DOI: 10.1111/mice.12655

ORIGINAL ARTICLE

COMPUTER-AIDED CIVIL AND INFRASTRUCTURE ENGINEERING WILEY

A knowledge-enhanced deep reinforcement learning-based shape optimizer for aerodynamic mitigation of wind-sensitive structures

Shaopeng Li[1] | Reda Snaiki[1,2] | Teng Wu[1]

[1] Department of Civil, Structural and Environmental Engineering, University at Buffalo State University of New York, Buffalo, New York, USA

[2] Department of Applied Sciences, Université du Québec à Chicoutimi, Chicoutimi, Québec, Canada

Correspondence
Teng Wu, Department of Civil, Structural and Environmental Engineering, University at Buffalo, Buffalo, NY 14260, USA.
Email: tengwu@buffalo.edu

Abstract

Structural shape optimization plays an important role in the design of wind-sensitive structures. The numerical evaluation of aerodynamic performance for each shape search and update during the optimization process typically involves significant computational costs. Accordingly, an effective shape optimization algorithm is needed. In this study, the reinforcement learning (RL) method with deep neural network (DNN)-based policy is utilized for the first time as a shape optimization scheme for aerodynamic mitigation of wind-sensitive structures. In addition, "tacit" domain knowledge is leveraged to enhance the training efficiency. Both the specific direct-domain knowledge and general cross-domain knowledge are incorporated into the deep RL-based aerodynamic shape optimizer via the transfer-learning and meta-learning techniques, respectively, to reduce the required datasets for learning an effective RL policy. Numerical examples for aerodynamic shape optimization of a tall building are used to demonstrate that the proposed knowledge-enhanced deep RL-based shape optimizer outperforms both gradient-based and gradient-free optimization algorithms.

1 | INTRODUCTION

The rapid increase in height of buildings and span of bridges makes these slender structures extremely sensitive to winds. In addition to optimizing structural properties (e.g., Kociecki & Adeli, 2014; Park & Adeli, 1997) and utilizing structural control techniques (e.g., Kim & Adeli, 2005; Wang & Adeli, 2015), various aerodynamic mitigation strategies by modifying external shapes are employed in the design process. The selection of an appropriate aerodynamic shape is traditionally based on several candidates resulting from a designer's engineering experience and judgment. Usually the iterative procedure to update these baseline geometries is not triggered unless a safety or serviceability issue of the structure under aerodynamic loads is identified. In case an aerodynamic improvement is required, a limited number of aerodynamic mitigation options are available, for example, corner modification or helical twisting for high-rise buildings (Davenport, 1971; Tanaka, Tamura, Ohtake, Nakai, & Kim, 2012) and edge fairing or central slot adding for long-span bridges (Nagao, Utsunomiya, Oryu, & Manabe, 1993; Yang, Wu, Ge, & Kareem, 2015). Although this cut-and-try design, essentially based on intuition, is routinely used by the engineering community as a viable problem-solving approach, a mathematically optimal (or near optimal) aerodynamic configuration and hence a cost-effective shape design is not necessarily acquired. However, the rapid increase of structural height/span (with innovative cross sections) and recent advances of performance-based wind engineering

methodology (with various engineering objectives) have placed a demand for more cost-effective aerodynamic designs. To this end, there is a need for an automated process to facilitate the comprehensive search of shape design space that is rigorously guided by optimization algorithms and the efficient evaluation of aerodynamic performance with each updated structural geometry (Ding & Kareem, 2018). To achieve this goal, the mathematical programming techniques are applied to the aerodynamic shape optimization process (Topping, 1983). The problem formulation of the aerodynamic shape optimization generally consists of assessment of aerodynamic performance, parameterization of external shape, and specification of a set of geometric requirements, respectively, represented by objective functions, design variables, and design constrains (Skinner & Zare-Behtash, 2018).

A wind tunnel experiment is considered as one of the most reliable ways to assess structural aerodynamic performance. However, a systematic testing procedure involving automated fabrication of structural models, acquisition, and processing of input–output data, and control of fan operations is not currently available. Usually, the computational fluid dynamics (CFD) simulation, along with its mesh update schemes, is utilized in each search step during the shape optimization process for aerodynamic mitigation of wind-sensitive structures (Elshaer & Bitsuamlak, 2018). Due to the extreme complexities involved in the bluff-body aerodynamics and wind–structure interactions at large Reynolds numbers, high computational cost is needed for a reliable CFD simulation. Although surrogate models can be used to effectively alleviate the computational burden, further investigation may be needed to enhance their performance in terms of interpolation/extrapolation accuracy in the simulation (e.g., using adaptive surrogate models) and this consideration is outside the scope of the current study (Peherstorfer, Willcox, & Gunzburger, 2018; Yazdi & Neyshabouri, 2014). On the other hand, there have been significant efforts on the development of effective optimization algorithms that can achieve the globally optimal solution with a relatively small number of iterations (Skinner & Zare-Behtash, 2018).

Among numerous mathematical formulations of various aerodynamic shape search and update rules, the gradient-based optimization algorithms (e.g., basic gradient decent, gradient decent with momentum and gradient descent with adaptive step size) are widely employed since they are easy to implement and sample efficient. However, the gradient-based algorithms often get trapped in local optima that heavily depend on the start points. The accumulated engineering experience and intuition may be helpful in appropriate selection of initial configurations (baseline designs); however, they provide little contribution to intelligently guiding the search of globally optimal solution since effective communication between the human-readable knowledge through cognition process (e.g., thought, experience, and sense) and machine-readable information for computational algorithms has not been well established yet. To increase the chance of acquiring the global optima from the whole search space, a number of gradient-free optimization algorithms (e.g., genetic algorithm, particle swarm optimization [PSO], and simulated annealing) have been developed at the expense of sample efficiency. It is noted that both gradient-based and gradient-free optimization algorithms are essentially hand-design approaches, where the determination of their parameters for a specific application is usually based on a costly, manual trial-and-error process (Andrychowicz et al., 2016). To further enhance the automation level based on mathematical programming techniques and hence save computational cost, an auto-learned optimization approach based on increasingly popular deep learning techniques would probably be a better choice. With newer and more powerful learning algorithms, deep learning has been utilized in many engineering fields (e.g., Benito-Picazo, Domínguez, Palomo, & López-Rubio, 2020; Simões, Lau, & Reis, 2020; Sørensen, Nielsen, & Karstoft, 2020). Despite the existing applications in civil engineering (e.g., Liang, 2019; Rafiei & Adeli, 2017a; Rafiei, Khushefati, Demirboga, & Adeli, 2017), its great potential to improve the optimization scheme for aerodynamic mitigation has not been well explored yet. To this end, the reinforcement learning (RL) methodology will be utilized here for the first time as a data-driven shape optimization scheme for aerodynamic mitigation of wind-sensitive structures. In RL setting, the effective policy (i.e., shape search and update rule) with a goal to efficiently achieve the globally optimal solution (i.e., maximizing aerodynamic mitigation) can be learnt by an agent (i.e., structure) through interacting with its environment (i.e., wind) based on an automated trial-and-error process (Sutton & Barto, 2018). In addition, the RL policy will be represented by a deep neural network (DNN). The obtained deep RL-based optimizer, by leveraging recent advances in deep learning, shows great promise in structural shape optimization for aerodynamic mitigation that is characterized as a typical nonlinear, high-dimensional, and nonconvex problem (Mnih et al., 2015).

