OPP and SUB models.

First, we investigated the wall actuation distribution produced by a trained model under the same flow environment. For the flow field, one of the tested fields for DRL- $(\frac{\partial w}{\partial y})$ at $tu_\tau/\delta = 0.5$, where testing begins at $t = 0$. For example, Fig. 7 provides a comparison of the wall actuation distribution ϕ between the DRL models, the OPP model, and the SUB model. As shown in Figs. 7(a) and 7(b), the actuation distributions by DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ and DRL- $(\frac{\partial u}{\partial y})$ are similar, whereas that by DRL- $(\frac{\partial w}{\partial y})$ is markedly different. On the other hand, the actuation by DRL- $(\frac{\partial w}{\partial y})$ bears a strong resemblance to the OPP model or the SUB model, as illustrated in Figs. 7(c) and 7(d). These similarities suggest that the drag reduction mechanisms of DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ and DRL- $(\frac{\partial u}{\partial y})$ are similar to each other, while those of DRL- $(\frac{\partial w}{\partial y})$ and the OPP and SUB models are similar, despite these two mechanisms being different. Scatter plots of the actuations shown in Fig. 8 clearly confirm this: the actuations by either DRL- $(\frac{\partial u}{\partial y})$ or DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ and DRL- $(\frac{\partial w}{\partial y})$ are almost uncorrelated, whereas the actuations by either the OPP model or the SUB model and DRL- $(\frac{\partial w}{\partial y})$ are strongly correlated. Furthermore, the actuations of the SUB model are significantly more correlated with DRL- $(\frac{\partial w}{\partial y})$. This indicates

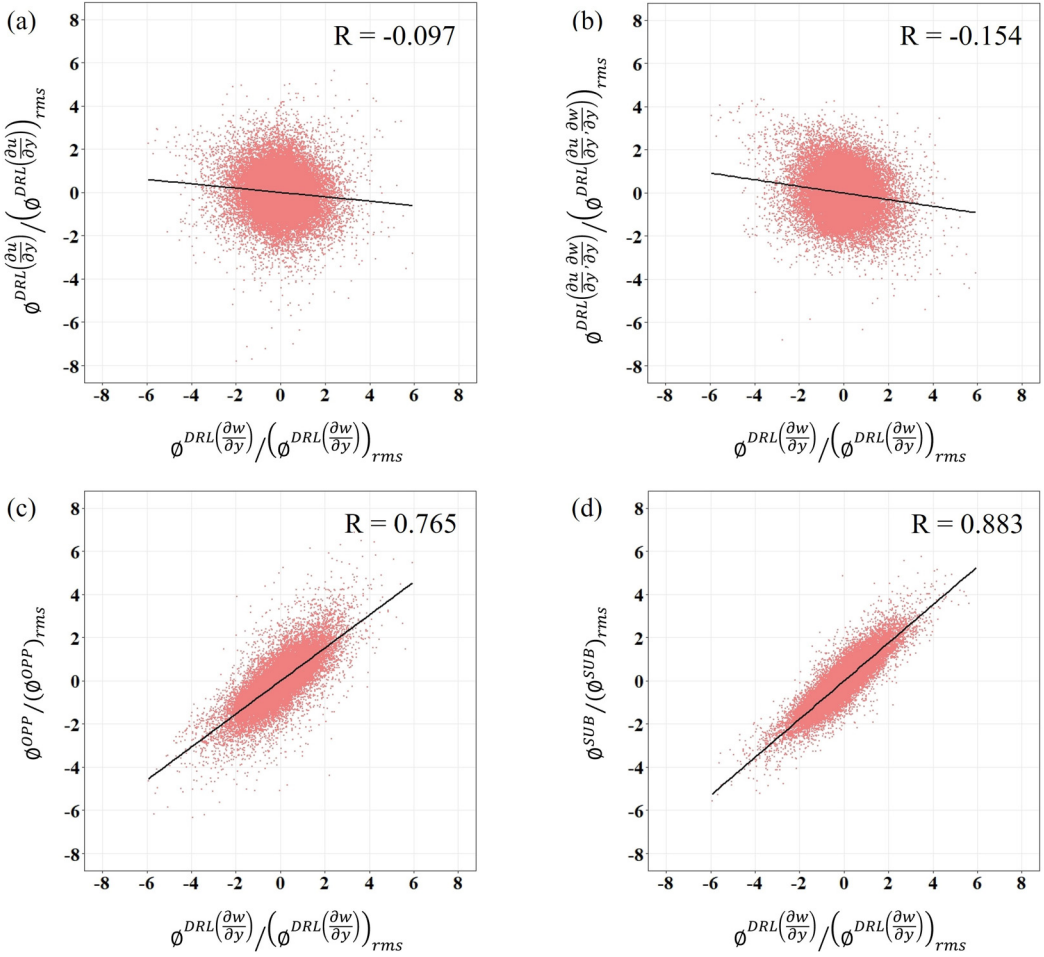


FIG. 8. Scatter plot between the actuations of (a) DRL-($\frac{\partial u}{\partial y}$), (b) DRL-($\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y}$), (c) OPP model, and (d) SUB model against the actuation of the DRL-($\frac{\partial w}{\partial y}$) model.

that the DRL-($\frac{\partial w}{\partial y}$) model reduces the drag by suppressing the up-and-down motion induced by the streamwise vortices because the behavior of the spanwise wall shear stress is a good indicator of the presence of the near-wall streamwise vortex. The lack of correlation between the actuations by the DRL models based on the information including the streamwise wall shear stress and the DRL based on the spanwise wall shear stress clearly suggests that the drag reduction mechanism of the former models is distinct from that of the latter model.