Learning a DNN-based policy usually involves a large amount of data. As mentioned above, the generation of high-quality input–output data of structural aerodynamics (i.e., aerodynamic performance with each set of updated design variables) from CFD simulations is very expensive. To reduce the required training datasets, the prior domain knowledge can be leveraged to enhance the regularization mechanism during the training process of a

neural network (Psichogios & Ungar, 1992). For example, deep learning enhanced by "explicit" domain knowledge in terms of physics-based and/or semiempirical equations has been recently utilized for effective simulations of tropical cyclone winds (Snaiki & Wu, 2019) and nonlinear structural dynamics (Wang & Wu, 2020) with a small training dataset. It is noted that both the tropical cyclone winds and nonlinear structural dynamics are actually governed by the Newton's second law, and the corresponding "explicit" domain knowledge is represented by the Navier–Stokes equations and equations of motion, respectively. On the other hand, the no-free-lunch theorem for search and optimization indicates that a universal law and associated governing equations for the optimization system may not exist (Wolpert & Macready, 1997). Hence, it is challenging to incorporate the equation-based "explicit" domain knowledge into the deep RL-based shape optimizer. Accordingly, the equation-free "tacit" domain knowledge will be leveraged here to greatly enhance the training efficiency. In this study, both the specific direct-domain knowledge and general cross-domain knowledge extracted from the low-cost source tasks will be integrated into the deep RL-based shape optimizer for its applications to high-cost target tasks (Min, Sagarna, Gupta, Ong, & Goh, 2017). The former can be efficiently obtained via the transfer-learning technique based on low-fidelity simulations of the current optimization problem (e.g., Pan & Yang, 2009; Yan, Zhu, Kuang, & Wang, 2019), and the latter is usually acquired via the meta-learning technique based on a group of inexpensive tasks generated from a common probability distribution (e.g., multivariate Gaussian distribution) that reflects important high-level structures of the current optimization problem (e.g., Finn, Abbeel, & Levine, 2017; Zhou, Li, & Zare, 2017). It is noted that the "tacit" domain knowledge usually presents a heuristic nature and its inappropriate incorporation into the target task may result in a negative impact (Rosenstein, Marx, Kaelbling, & Dietterich, 2005). Accordingly, suitable relatedness (or similarity) and transferability measures between source and target tasks should be established (Eaton & Lane, 2008). In this study, the low-cost source tasks for extracting both direct- and cross-domain knowledge are carefully selected to avoid the negative knowledge transfer. Numerical examples of a simple case study (i.e., shape optimization of a high-rise building cross section to minimize its drag) are carried out by the proposed scheme as well as gradient-based and gradient-free algorithms. The comparison results demonstrate that an improved performance of the developed knowledge-enhanced deep RL-based shape optimizer for aerodynamic mitigation of wind-sensitive structures is achieved.

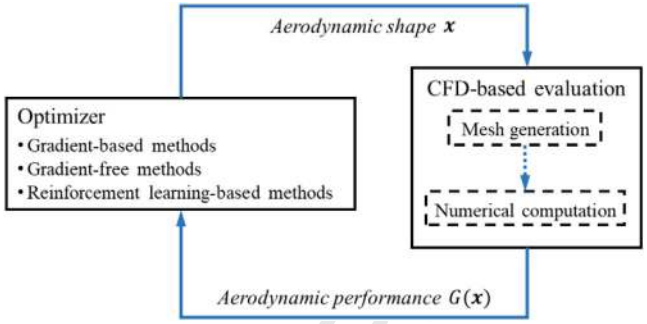

FIGURE 1 Typical process of aerodynamic shape optimization

2 | AERODYNAMIC SHAPE OPTIMIZATION SCHEMES

A general optimization problem could be simply formulated as

$$\min_{\boldsymbol{x}} G(\boldsymbol{x}) \tag{1a}$$

where $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]$ is a vector of n design variables and $G(\boldsymbol{x})$ is the objective function. The optimization process is usually subjected to R_c equality and/or S_c inequality constraints:

$$C_r(\boldsymbol{x}) = 0\, r = 1, 2, \ldots, R_c, \tag{1b}$$

$$D_s(\boldsymbol{x}) \le 0\, s = 1, 2, \ldots, S_c. \tag{1c}$$

In the setting of aerodynamic shape optimization of wind-sensitive structures, the design variables $\boldsymbol{x}$ characterize the external shape; the objective function $G(\boldsymbol{x})$ represents the aerodynamic performance of wind-sensitive structures (e.g., the drag coefficient for tall buildings or the critical flutter wind speed for long-span bridges); the equality and inequality constraints are usually based on practical considerations (e.g., geometric symmetry and structural dimension). A typical aerodynamic shape optimization process is shown in Figure 1. The optimization process utilizes an optimizer to propose a new design (i.e., to take "action") based on aerodynamic performance of current external shape (i.e., "state"). As mentioned in the previous section, the evaluations of aerodynamic performance for bluff-body structures using CFD are very expensive due to the nature of turbulent flow field and intense flow separation. Hence, it is highly desirable to reduce the amount of CFD-based performance evaluations for the aerodynamic shape optimization problems. To this end, a sample-efficient optimizer that can achieve the globally optimal solution with a relatively small number of

iterations is needed. In this section, the conventional optimization schemes used for comparison purposes in this study are first briefly reviewed for the sake of completeness. Then, the newly developed deep RL-based optimization schemes are introduced in the context of aerodynamic shape optimization.

2.1 | Conventional optimization schemes

The conventional optimization schemes for aerodynamic shape optimization could be generally classified into gradient-based and gradient-free methods. Based on the argument that the objective function decreases fastest in the direction of negative gradient, the basic gradient descent takes the increment of design variables proportional to the negative gradient of objective function at current design (Skinner & Zare-Behtash, 2018). The gradients here are computed using a simple finite-difference method. It is noted that the computational cost for optimization may be reduced by using other approaches to calculate gradients (e.g., adjoint method). Since the gradient-based methods follow a deterministic rule to calculate the next design (i.e., moving greedily in direction of steepest descent based on gradient), the optimization process is likely to be stuck in local optima (Skinner & Zare-Behtash, 2018).

To address the issue of being trapped in local optima, the gradient-free methods could be introduced to better search for global optima. Instead of using a deterministic search and update rule as in the gradient-based methods, the gradient-free methods (e.g., genetic algorithm, PSO, and simulated annealing) usually follow rules that allow for random explorations in the design space. Furthermore, they can benefit from working with a population of candidate designs to search the design space with shared information among the population (Skinner & Zare-Behtash, 2018). Despite the high chance of finding global optima, the gradient-free methods usually need a significant number of samples.

2.2 | Deep reinforcement learning-based optimization schemes

Considering the inherent limitations resulting from the hand-designed search and update rules for conventional gradient-based and gradient-free methods, it is desirable to design an auto-learned rule for optimization problems:

$$\boldsymbol{x}_{t+1} = \pi\left[\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t, G\left(\boldsymbol{x}_0\right), G\left(\boldsymbol{x}_1\right), \ldots, G\left(\boldsymbol{x}_t\right)\right], \quad (2)$$

where function π represents a general form of effective optimization scheme in terms of reaching global optima with a relatively small number of iterations. It can be effectively obtained based on the RL methodology without human intervention (Silver et al., 2017).

2.2.1 | RL in aerodynamic shape optimization

The mathematical model for RL in a fully observable environment is usually based on the Markov decision process characterized by a tuple $[S, A, P(s_t, s_{t+1}, a_t), R(s_t, s_{t+1}, a_t)]$, where S and A denote the set of state s and action a, respectively; $P(s_t, s_{t+1}, a_t)$ is the state-transition probability from state s_t to state s_{t+1} under action a_t (Sutton & Barto, 2018). After moving from s_t to s_{t+1} under action a_t, an immediate reward r_t is received from the environment based on the reward function $R(s_t, s_{t+1}, a_t)$. It is noted that $P(s_t, s_{t+1}, a_t)$ and $R(s_t, s_{t+1}, a_t)$ are the properties of the environment. In RL, an agent aims to learn a policy π that maps from state to action [i.e., $a = \pi(s)$] such that the expected cumulative reward $\mathrm{E}(\sum_{k=0}^{\infty} \gamma^k r_{t+k})$ (also known as return $R_{\text{return}} = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$) is maximized, where the discount factor γ (usually $0 \leq \gamma \leq 1$) determines the relative importance of future reward compared with immediate reward. The policy maximizing the expected cumulative reward is known as the optimal policy π_*. Unlike a closely related field of dynamic programming with explicitly given environment dynamics (i.e., the state-transition probability and reward function), the environment in RL is usually unknown and the optimal policy π_* is learned based only on the agent's interaction experiences with the observable environment.

A schematic description of RL to learn an effective search and update rule for aerodynamic shape optimization π is shown in Figure 2. The term "effective" policy is used here instead of "optimal" policy since it is usually difficult to exhaust the policy space to find the best one in practice. It is noted that the strict optimality of the policy is generally not the primary concern considering effectively finding the optimal aerodynamic shape is the focus. Although only $\boldsymbol{x}_t$ and $G(\boldsymbol{x}_t)$ are shown in Figure 2 for the sake of simplicity, the state could include all previously evaluated designs $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t$ and their performance $G(\boldsymbol{x}_0), G(\boldsymbol{x}_1), \ldots, G(\boldsymbol{x}_t)$ in wind environment as indicated in Equation (2). The action resulting from the policy determines the new design $\boldsymbol{x}_{t+1}$. The RL agent (i.e., the structure) interacts with the wind environment to obtain an effective policy π such that the optimal aerodynamic shape could be found within limited number of steps, which is learned by maximizing the user-defined rewards based on

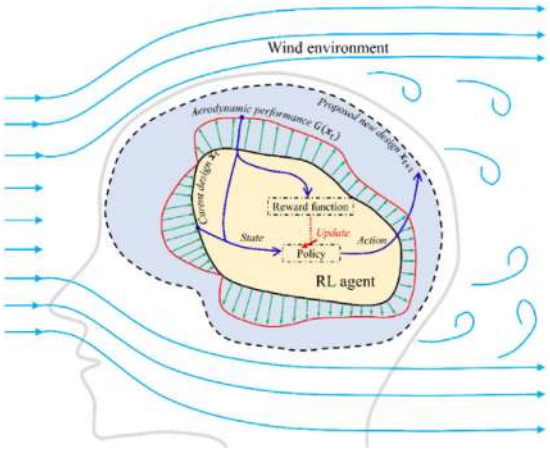

FIGURE 2 Schematic of RL in aerodynamic shape optimization

selected RL algorithm. Among numerous RL algorithms, the value-based and policy-based methods are two most popularly used ones in optimization problems.

2.2.2 | Value-based methods

In addition to the automated trial-and-error search through interacting with the environment, another important feature of RL is the use of delayed reward. In such a case, actions may affect not only the immediate reward but also the next situation, and through that, all subsequent rewards, and accordingly a state-value function $v_\pi(s)$ is defined as the expected cumulative future reward starting from the state s and following policy π afterwards (Sutton & Barto, 2018). The optimal policy π_* corresponds to the optimal state-value function $v_{\pi_*}(s)$ that is larger than those following all other policies for all states. However, it is impossible to extract π_* based only on $v_{\pi_*}(s)$ due to the lack of action knowledge. To address this issue, the action-value function $q_\pi(s,a)$ (also known as the state-action value) is introduced as the expected cumulative future reward starting from the state s, taking action a and following policy π afterwards (Sutton & Barto, 2018). The optimal policy could be identified by searching a greedy action that leads to highest value, that is, $a = \underset{a}{\text{argmax}}[q_{\pi_*}(s,a)]$ (where $q_{\pi_*}(s,a)$ is optimal action-value function). Most of value-based methods to obtain π_* (e.g., Q learning) are based on the Bellman equation, which recursively relates the action value of current state to sum of the immediate reward and the discounted action value of next state (Sutton & Barto, 2018):

$$Q_{k+1}(s_t,a_t) = Q_k(s_t,a_t) + \eta_Q\left[r_t + \gamma \max_a Q_k(s_{t+1},a) - Q_k(s_t,a_t)\right], \quad (3)$$

where capital Q indicates an estimate of lower-case q; subscript "k" represents the iteration number; η_Q is the learning rate.

For a high-dimensional continuous state space, the tabular representation of Q functions (e.g., a lookup table) in conventional Q learning could be replaced by function approximators (e.g., a DNN). In a deep Q learning, the input of deep Q network is the high-dimensional continuous state and the output is the Q value for each discrete action (Mnih et al., 2015). It is noted that the combination of Q learning with the DNN-based function approximations often suffers from divergence due mainly to two reasons, namely strong correlation between the consecutive samples $(s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, ...)$ and nonstationarity of the target $[r_t + \gamma \max_a Q_k(s_{t+1},a)]$ in Equation (3) (Mnih et al., 2015). To address the divergence issue from correlation, a replay buffer is usually employed to store the past experiences and the randomly sampled experiences from the replay buffer are utilized to update the deep Q network. The use of replay buffer not only removes the strong correlation of samples but also improves sample efficiency with repetitively accessed learning experiences. To overcome the divergence issue from nonstationarity, an additional DNN called a target Q network with weights slowly tracking that of the original Q network is introduced to make the target Q value changing slowly and hence improve stability (Mnih et al., 2015). Although the deep Q learning shows very promising results in solving complicated tasks with high-dimensional continuous state space due to the powerful function approximation ability of DNN, it can only handle the low-dimensional discrete action space due to the curse of dimensionality (Lillicrap et al., 2016). To consider general application of RL to aerodynamic shape optimization, the policy in Figure 2 needs to be represented by a DNN mapping from continuous states to continuous actions.