For a clearer identification of the drag reduction mechanism of the tested DRL models, we investigated the correlation between the actuation of each model and the wall shear stresses. The wall actuation distribution by DRL-($\frac{\partial u}{\partial y}$), DRL-($\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y}$), the SUB model, and DRL-($\frac{\partial w}{\partial y}$) for the same flow field shown in the line contour (solid for positive and dashed for negative values) is compared with the corresponding streamwise wall shear stress distribution in the color flood contours in Figs. 9(a)–9(d). The wall actuation by DRL-($\frac{\partial u}{\partial y}$) and DRL-($\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y}$) exhibits a strong correlation with the streamwise wall shear stress, whereas the wall actuation by the SUB model and DRL-($\frac{\partial w}{\partial y}$) is hardly correlated with the streamwise wall shear stress. This suggests that the DRL-($\frac{\partial u}{\partial y}$) and DRL-($\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y}$) models reduce drag through the suppression of streaky structures,

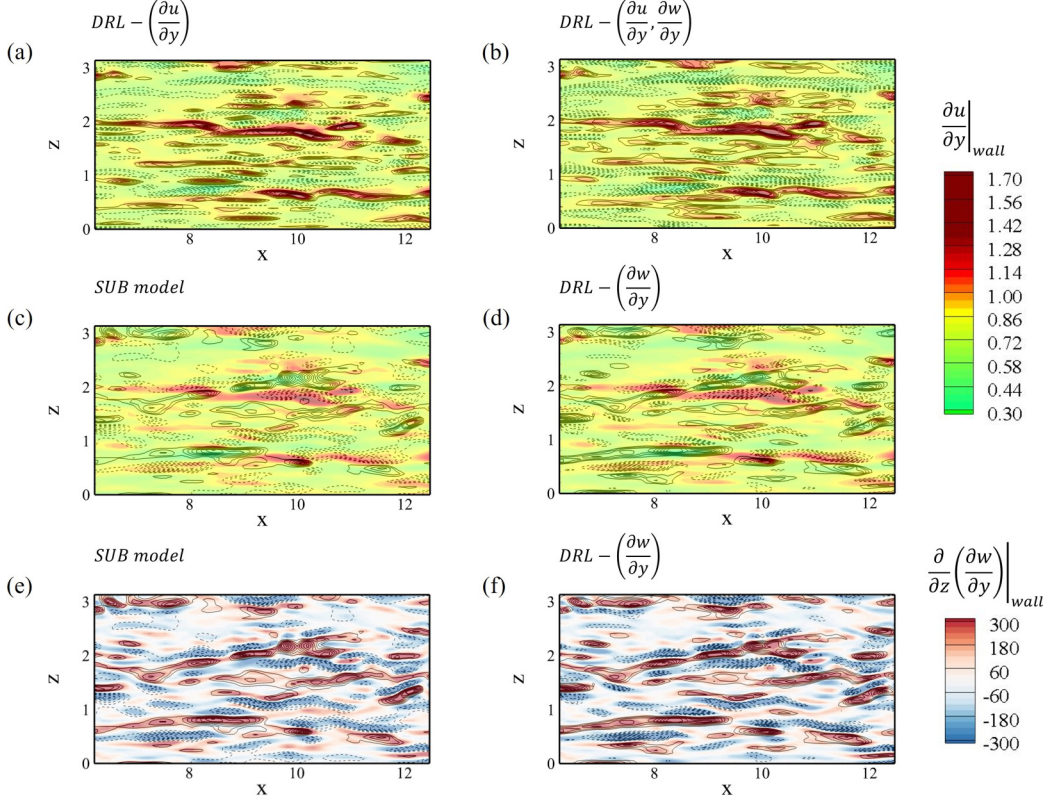


FIG. 9. Streamwise wall shear stress field (flood contours) and the actuation field (line contours) for (a) $DRL - \left(\frac{\partial u}{\partial y}\right)$, (b) $DRL - \left(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y}\right)$, (c) SUB model, and (d) $DRL - \left(\frac{\partial w}{\partial y}\right)$. Parts (a)–(d) show overlap of the streamwise wall shear field (flood contour) and each model’s action field (line contour). Spanwise gradient of the spanwise wall shear stress field (flood contours) and the actuation field (line contours) for (e) SUB model and (f) $DRL - \left(\frac{\partial w}{\partial y}\right)$. Among line contours, the dashed line denotes negative values.

both of which apply blowing to high-speed streaks and suction to low-speed streaks. This type of mechanism has never been found in any previous control scheme, such as opposition control or suboptimal control. However, the actuation by the SUB model and $DRL - \left(\frac{\partial w}{\partial y}\right)$ is strongly correlated with the spanwise derivative of the spanwise wall shear stress $\frac{\partial}{\partial z} \left(\frac{\partial w}{\partial y}\right)|_{\text{wall}}$ as demonstrated in Figs. 9(e) and 9(f). $\frac{\partial}{\partial z} \left(\frac{\partial w}{\partial y}\right)|_{\text{wall}}$, a good footprint indicating the presence of a near-wall streamwise vortex, confirms that both SUB and $DRL - \left(\frac{\partial w}{\partial y}\right)$ models suppress the up-and-down motion associated with the streamwise vortex since the sign of $\frac{\partial}{\partial z} \left(\frac{\partial w}{\partial y}\right)|_{\text{wall}}$ is negatively correlated with the wall-normal velocity near a streamwise vortex.

The scatter plots between the wall shear stress and actuation by the models shown in Fig. 10 quantitatively confirm the strong correlation between them. Noticeably, the correlation between actuation by $DRL - \left(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y}\right)$ and streamwise wall shear stress is stronger than that between $DRL - \left(\frac{\partial u}{\partial y}\right)$ and the streamwise wall shear stress. The extra information of $\frac{\partial w}{\partial y}$ in the input appears to be helpful in determining the optimal wall actuation for drag reduction, resulting in better drag reduction performance. Interestingly, the corresponding actuation is more correlated with the streamwise wall shear stress than actuation by $DRL - \left(\frac{\partial u}{\partial y}\right)$. According to quadrant analysis, high wall shear stress is generated by sweep events near the wall, whereas low wall shear stress is induced by ejection