2.2.3 | Policy-based methods

Policy-based methods are popularly employed in RL problems with a continuous action space. In contrast to value-based methods, the policy-based algorithms directly learn a parameterized policy $a = \pi(s|\theta)$ (where θ is the policy parameters) mapping states to high-dimensional continuous actions without using the action-value function as the intermediary to compute the policy. In the policy-based methods, the RL agent directly updates the policy parameter θ (e.g., the weights of DNN) by gradient ascent (Sutton & Barto, 2018):

$$\theta_{k+1} = \theta_k + \eta_{PB}\nabla_\theta J(\theta_k), \quad (4)$$

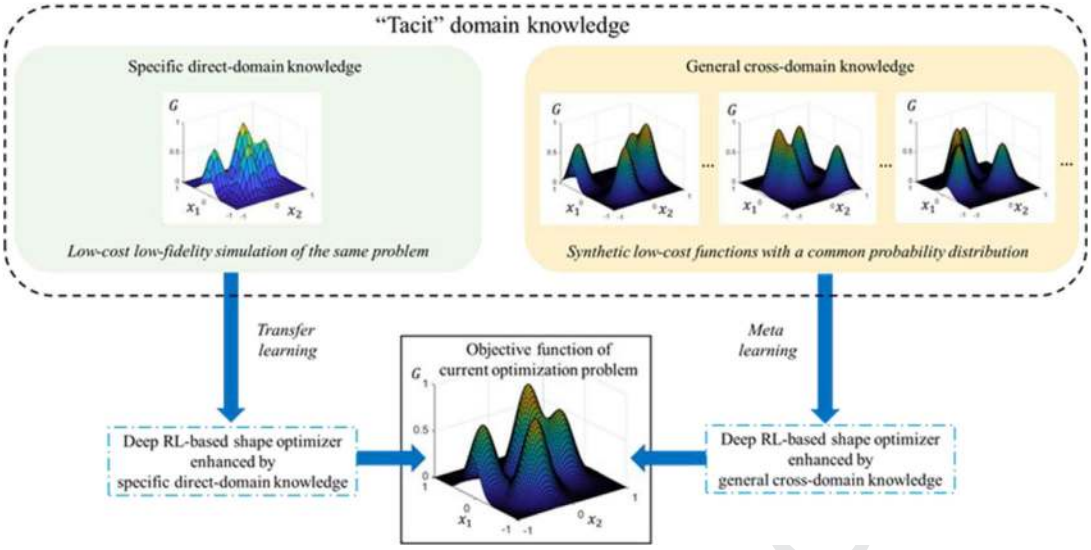

FIGURE 3 Incorporation of "tacit" knowledge into deep RL-based aerodynamic shape optimizer

where η_{PB} is the learning rate; $J(\theta_k)$ is a performance index of current policy $\pi(s|\theta_k)$ in terms of expected cumulative reward and could be estimated with the cumulative reward (i.e., return R_{return}) of sampled sequences using current policy; $\nabla_\theta J(\theta_k)$ is the gradient of performance index with respect to policy parameters θ.

Although it is straightforward and effective to adjust the policy parameters in the direction of policy gradient, R_{return} and hence obtained gradients usually present high variance in a stochastic environment and learning difficulty may occur. To reduce the variance of policy gradients, the action value $Q(s,a)$ following current policy is employed to estimate $J(\theta_k)$ (Sutton & Barto, 2018). The obtained "actor-critic" scheme actually inherits essential features from both policy-based and value-based methods, where the "actor" proposes the action $a = \pi(s|\theta)$ as in policy-based methods and the "critic" evaluates the quality of the action (i.e., action value $Q[s, a = \pi(s|\theta)]$) as in value-based methods. In the application of deep Q function to the "actor-critic" scheme, the previously mentioned numerical tricks of replay buffer and target network for deep Q learning should also be adopted to improve the learning performance.

3 | KNOWLEDGE-ENHANCED DEEP RL-BASED SHAPE OPTIMIZER

Deep RL algorithms usually start from a random policy (i.e., DNN with randomly initialized weights), and hence a large amount of interactions with the high-cost CFD environment may be necessary for convergence to an effective DNN-based policy. To further enhance the search efficiency, the domain knowledge will be incorporated into the current learning problem. Unlike recent attempts of using the "explicit" domain knowledge in terms of physics-based and/or semiempirical equations to enhance training efficiency of conventional deep learning (Snaiki & Wu, 2019; Wang & Wu, 2020), the domain knowledge utilized to efficiently obtain aerodynamic shape optimization policy is equation-free "tacit" domain knowledge due to the nonexistence of a universal law and associated governing equations for the optimization system according to the no-free-lunch theorem for search and optimization (Wolpert & Macready, 1997). As shown in Figure 3, the "tacit" knowledge extracted from the low-cost environment in this study includes both specific direct-domain knowledge and general cross-domain knowledge. The former is obtained via the transfer-learning technique based on low-fidelity simulations of the current optimization problem (e.g., Pan & Yang, 2009; Yan et al., 2019), and the latter is acquired via the meta-learning technique based on a group of inexpensive tasks generated from a common probability distribution (e.g., multivariate Gaussian distribution) that reflects important high-level structures of the current optimization problem (e.g., Finn et al., 2017; Zhou et al., 2017).

3.1 | Incorporation of specific direct-domain knowledge via transfer learning

To incorporate the specific direct-domain knowledge in the deep RL-based shape optimizer, the widely used transfer learning in deep learning community, which stores knowledge gained from solving one problem (source task) and applies it to a different but related problem (target

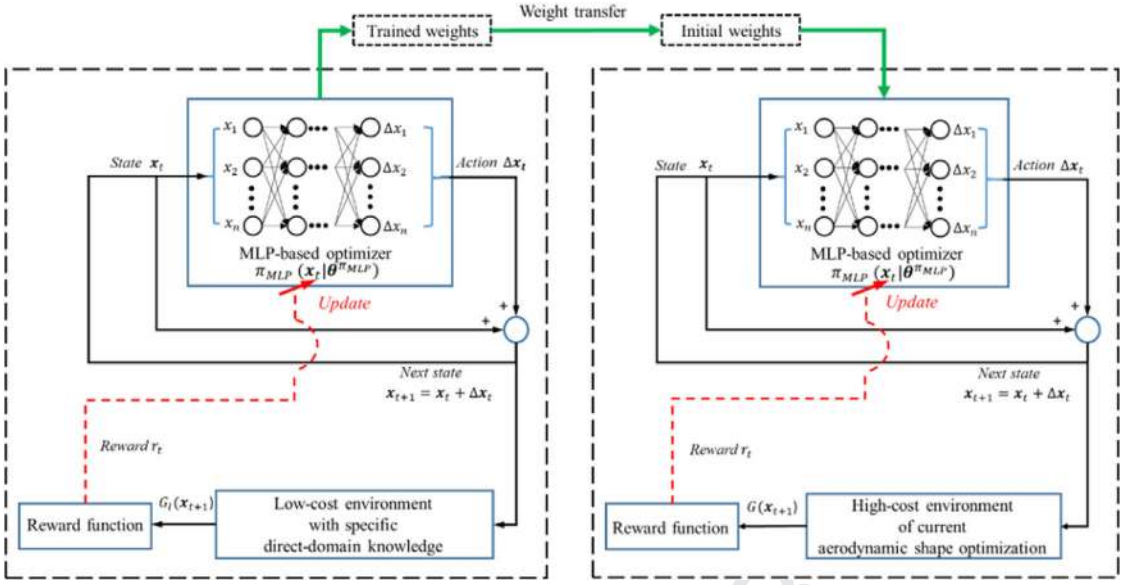

FIGURE 4 Incorporation of specific direct-domain knowledge into deep RL-based shape optimizer via transfer learning

task), is utilized here (Pan & Yang, 2009). Among various transfer learning schemes (e.g., instance-based, feature-based, parameter-based, and relational-based ones), this study utilizes the parameter-based approach transferring the knowledge in terms of the weights of DNN-based policy from the source task (e.g., in a low-cost environment based on Reynolds-averaged Navier–Stokes [RANS] scheme) to the target task (e.g., in a high-cost environment based on large-eddy simulation [(LES] scheme). As shown in Figure 4, the policy is represented by a fully connected feedforward DNN (i.e., a multilayer perceptron [MLP]). The input (state) of the MLP is the current design $\boldsymbol{x}_t$ and the output (action) is the design variation $\Delta\boldsymbol{x}_t$ for the calculation of the next design $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \Delta\boldsymbol{x}_t$:

$$\Delta \boldsymbol{x}_t = \pi_{\mathrm{MLP}}\left(\boldsymbol{x}_t \theta^{\pi_{\mathrm{MLP}}}\right), \tag{5}$$

where the $\theta^{\pi_{\mathrm{MLP}}}$ is the weights of the policy network π_{MLP}. Since there is a unique objective function in transfer learning, an effective search direction at current design $\boldsymbol{x}_t$ is sufficient for the purpose of optimization. Hence, only $\boldsymbol{x}_t$ is utilized here as state and input to the MLP-based policy. In addition, the $G(\boldsymbol{x}_t)$ can be fully determined by $\boldsymbol{x}_t$ and hence not included as state. It is noted that the objective function in the low-cost environment is denoted as $G_l(\boldsymbol{x})$ to differentiate from the objective function $G(\boldsymbol{x})$ in high-cost environment. The reward r_t received at each step for minimizing the objective function is chosen to be the aerodynamic performance improvement of proposed new design $G_l(\boldsymbol{x}_{t+1})$ compared to the baseline design $G_l(\boldsymbol{x}_0)$, that is, $r_t = G_l(\boldsymbol{x}_0) - G_l(\boldsymbol{x}_{t+1})$. In the case where the optimization purpose is to maximize the objective function, a negative sign needs to be added to the reward.

The deep deterministic policy gradient (DDPG) algorithm, which has been successfully applied in numerous continuous control tasks with high sample-efficiency (Lillicrap et al., 2016), is utilized to obtain the effective policy for aerodynamic shape optimization. In addition to the policy network $\pi_{\mathrm{MLP}}(\boldsymbol{x}_t|\theta^{\pi_{\mathrm{MLP}}})$, there are three additional DNN in DDPG, namely Q network $[Q_{\mathrm{MLP}}(\boldsymbol{x}_t, \Delta\boldsymbol{x}_t|\theta^{Q_{\mathrm{MLP}}})]$, target policy network $[\pi_{\mathrm{MLP}}{}'(\boldsymbol{x}_t|\theta^{\pi_{\mathrm{MLP}}{}'})]$ and target Q network $[Q_{\mathrm{MLP}}{}'(\boldsymbol{x}_t, \Delta\boldsymbol{x}_t|\theta^{Q_{\mathrm{MLP}}{}'})]$, with $\theta^{Q_{\mathrm{MLP}}}$, $\theta^{\pi_{\mathrm{MLP}}{}'}$, and $\theta^{Q_{\mathrm{MLP}}{}'}$ representing their corresponding weights. The policy $\pi_{\mathrm{MLP}}(\boldsymbol{x}_t|\theta^{\pi_{\mathrm{MLP}}})$ is learned in an "actor-critic" mode. The "actor," represented by policy network $\pi_{\mathrm{MLP}}(\boldsymbol{x}_t|\theta^{\pi_{\mathrm{MLP}}})$, follows a deterministic policy to output the action $\Delta\boldsymbol{x}_t$ based on the observed state $\boldsymbol{x}_t$. The "critic," represented by the Q network $Q_{\mathrm{MLP}}(\boldsymbol{x}_t, \Delta\boldsymbol{x}_t|\theta^{Q_{\mathrm{MLP}}})$, evaluates the actor's action value to provide valuable update information by encouraging the actions leading to large future rewards (large Q values) and penalizing the actions leading to small future rewards (small Q values). The delayed copies of the policy network $\pi_{\mathrm{MLP}}(\boldsymbol{x}_t|\theta^{\pi_{\mathrm{MLP}}})$ and Q network $Q_{\mathrm{MLP}}(\boldsymbol{x}_t, \Delta\boldsymbol{x}_t|\theta^{Q_{\mathrm{MLP}}})$ are used to compute the target policy network $\pi_{\mathrm{MLP}}{}'(\boldsymbol{x}_t|\theta^{\pi_{\mathrm{MLP}}{}'})$ and target Q network $Q_{\mathrm{MLP}}{}'(\boldsymbol{x}_t, \Delta\boldsymbol{x}_t|\theta^{Q_{\mathrm{MLP}}{}'})$. The learning details based on DDPG is presented in Algorithm 1. The trained weights of policy networks (and other three networks) are utilized as the initial weights for training progress in the high-cost environment using the same DDPG algorithm.

3.2 | Incorporation of general cross-domain knowledge via meta learning

In the case that the specific direct-domain knowledge extracted from the low-cost environment (e.g., low-fidelity