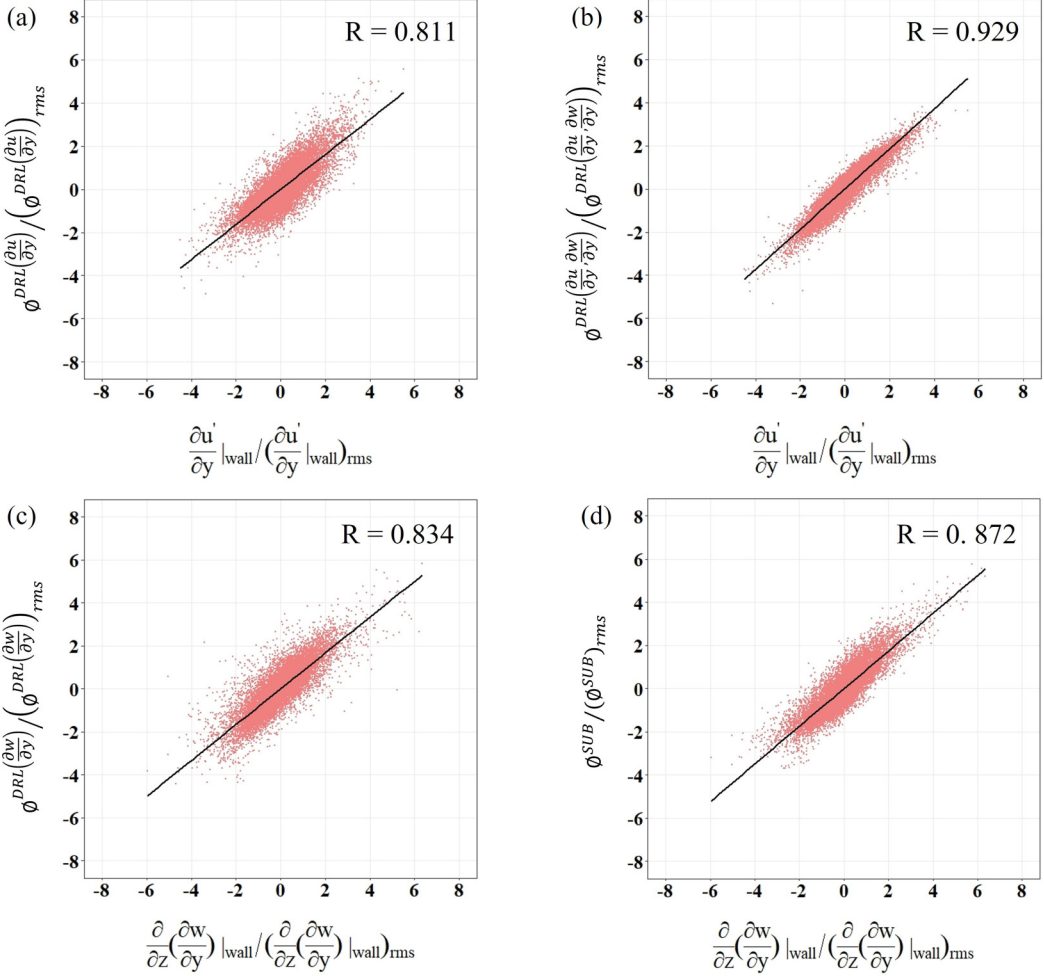


FIG. 10. Scatter plot between shear stress and actuation with correlation coefficient: (a), (b) scatter plots of the actuation of DRL- $(\frac{\partial u}{\partial y})$ and DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ with respect to the normalized streamwise shear stress, and (c), (d) scatter plots of the actuation of DRL- $(\frac{\partial w}{\partial y})$ and SUB model with respect to the normalized z -gradient of the spanwise shear stress.

events [44–46]. Here, we can see that the DRL- $(\frac{\partial u}{\partial y})$ and DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ models have nearly identical mechanisms for directly canceling out sweep and ejection events using the streamwise shear stress information. However, the actuation by DRL- $(\frac{\partial w}{\partial y})$ and the SUB model is equally strongly correlated with $\frac{\partial}{\partial z}(\frac{\partial w}{\partial y})|_{\text{wall}}$, as shown in Figs. 10(c) and 10(d), confirming that the actuation by the suboptimal control given by Eq. (1) is indeed close to the optimal actuation for drag reduction as long as only the spanwise wall shear stress is used as input. However, as shown in the drag reduction performance in Fig. 5 and Table III, DRL- $(\frac{\partial w}{\partial y})$ reduces more drag than the SUB model, suggesting that DRL found more optimal actuations than the SUB model.

To understand the role of actuations in the manipulation of near-wall turbulence for drag reduction, we investigated the near-wall flow field and the corresponding actuation by the DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ model in Fig. 11. Three example flow fields in the x - y plane clearly show that the optimal wall actuations counteract the near-wall wall-normal motion based on streamwise wall shear stress