Algorithm 1 Training MLP-based optimizer using DDPG

Initialize the policy network $\pi_{MLP}(\boldsymbol{x}_t|\boldsymbol{\theta}^{\pi_{MLP}})$ and the Q network $Q_{MLP}(\boldsymbol{x}_t, \Delta\boldsymbol{x}_t|\boldsymbol{\theta}^{Q_{MLP}})$
Initialize two target networks $\pi_{MLP}{}'(\boldsymbol{x}_t|\boldsymbol{\theta}^{\pi_{MLP}{}'})$ and $Q'_{MLP}(\boldsymbol{x}_t, \Delta\boldsymbol{x}_t|\boldsymbol{\theta}^{Q_{MLP}{}'})$ with copied weight $\boldsymbol{\theta}^{\pi_{MLP}{}'} = \boldsymbol{\theta}^{\pi_{MLP}}$, $\boldsymbol{\theta}^{Q_{MLP}{}'} = \boldsymbol{\theta}^{Q_{MLP}}$
Initialize replay buffer as an empty set
While not convergent **do**
  Take the initial design $\boldsymbol{x}_0$ as current state $\boldsymbol{x}_t = \boldsymbol{x}_0$
  For i_{step}=0, 1, 2, …, n_{step} **do**
    Select an action based on policy network: $\Delta\boldsymbol{x}_t = \pi_{MLP}(\boldsymbol{x}_t|\boldsymbol{\theta}^{\pi_{MLP}})$
    Add random noise N (e.g., Gaussian noise) to the action to encourage exploration: $\Delta\boldsymbol{x}_t = \Delta\boldsymbol{x}_t + \boldsymbol{N}$
    Execute action $\Delta\boldsymbol{x}_t$, obtain reward r_t and observe new state $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \Delta\boldsymbol{x}_t$
    Store the experience $(\boldsymbol{x}_t, \Delta\boldsymbol{x}_t, r_t, \boldsymbol{x}_{t+1})$ in replay buffer
    Sample n_{batch} experiences from replay buffer denoted as $(\boldsymbol{x}_t^i, \Delta\boldsymbol{x}_t^i, r_t^i, \boldsymbol{x}_{t+1}^i)$, where i=1, 2, …, n_{batch}
    For i=1, 2, …, n_{batch} **do**
      Set $y_i = r_t^i + \gamma Q'_{MLP}[\boldsymbol{x}_{t+1}^i, \pi_{MLP}{}'(\boldsymbol{x}_{t+1}^i|\boldsymbol{\theta}^{\pi_{MLP}{}'})|\boldsymbol{\theta}^{Q_{MLP}{}'}]$
    End for
    Update Q network by gradient descent with learning rate η_Q:

$$\boldsymbol{\theta}^{Q_{MLP}} = \boldsymbol{\theta}^{Q_{MLP}} - \eta_Q \frac{1}{n_{batch}} \sum_{i=1}^{n_{batch}} \nabla_{\boldsymbol{\theta}^{Q_{MLP}}} [y_i - Q_{MLP}(\boldsymbol{x}_t^i, \Delta\boldsymbol{x}_t^i|\boldsymbol{\theta}^{Q_{MLP}})]^2$$

    Update policy network by gradient ascent with learning rate η_π:

$$\boldsymbol{\theta}^{\pi_{MLP}} = \boldsymbol{\theta}^{\pi_{MLP}} + \eta_\pi \frac{1}{n_{batch}} \sum_{i=1}^{n_{batch}} \nabla_{\boldsymbol{\theta}^{\pi_{MLP}}} Q_{MLP}[\boldsymbol{x}_t^i, \pi_{MLP}(\boldsymbol{x}_t^i|\boldsymbol{\theta}^{\pi_{MLP}})|\boldsymbol{\theta}^{Q_{MLP}}]$$

    Update the two target networks with the update factor τ:

$$\boldsymbol{\theta}^{Q_{MLP}{}'} = \tau\boldsymbol{\theta}^{Q_{MLP}} + (1-\tau)\boldsymbol{\theta}^{Q_{MLP}{}'}, \boldsymbol{\theta}^{\pi_{MLP}{}'} = \tau\boldsymbol{\theta}^{\pi_{MLP}} + (1-\tau)\boldsymbol{\theta}^{\pi_{MLP}{}'}$$

    Assign the new state as the current state $\boldsymbol{x}_t = \boldsymbol{x}_{t+1}$
  End for
End while

simulations) is not available, the meta-learning technique may be utilized to incorporate general cross-domain knowledge extracted from a set of prescribed functions that share high-level similarities with the objective function of the current aerodynamic shape optimization problem into the deep RL-based aerodynamic shape optimizer. Meta learning has recently drawn a great attention due to the fact that knowledge obtained from learning to master a set of tasks can be generalized to master new tasks (Finn et al., 2017), and it is used here to leverage the generalization ability of the trained deep RL-based optimizer based on a set of prescribed low-cost functions for optimizing the high-cost objective functions for an unseen aerodynamic shape optimization problem. Since the effective optimization policy in meta learning is obtained from and will be utilized for optimizing a large number of functions, the current design $\boldsymbol{x}_t$ and corresponding objective function $G(\boldsymbol{x}_t)$ may not be able to provide enough information to take next action. In this study, all previously evaluated designs and associated aerodynamic performance [i.e., $\boldsymbol{x}_0$, $G(\boldsymbol{x}_0), \boldsymbol{x}_1$, $G(\boldsymbol{x}_1), \ldots, \boldsymbol{x}_t$, $G(\boldsymbol{x}_t)$] are considered as state input to the policy network. Accordingly, the recurrent neural networks (RNN) is used to parameterize the policy due to its convenience to pass information across time steps (Li, Wu, & Liu, 2020; Wu & Kareem, 2011). In addition, RNN utilizes shared weights for different time steps and hence greatly reduces the number of weights to be learned in an optimization problem. To solve the issue of vanishing and exploding gradients in backpropagation for a plain-vanilla RNN, the long-short time memory (LSTM) cell is utilized to replace the conventional neurons in the hidden layers (Chen et al., 2017; Zhou et al., 2017). As shown in Figure 5, the proposed next design of aerodynamic shape $\boldsymbol{x}_{t+1}$ (and associated vector $\boldsymbol{h}_{t+1}$ representing cell output) is calculated through the RNN with LSTM cell (denoted by π_{RNN} with weights of θ^{RNN}):

$$[\boldsymbol{x}_{t+1}, \boldsymbol{h}_{t+1}] = \pi_{\mathrm{RNN}}\ [\boldsymbol{x}_t,\ G(\boldsymbol{x}_t),\ \boldsymbol{h}_t|\theta^{\mathrm{RNN}}, \qquad (6)$$

where cell output vector $\boldsymbol{h}_t$ is composed of LSTM cell state and hidden state with the memory contributions from all previously evaluated designs and their associated objective functions $[\boldsymbol{x}_0, G(\boldsymbol{x}_0), \boldsymbol{x}_1, G(\boldsymbol{x}_1), \ldots, \boldsymbol{x}_{t-1}, G(\boldsymbol{x}_{t-1})]$. The effective optimization policy π_{RNN} is achieved by maximizing the cumulative reward (return) of the n_{step} optimization steps in the low-cost environment, i.e., $R_{\mathrm{return}} = -\sum_{t=1}^{n_{\mathrm{step}}} G_l(\boldsymbol{x}_t)$. It should be noted that functions used to construct the low-cost environment in meta learning is usually synthesized based on a common probability distribution (e.g., multivariate Gaussian distribution) and easy to learn, and hence the basic policy-gradient algorithm performs well. The learning details based on policy gradient used in this study is described in Algorithm 2. The trained RNN-based optimizer in the low-cost environment

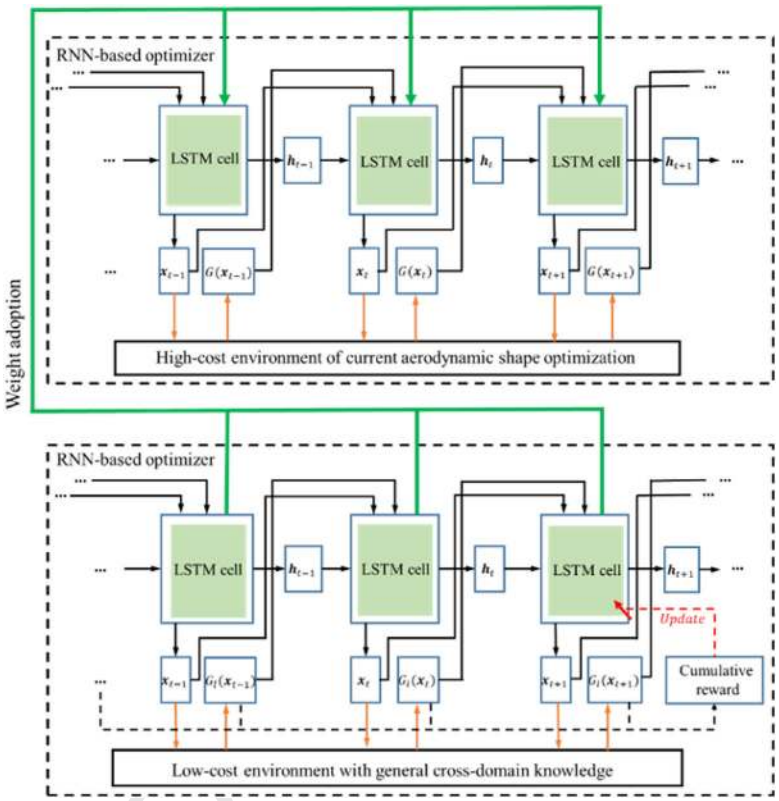

FIGURE 5 Incorporation of general cross-domain knowledge into deep RL-based shape optimizer via meta learning

Algorithm 2 Training RNN-based optimizer using policy gradient

Initialize the RNN weights $\boldsymbol{\theta}^{RNN}$
While not convergent **do**
  Sample a mini batch from the set of prescribed objective functions with batch size of n_b: $G_1, G_2, \ldots, G_{n_b}$
  For i_b=1, 2,…, n_b **do**
    Initialize $\boldsymbol{h}_0$ and the $\boldsymbol{x}_0$, and then calculate $G_{i_b}(\boldsymbol{x}_0)$
    For t=0, 1,…, n_{step} **do**
      Determine $[\boldsymbol{x}_{t+1}, \boldsymbol{h}_{t+1}] = \pi_{RNN}[\boldsymbol{x}_t, G_{i_b}(\boldsymbol{x}_t), \boldsymbol{h}_t | \boldsymbol{\theta}^{RNN}]$ and calculate $G_{i_b}(\boldsymbol{x}_{t+1})$
    End for
    Calculate the cumulative reward: $R^{i_b}_{return} = -\sum_{t=1}^{n_{step}} G_{i_b}(\boldsymbol{x}_t)$
  End for
  Update the policy by gradient ascent with learning rate η_{RNN}: $\boldsymbol{\theta}^{RNN} = \boldsymbol{\theta}^{RNN} + \eta_{RNN} \frac{1}{n_b} \sum_{i_b=1}^{n_b} \nabla_{\boldsymbol{\theta}^{RNN}} R^{i_b}_{return}$
End while

is directly utilized in current aerodynamic shape optimization process without further learning to highlight its generalization ability.

4 | NUMERICAL EXAMPLES

Due to the extremely high computational demand for three-dimensional (3D) CFD simulations (e.g., Elshaer & Bitsuamlak, 2018; Kim & Yhim, 2014), a simple case study of aerodynamic shape optimization of the cross section of a typical high-rise building is used to demonstrate the improved performance of the proposed knowledge-enhanced deep RL-based aerodynamic shape optimizer compared to conventional gradient-based and gradient-free schemes. Since state of the practice in analyzing wind effects on 3D realistic slender structures is essentially based on the two-dimensional (2D) cross-section aerodynamic properties according to strip theory (e.g., Davenport, 1962; Hao & Wu, 2018; Hou & Sarkar, 2018; Scanlan, 1978), the 2D numerical examples can be considered as the fundamental building blocks for more complex 3D realistic scenarios. As shown in Figure 6, the baseline design is a square with nondimensional width $D = 1$ and rounded corners with radius $r_c = 0.4$, and the straight-line segments are fixed while the rounded parts are allowed to change. The optimization task is to minimize the mean drag force coefficient μ_{Cd} by updating the two design

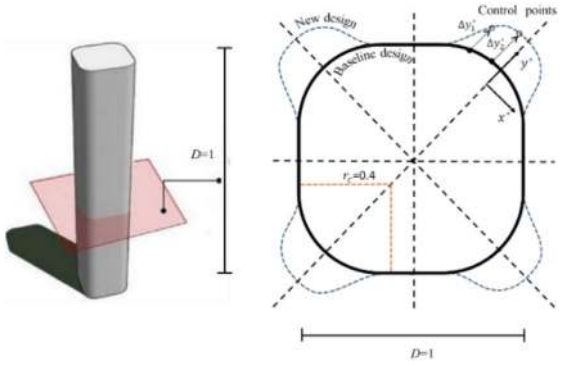

FIGURE 6 Geometric configuration of a typical tall-building cross section

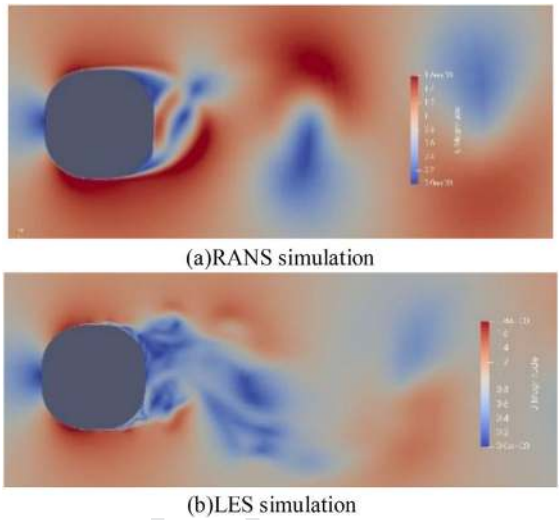

FIGURE 7 Velocity fields of CFD simulations for a selected shape design

variables Δy_1^* and Δy_2^* defined as the relative displacements in the y^* direction with respect to the control points in a baseline design located at $x_1^* = -0.16$ and $x_2^* = -0.04$ (Ding & Kareem, 2018). The resulting aerodynamic shape is obtained through interpolation with a cubic spline passing through the new control points. Considering the existence of the four axes of symmetry, the geometry could be fully determined by the two design variables Δy_1^* and Δy_2^*. The constraints $|\Delta y_1^*| \leq 0.1$ and $|\Delta y_2^*| \leq 0.1$ are imposed to limit the maximum allowable geometric change. It is noted that the parameterization scheme used here has been successfully applied to the aerodynamic shape optimization of tall buildings (Bernardini, Spence, Wei, & Kareem, 2015; Ding & Kareem, 2018). Although the 2D numerical examples are employed for the sake of convenience, it is expected that the knowledge-enhanced deep RL-based shape optimizer will present greater promise in high-dimensional applications due mainly to the inclusion of DNN function approximators (Chen et al., 2017; Yan et al., 2019).