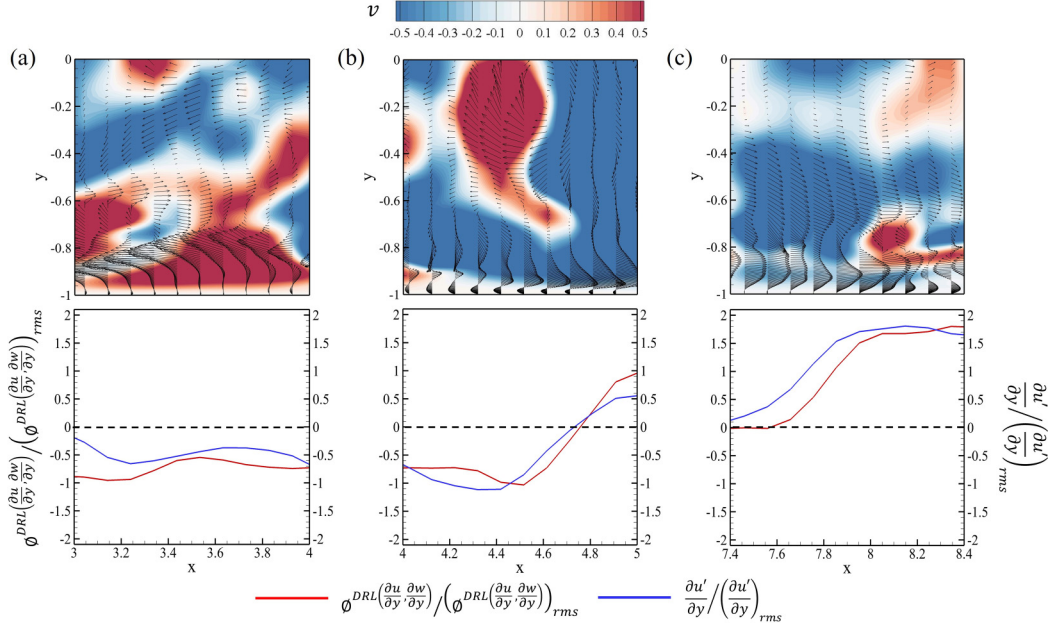


FIG. 11. Vertical velocity field (flood contour) with u - v velocity vector field and the wall actuation of the DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ model. In the 1D actuation profile, the red line indicates normalized actuation, and the blue line indicates normalized streamwise wall-fluctuation shear stress.

information. Where an ejection event occurs when $u' < 0$, $v > 0$ [most regions in Fig. 11(a) and the upstream portion in Fig. 11(b)], suction is applied to cancel the event, whereas a sweeping motion with $u' > 0$, $v < 0$ [the downstream portion in Fig. 11(b) and the biggest region in Fig. 11(c)] is suppressed by the wall blowing. It appears that $\frac{\partial u'}{\partial y}$ played a key role in identifying such near-wall events.

However, three near-wall flow fields in the y - z plane with actuation by the DRL- $(\frac{\partial w}{\partial y})$ model shown in Fig. 12 clearly show that the wall blowing and suction suppress the up-and-down motion caused by the streamwise vortices. Furthermore, the distribution of the spanwise gradient of the spanwise wall shear stress, $\frac{\partial}{\partial z}(\frac{\partial w}{\partial y})$, well captures the distribution of the optimal wall actuation by the DRL- $(\frac{\partial w}{\partial y})$ model, at least in the signwise sense.

It is noteworthy to compare the spatiotemporal distribution of the optimal wall actuations between ours and Sonoda *et al.* [32], although the sensing location for the state is different; the wall-shear stresses are measured in our control while the streamwise and wall-normal velocity components at $y^+ = 15$ are sensed in their control. As discussed above, the wall actuations obtained from our successful learning show streaky distribution elongated in the streamwise direction reflecting the local flow condition for the two distinct mechanisms found (see Fig. 9), whereas the wall actuations from their learning exhibit a spanwise uniform and streamwise wavy distribution with the wavelength of the whole domain (their Fig. 14). Different sensing location alone is not responsible for this difference in the actuation distribution. Different structures of the policy networks in learning such as the convolutional network adopted in our actor network and the pointwise nonlinear network in their actor network and different network parameters such as the frequency of sensing and actuation update might result in discovering different local optimum solutions.

Finally, the effect of actuation by DRL models on the near-wall turbulence is investigated through the two-point correlation between the wall shear stress and a proper quantity indicating near-wall events for each model. For the DRL- $(\frac{\partial w}{\partial y})$ model, the correlation between the spanwise wall shear

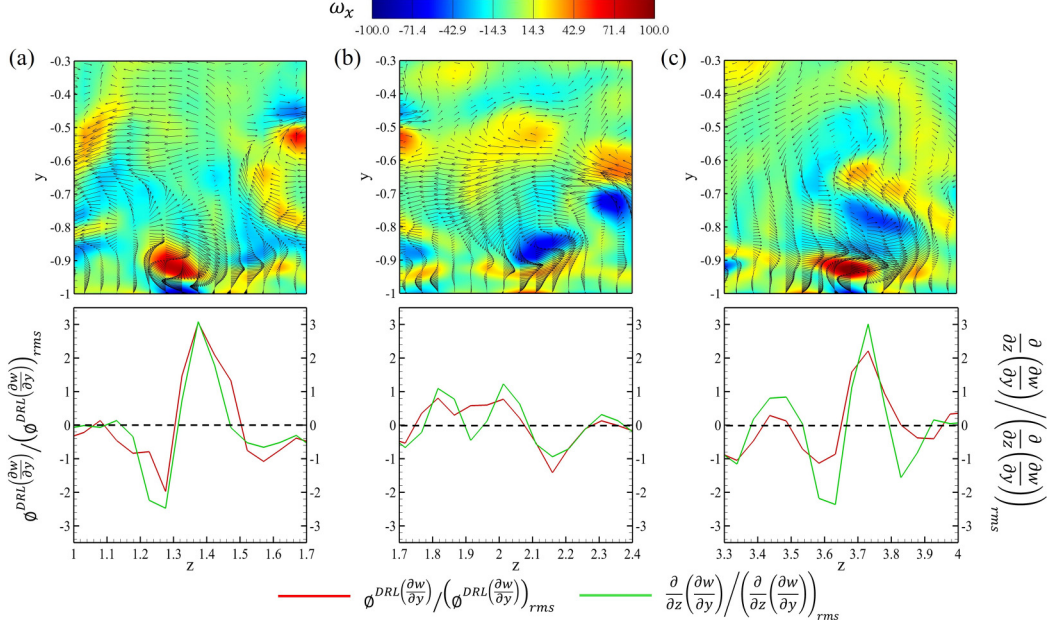


FIG. 12. Streamwise vorticity field (flood contour) with v - w velocity vector field and the wall actuation of the DRL- $(\frac{\partial w}{\partial y})$ model. In the 1D actuation profile, the red line indicates normalized actuation, and the green line indicates normalized z -gradient of the spanwise wall-swall shear stress.

stress and the streamwise vorticity component is investigated, whereas for the DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ model, the correlation between the streamwise wall shear stress and the streamwise fluctuation velocity is examined as defined by

$$R(\Delta y^+, \Delta z^+) = \frac{\langle \frac{\partial w}{\partial y}(x, 0, z) \omega_x(x, \Delta y^+, z + \Delta z^+) \rangle_{x,z}}{\frac{\partial w}{\partial y} |_{\text{wall,rms}} \omega_x |_{\Delta y^+, \text{rms}}}, \quad (11)$$

$$R(\Delta x^+, \Delta y^+) = \frac{\langle \frac{\partial u'}{\partial y}(x, 0, z) u'(x + \Delta x^+, \Delta y^+, z) \rangle_{x,z}}{\frac{\partial u'}{\partial y} |_{\text{wall,rms}} u' |_{\Delta y^+, \text{rms}}}, \quad (12)$$

where Δx^+ , Δy^+ , and Δz^+ indicate the spatial separation in the streamwise, vertical, and spanwise directions, respectively. $\langle \rangle_{x,z}$ denotes the average over the x - z space. Figure 13 illustrates the contour plots of the two-point correlation for the unmanipulated flow and the controlled flows by DRL- $(\frac{\partial w}{\partial y})$ and DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ models. Compared to the unmanipulated flow, these correlations are suppressed by the respective DRL models. This confirms that the streamwise vortices are weakened by the DRL- $(\frac{\partial w}{\partial y})$ model, and the sweep and ejection events are suppressed by the DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ model, that is, the effect of the wall actuation successfully propagates to the region far from the wall through accumulated control based on DRL.

IV. CONCLUSION

A deep reinforcement learning method based on the deep policy gradient model was applied to turbulence control for drag reduction. In the direct numerical simulation of channel turbulence, the optimal distribution of wall blowing and suction was determined based on the wall shear stress information to minimize the pressure gradient for a given flow rate. The tested DRL-based

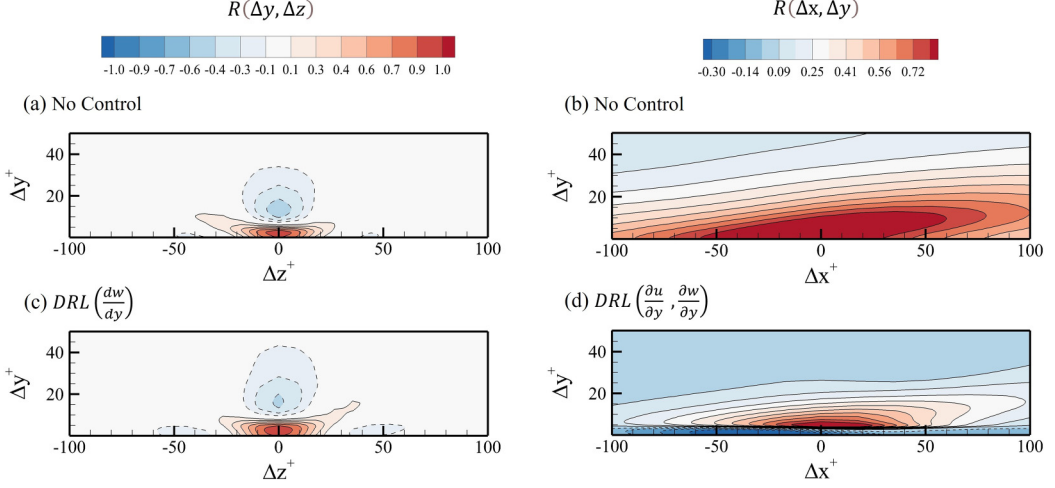


FIG. 13. Two-point correlation contour between $\frac{\partial w}{\partial y}|_w(0,0)$ and $\omega_x(\Delta y, \Delta z)$ in the y - z plane for (a) unmanipulated field and (b) DRL- $(\frac{\partial w}{\partial y})$ -controlled field, respectively. Two-point correlation between $\frac{\partial u'}{\partial y}|_w(0,0)$ and $u(\Delta x, \Delta y)$ in the x - y plane for (c) unmanipulated field and (d) DRL- $(\frac{\partial u}{\partial y}, \frac{\partial w}{\partial y})$ -controlled field.

models reduced the drag substantially, and the amount of drag reduction was approximately 20% at $Re_\tau = 180$ and 360, which is comparable with the opposition control using the wall-normal velocity sensed inside the flow domain and the suboptimal control based on the spanwise wall shear stress.

From the investigation of the optimal wall actuation distribution that each tested model would produce for the same flow environment, we found that the DRL models based on the information including the streamwise wall shear stress respond to the environment differently from the DRL model based on the spanwise wall shear stress only. There are indeed two distinct drag reduction mechanisms that depend on the input information. The DRL model based on the spanwise wall shear stress only reduces drag through the streamwise vortex cancellation mechanism, which is the drag reduction mechanism of the opposition control and suboptimal control. A new type of drag reduction mechanism was found in DRL models using the streamwise wall shear stress as input; the sweep and ejection motion was suppressed by the optimal wall actuation. In doing so, the spanwise alternating streaky structures frequently found in near-wall turbulence weaken. Interestingly, when both the streamwise and spanwise wall shear stresses were used as inputs, the drag reduction mechanism of the sweep and ejection cancellation seems to be selected, although the model using both information reduces drag slightly more than the model using the streamwise wall shear stress only. In this case, the streamwise wall shear stress information was learned prior to the spanwise wall shear stress information, and thus a mechanism that counteracts the streamwise vorticity could not be activated. This implies that the two types of reduction mechanism cannot be learned simultaneously. This also suggests a possibility that a different setup of DRL or a different duration of learning might result in the discovery of a different mechanism.

A comparison between the optimal wall actuation of the DRL models using both stresses and the streamwise wall shear stress only shows that the wall actuation distribution is highly correlated with the local streamwise wall shear stress. This suggests that there may exist an explicit control law, such as $\phi \sim \frac{\partial u'}{\partial y}$, similar to Eq. (1) of the suboptimal control. We tested this scheme, but its performance was poor. This was attributed to the extremely strong sensitivity between the local actuation and sensed input at the same point.

On the other hand, from our observation that the drag reduction mechanism of the DRL model based on the spanwise wall shear stress is almost identical to that of the suboptimal control, we can

conclude that the suboptimal model is close to the globally optimal solution for drag reduction as long as only the spanwise wall shear stress is used as the input. However, the DRL model based on the spanwise wall shear stress performs better than the suboptimal control, particularly when tested at Reynolds numbers higher than that of the learned flow.

The DRL optimization in the drag reduction problem has some advantages over the traditional adjoint-based approaches such as an optimal control theory [47] or the suboptimal approach [7]. The optimal control algorithm requires full flow field information and involves iterative direct numerical simulations, restricting a practical application, whereas the DRL optimization is easily applicable to practical situations such as an experimental setting with minimal information such as the wall-measurable quantities and drag, and the computational overhead due to the DRL per time step is moderate (11% of total time). Compared with the suboptimal control approach, which yielded an explicit control scheme with the undetermined actuation amplitude, the DRL-based control possesses the adaptive scaling capability, sensitively affecting the performance of control. For instance, although the DRL- $(\frac{\partial w}{\partial y})$ model and the SUB model with the fixed amplitude of the actuation are similar in the drag reduction mechanism, the DRL model performs better than the SUB model due to this self-controlled action mechanism.

The reinforcement learning algorithm used in this study is particularly adept at locating the optimal solution in high dimensions, particularly in a continuous physical space or time. This algorithm performed well in both simulations and experiments. However, along with the advantage of self-learning, reinforcement learning has the disadvantage of requiring the selection of numerous hyperparameters and rewards. Performance is often highly sensitive to hyperparameter values, rewards, and other human-determined heuristics; thus, the learning results can vary significantly. Therefore, it is still questionable whether the hyperparameters discovered in this study are optimal or whether the actuation obtained is indeed optimal. However, we demonstrated that deep reinforcement learning can be successfully applied to the drag reduction problem, and an interpretation of the optimal action can lead to the discovery of a new type of physical mechanism behind successful control.

ACKNOWLEDGMENTS

This work was supported by a National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIP) (2022R1A2C2005538).

APPENDIX: DERIVATION OF EQ. (1) IN SUBOPTIMAL CONTROL

In this Appendix, we provide the derivation of the explicit control scheme based on the spanwise wall-shear stress in the suboptimal control given by Eq. (1) for easy accessibility, although it has been introduced in Ref. [7]. From the observation that the spanwise wall-shear stress increases over one time step by the wall blowing and suction given by the opposition control (Fig. 3 of Ref. [7]), the cost functional to be minimized is set to be

$$\mathcal{J} = -\frac{1}{2A\Delta t} \int_S \int_t^{t+\Delta t} \left. \frac{\partial w}{\partial y} \right|_w dt dS + \frac{\ell}{2A\Delta t} \int_S \int_t^{t+\Delta t} \phi^2 dt dS, \quad (\text{A1})$$

where dS is an area element over the wall, and ℓ is the relative cost of the actuation ϕ . We define the differential states of the velocity and pressure (θ_i, ρ) using a Fréchet differential,

$$\theta_i = \frac{Du_i(\phi)}{\mathcal{D}\phi} \tilde{\phi}, \quad (\text{A2})$$

$$\rho = \frac{Dp(\phi)}{\mathcal{D}\phi} \tilde{\phi}, \quad (\text{A3})$$

where

$$\frac{\mathcal{D}f(\phi)}{\mathcal{D}\phi} \tilde{\phi} = \lim_{\epsilon \rightarrow 0} \frac{f(\phi + \epsilon \tilde{\phi}) - f(\phi)}{\epsilon}, \quad (\text{A4})$$

with $\tilde{\phi}$ being an arbitrary perturbation field to ϕ . In the discretization of Eqs. (2) and (3), the choice of the Crank-Nicolson scheme for the viscous and pressure terms and any explicit scheme for the nonlinear terms yields

$$u_i^{n+1} - \frac{\Delta t}{2 \text{Re}_\tau} \frac{\partial^2 u_i^{n+1}}{\partial x_j \partial x_j} + \frac{\Delta t}{2} \frac{\partial p^{n+1}}{\partial x_i} = R_i^n, \quad (\text{A5})$$

$$\frac{\partial u_j^{n+1}}{\partial x_j} = 0, \quad (\text{A6})$$

with

$$u_2^{n+1}|_w = \phi, \quad (\text{A7})$$

where the superscripts n and $n+1$ indicate two consecutive time steps, and R_i^n includes all the explicit terms depending on the n th time step information. Taking the Fréchet differential of Eqs. (A5) and (A6) results in

$$\theta_i^{n+1} - \frac{\Delta t}{2 \text{Re}_\tau} \frac{\partial^2 \theta_i^{n+1}}{\partial x_j \partial x_j} + \frac{\Delta t}{2} \frac{\partial \rho^{n+1}}{\partial x_i} = 0, \quad (\text{A8})$$

$$\frac{\partial \theta_j^{n+1}}{\partial x_j} = 0, \quad (\text{A9})$$

with

$$\theta_2^{n+1}|_w = \tilde{\phi}. \quad (\text{A10})$$

Hereafter, the superscript $n+1$ is dropped. Since Eqs. (A8) and (A9) are linear, the Fourier coefficients $\hat{\theta}_i$ and $\hat{\rho}$ of θ_i and ρ defined by

$$\theta_i(x, y, z) = \sum_{\kappa_x} \sum_{\kappa_z} \hat{\theta}_i(y) e^{i(\kappa_x x + \kappa_z z)}, \quad \rho(x, y, z) = \sum_{\kappa_x} \sum_{\kappa_z} \hat{\rho}(y) e^{i(\kappa_x x + \kappa_z z)}, \quad (\text{A11})$$

are found in the explicit form under the approximation that $2 \text{Re}_\tau / \Delta t \gg \kappa^2$:

$$\hat{\theta}_1(y) = \frac{i\kappa_x}{\kappa} \hat{\phi} (\exp[-(2 \text{Re}_\tau / \Delta t)^{1/2} y] - \exp(-\kappa y)), \quad (\text{A12})$$

$$\hat{\theta}_2(y) = \hat{\phi} \exp(-\kappa y), \quad (\text{A13})$$

$$\hat{\theta}_3(y) = \frac{i\kappa_z}{\kappa} \hat{\phi} (\exp[-(2 \text{Re}_\tau / \Delta t)^{1/2} y] - \exp(-\kappa y)), \quad (\text{A14})$$

$$\hat{\rho}(y) = \frac{2}{\kappa \Delta t} \hat{\phi} \exp(-\kappa y), \quad (\text{A15})$$

where $\hat{\phi}$ is the Fourier coefficient of $\tilde{\phi}$.

Taking the Fréchet differential of the cost functional [Eq. (A1)] yields

$$\frac{\mathcal{D}J}{\mathcal{D}\phi} \tilde{\phi} = -\frac{1}{A \Delta t} \int_S \int_t^{t+\Delta t} \left. \frac{\partial w}{\partial y} \right|_w \left. \frac{\partial \theta_3}{\partial y} \right|_w dt dS + \frac{\ell}{A \Delta t} \int_S \int_t^{t+\Delta t} \phi \tilde{\phi} dt dS, \quad (\text{A16})$$

the Fourier representation of which is

$$\frac{\widehat{D}J}{D\phi} \hat{\phi}^* = - \left. \frac{\partial w}{\partial y} \right|_w \frac{\partial \widehat{\theta}_3^*}{\partial y} \Big|_w + \ell \hat{\phi} \hat{\phi}^*, \quad (\text{A17})$$

where the hat denotes the Fourier coefficient, and the superscript * implies the complex conjugate. From Eq. (A14),

$$\left. \frac{\partial \widehat{\theta}_3}{\partial y} \right|_w = \frac{i\kappa_z}{\kappa} \hat{\phi} \left[- \left(\frac{2 \text{Re}\tau}{\Delta t} \right)^{1/2} + \kappa \right] \simeq - \frac{i\kappa_z}{\kappa} \left(\frac{2 \text{Re}\tau}{\Delta t} \right)^{1/2} \hat{\phi}. \quad (\text{A18})$$

Therefore, Eq. (A17) reduces to

$$\frac{\widehat{D}J}{D\phi} \hat{\phi}^* = - \frac{i\kappa_z}{\kappa} \left(\frac{2 \text{Re}\tau}{\Delta t} \right)^{1/2} \left. \frac{\partial w}{\partial y} \right|_w \hat{\phi}^* + \ell \hat{\phi} \hat{\phi}^*, \quad (\text{A19})$$

which should be satisfied for an arbitrary $\hat{\phi}$, yielding

$$\frac{\widehat{D}J}{D\phi} = - \frac{i\kappa_z}{\kappa} \left(\frac{2 \text{Re}\tau}{\Delta t} \right)^{1/2} \left. \frac{\partial w}{\partial y} \right|_w + \ell \hat{\phi}. \quad (\text{A20})$$

The condition in which the Fréchet differential of the cost functional is minimized yields

$$\hat{\phi} = C \frac{i\kappa_z}{\kappa} \left. \frac{\partial w}{\partial y} \right|_w. \quad (\text{A21})$$

-
- [1] J. Kim, Physics and control of wall turbulence for drag reduction, *Philos. Trans. R. Soc. A* **369**, 1396 (2011).
- [2] S. L. Brunton and B. R. Noack, Closed-loop turbulence control: Progress and challenges, *Appl. Mech. Rev.* **67**, 050801 (2015).
- [3] J. Rabault, M. Kuchta, A. Jensen, U. Réglade, and N. Cerardi, Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control, *J. Fluid Mech.* **865**, 281 (2019).
- [4] H. Choi, P. Moin, and J. Kim, Active turbulence control for drag reduction in wall-bounded flows, *J. Fluid Mech.* **262**, 75 (1994).
- [5] C. M. Ho and Y. C. Tai, Review: MEMS and its application to flow control, *J. Fluids Eng.* **118**, 437 (1996).
- [6] C. Lee, J. Kim, D. Babcock, and R. Goodman, Application of neural networks to turbulence control for drag reduction, *Phys. Fluids* **9**, 1740 (1997).
- [7] C. Lee, J. Kim, and H. Choi, Suboptimal control of turbulent channel flow for drag reduction, *J. Fluid Mech.* **358**, 245 (1998).
- [8] L. Löfdahl and M. Gad-el-Hag, MEMS-based pressure and shear stress sensors for turbulent flows, *Meas. Sci. Technol.* **10**, 665 (1999).
- [9] J.-I. Choi and H. J. Sung, Assessment of suboptimal control for drag reduction in turbulent channel flow, *J. Turbul.* **3**, N29 (2002).
- [10] H. Rebbeck and K. S. Choi, A wind-tunnel experiment on real-time opposition control of turbulence, *Phys. Fluids* **18**, 035103 (2006).
- [11] N. Kasagi, Y. Suzuki, and K. Fukagata, Microelectromechanical systems-based feedback control of turbulence for skin friction reduction, *Annu. Rev. Fluid Mech.* **41**, 231 (2009).
- [12] J. Kim and C. Lee, Deep unsupervised learning of turbulence for inflow generation at various Reynolds numbers, *J. Comput. Phys.* **406**, 109216 (2020).

- [13] J. Kim and C. Lee, Prediction of turbulent heat transfer using convolutional neural networks, *J. Fluid Mech.* **882**, A18 (2020).
- [14] H. Kim, J. Kim, S. Won, and C. Lee, Unsupervised deep learning for super-resolution reconstruction of turbulence, *J. Fluid Mech.* **910**, A29 (2021).
- [15] J. N. Kutz, Deep learning in fluid dynamics, *J. Fluid Mech.* **814**, 1 (2017).
- [16] M. P. Brenner, J. D. Eldredge, and J. B. Freund, Perspective on machine learning for advancing fluid mechanics, *Phys. Rev. Fluids* **4**, 100501 (2019).
- [17] K. Duraisamy, G. Iaccarino, and H. Xiao, Turbulence modeling in the age of data, *Annu. Rev. Fluid Mech.* **51**, 357 (2019).
- [18] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, Machine learning for fluid mechanics, *Annu. Rev. Fluid Mech.* **52**, 477 (2020).
- [19] B.-Z. Han and W.-X. Huang, Active control for drag reduction of turbulent channel flow based on convolutional neural networks, *Phys. Fluids* **32**, 095108 (2020).
- [20] J. Park and H. Choi, Machine-learning-based feedback control for drag reduction in a turbulent channel flow, *J. Fluid Mech.* **904**, A24 (2020).
- [21] L. P. Kaelbling, M. L. Littman, and A. W. Moore, Reinforcement learning: A survey, *J. Artif. Intell. Res.* **4**, 237 (1996).
- [22] N. S. Nise, *Control Systems Engineering* (Wiley, Hoboken, NJ, 2020).
- [23] H. Tang, J. Rabault, A. Kuhnle, Y. Wang, and T. Wang, Robust active flow control over a range of Reynolds numbers using an artificial neural network trained through deep reinforcement learning, *Phys. Fluids* **32**, 053605 (2020).
- [24] H. Ghraieb, J. Viquerat, A. Larcher, P. Meliga, and E. Hachem, Single-step deep reinforcement learning for open-loop control of laminar and turbulent flows, *Phys. Rev. Fluids* **6**, 053902 (2021).
- [25] K. Gustavsson, L. Biferale, A. Celani, and S. Colabrese, Finding efficient swimming strategies in a three-dimensional chaotic flow by reinforcement learning, *Eur. Phys. J. E* **40**, 110 (2017).
- [26] J. Viquerat, J. Rabault, A. Kuhnle, H. Ghraieb, A. Larcher, and E. Hachem, Direct shape optimization through deep reinforcement learning, *J. Comput. Phys.* **428**, 110080 (2021).
- [27] G. Beintema, A. Corbetta, L. Biferale, and F. Toschi, Controlling Rayleigh–Bénard convection via reinforcement learning, *J. Turbul.* **21**, 585 (2020).
- [28] P. Garnier, J. Viquerat, J. Rabault, A. Larcher, A. Kuhnle, and E. Hachem, A review on deep reinforcement learning for fluid mechanics, *Comput. Fluids* **225**, 104973 (2021).
- [29] G. Novati, H. L. de Laroussilhe, and P. Koumoutsakos, Automating turbulence modelling by multi-agent reinforcement learning, *Nat. Mach. Intell.* **3**, 87 (2021).
- [30] J. Kim, H. Kim, J. Kim, and C. Lee, Deep reinforcement learning for large-eddy simulation modeling in wall-bounded turbulence, *Phys. Fluids* **34**, 105132 (2022).
- [31] D. Fan, L. Yang, Z. Wang, M. S. Triantafyllou, and G. E. Karniadakis, Reinforcement learning for bluff body active flow control in experiments and simulations, *Proc. Natl. Acad. Sci. USA* **117**, 26091 (2020).
- [32] T. Sonoda, Z. Liu, T. Itoh, and Y. Hasegawa, Reinforcement learning of control strategies for reducing skin friction drag in a fully developed channel flow, [arXiv:2206.15355](https://arxiv.org/abs/2206.15355).
- [33] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, Cambridge, MA, 2018).
- [34] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, Continuous control with deep reinforcement learning, [arXiv:1509.02971](https://arxiv.org/abs/1509.02971).
- [35] S. Fujimoto, H. Hoof, and D. Meger, Addressing function approximation error in actor-critic methods, in *Proceedings of the 35th International Conference on Machine Learning* (PMLR, 80, 2018), pp. 1587–1596.
- [36] J. Rabault and A. Kuhnle, Accelerating deep reinforcement learning strategies of flow control through a multi-environment approach, *Phys. Fluids* **31**, 094105 (2019).
- [37] <https://github.com/taehyuklee/TurbulenceControlCode>.

- [38] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, Deterministic policy gradient algorithms, in *Proceedings of the 31st International Conference on Machine Learning* (PMLR, 32(1), 2014), pp. 387–395.
- [39] M. T. Wojnowicz, The Ornstein-Uhlenbeck process in neural decision-making: Mathematical foundations and simulations suggesting the adaptiveness of robustly integrating stochastic neural evidence, Ph.D. thesis, University of Washington, 2013.
- [40] S. B. Pope, *Turbulent Flows* (Cambridge University Press, Cambridge, UK, 2000).
- [41] M. Plappert, R. Houthoofd, P. Dhariwal, S. Sidor, R. Y. Chen, X. Chen, T. Asfour, P. Abbeel, and M. Andrychowicz, Parameter space noise for exploration, [arXiv:1706.01905](https://arxiv.org/abs/1706.01905).
- [42] V. Belus, J. Rabault, J. Viquerat, Z. Che, E. Hachem, and U. Reglade, Exploiting locality and translational invariance to design effective deep reinforcement learning control of the 1-dimensional unstable falling liquid film, *AIP Adv.* **9**, 125014 (2019).
- [43] A. Cahill, Catastrophic forgetting in reinforcement-learning environments, Ph.D. thesis, University of Otago, 2011.
- [44] J. Kim and P. Moin, The structure of the vorticity field in turbulent channel flow. part 2. study of ensemble-averaged fields, *J. Fluid Mech.* **162**, 339 (1986).
- [45] J. M. Wallace, H. Eckelmann, and R. S. Brodkey, The wall region in turbulent shear flow, *J. Fluid Mech.* **54**, 39 (1972).
- [46] R. J. Adrian, Hairpin vortex organization in wall turbulence, *Phys. Fluids* **19**, 041301 (2007).
- [47] T. R. Bewley, P. Moin, and R. Temam, DNS-based predictive control of turbulence: An optimal benchmark for feedback algorithms, *J. Fluid Mech.* **447**, 179 (2001).