4.1 | Specific direct-domain knowledge-enhanced deep RL-based optimization

To effectively incorporate the specific direct-domain knowledge into the current aerodynamic shape optimization process, the low-fidelity low-cost RANS simulations and high-fidelity high-cost LES simulations are employed as the source and target tasks, respectively. The RANS-level and LES-level CFD simulations given by Ding and Kareem (2018) are utilized to evaluate the mean drag force coefficient μ_{Cd}. The CFD simulations are carried out based on the Open Source Field Operation and Manipulation (OpenFOAM) C ++ class library, where the spatial domain is discretized utilizing the Finite volume method. Specifically, a uniform wind flow approaching the building at a fixed angle of attack is considered. The Reynolds number is 10^5 for both RANS and LES. In a computational domain of 30D × 20D (where D is the cross-section width), structured mesh is utilized with a mesh number of around 435,000 (for RANS) and 1,418,000 (for LES), and the mesh independence is checked following the AJI guideline (Tominaga et al., 2008). The k–ω shear stress transport model is used for RANS, while dynamic Lagrangian subgrid-scale model for LES. The y+ number is around 40 (with standard wall functions) for RANS and 1 for LES. The Courant–Friedrichs–Lewy numbers for RANS and LES are 0.1 and 0.4, respectively. The numerical simulations are terminated after four cycles of flow passing the computational domain. The computational time for one 2D shape design is around 10 and 150 h for RANS and LES using 16 CPU cores [Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50 GHz]. The representative velocity fields of RANS and LES for a selected shape design are shown in Figure 7. It is noted that the RANS-based simulations are first used for establishing a reliable surrogate model over the design space to save the computational demand (Ding & Kareem, 2018). Hence, the computational cost in low-cost environment is considered as negligible compared to that in high-cost environment. In this numerical example, the computational budget is restricted to 20 LES-based evaluations of the objective functions for each method. Since each "step" represents one LES-based evaluation, the maximum step n_{step} for the optimization is 20. The hyperparameters of DDPG used in the numerical example are shown in Table 1. The trained weights of the four networks in DDPG in the low-cost RANS environment are used as the initial weights for learning and optimizing processes in the high-cost environment.

TABLE 1 Hyperparameters of DDPG

Hyperparameters	Values
Number of layers (policy network)	4
Number of neurons (policy network)	40
Learning rate of policy network η_π	0.0001
Activation functions in hidden layers (policy network)	Rectified linear unit
Activation functions in output layer (policy network)	Hyper tangent
Number of layers (Q network)	4
Number of neurons (Q network)	40
Learning rate of Q network η_Q	0.001
Activation functions in hidden layers (Q network)	Rectified linear unit
Activation functions in output layer (Q network)	Linear
Discount factor γ	0.99
Update factor τ	0.001
Batch size n_{batch}	128

For the sake of simplicity, the basic gradient descent (Skinner & Zare-Behtash, 2018) and PSO (Kennedy & Eberhart, 1995) are, respectively, selected as examples of gradient-based and gradient-free optimization schemes, respectively, and used for comparison with deep RL-based shape optimizer. For a fair comparison, the parameters of the basic gradient descent and PSO methods are selected to ensure that both of them present good performance in the optimization based on low-cost simulations. The step size η_{BGD} of the basic gradient descent is set to be 0.01, and each step requires extra evaluations of the objective function to compute its gradient. The population size n_{pop} and step size η_{PSO} of PSO are set to be 5 and 1, respectively. The starting point is taken as (−0.1, −0.1) far from the optimal design for effective evaluation of various optimization schemes. The sampled designs for different methods are shown in Figure 8a–d, where the cross mark with a step number aside denotes the proposed designs in the optimization process. It is noted that the deep RL-based optimizer without integrating domain knowledge (i.e., directly interacting with the high-cost environment), as shown in Figure 8c, does not necessarily perform better than conventional optimization schemes for this simple 2D case considering the random initialization of the optimization policy. The comparison results in Figure 8f indicate that the specific direct-domain knowledge from RANS-level simulations can greatly facilitate the efficient search of deep RL-based optimizer for the optimal (or near optimal) design in LES-based simulations, which demonstrate that direct-domain knowledge-enhanced deep RL-based optimizer outperforms both gradient-based and gradient-free optimization algorithms in this case study. The two design variables are −0.011 and 0.028 for the selected optimal shape, which results in a drag coefficient of 0.276. The small oscillations in the sampled designs as indicated in Figure 8d may be attributed to the synchronized learning and optimizing processes in the high-cost environment.

TABLE 2 Hyperparameters of policy gradient

Hyperparameters	Values
LSTM cell state and hidden state size	100
Learning rate of policy network η_RNN	0.0001
Batch size n_b	128

4.2 | General cross-domain knowledge-enhanced deep RL-based optimization

To effectively incorporate the general cross-domain knowledge into the current optimization process, it is important to select a set of appropriate prescribed functions for constructing a low-cost environment. For the aerodynamic shape optimization of tall buildings, the objective function is likely to have multiple local optima. Considering the nonconvex continuous Gaussian process functions are often used for the synthesis of functions with multiple local optima (e.g., Chen et al., 2017; Zhou et al., 2017), they are utilized here by assuming that the high-cost objective functions of aerodynamic shape optimization could be well approximated by a mixture of Gaussian process functions. Accordingly, a set of supporting points $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m]$ $(m \times d)$ and their corresponding values $\boldsymbol{G}_s = [G_1, G_2, \ldots G_m]$ $(m \times 1)$ are first randomly generated, where m represents the number of supporting points and d denotes the dimension of design variables. The synthetic Gaussian process function passing the supporting points is then given by

$$G_l(\boldsymbol{x}) = (\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}}^{-1}\boldsymbol{G}_s)^T \boldsymbol{K}_{\boldsymbol{X}}(\boldsymbol{x}) \tag{7}$$

where the essential element in matrix $\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}}$ (m by m) is $\exp(-\frac{|\boldsymbol{x}_i-\boldsymbol{x}_j|^2}{2l^2})$ (for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, m$) and the essential element in matrix $\boldsymbol{K}_{\boldsymbol{X}}(\boldsymbol{x})$ (m by 1) is $\exp(-\frac{|\boldsymbol{x}_k-\boldsymbol{x}|^2}{2l^2})$ (for $k = 1, 2, \ldots, m$). The parameters m and l are set to be 6 and 0.5, respectively, in this case. A total of 4000 synthetic Gaussian process functions are utilized to train the RNN-based shape optimizer using policy-gradient algorithm, and the used hyperparameters are shown in Table 2. The obtained RNN-based shape optimizer and the

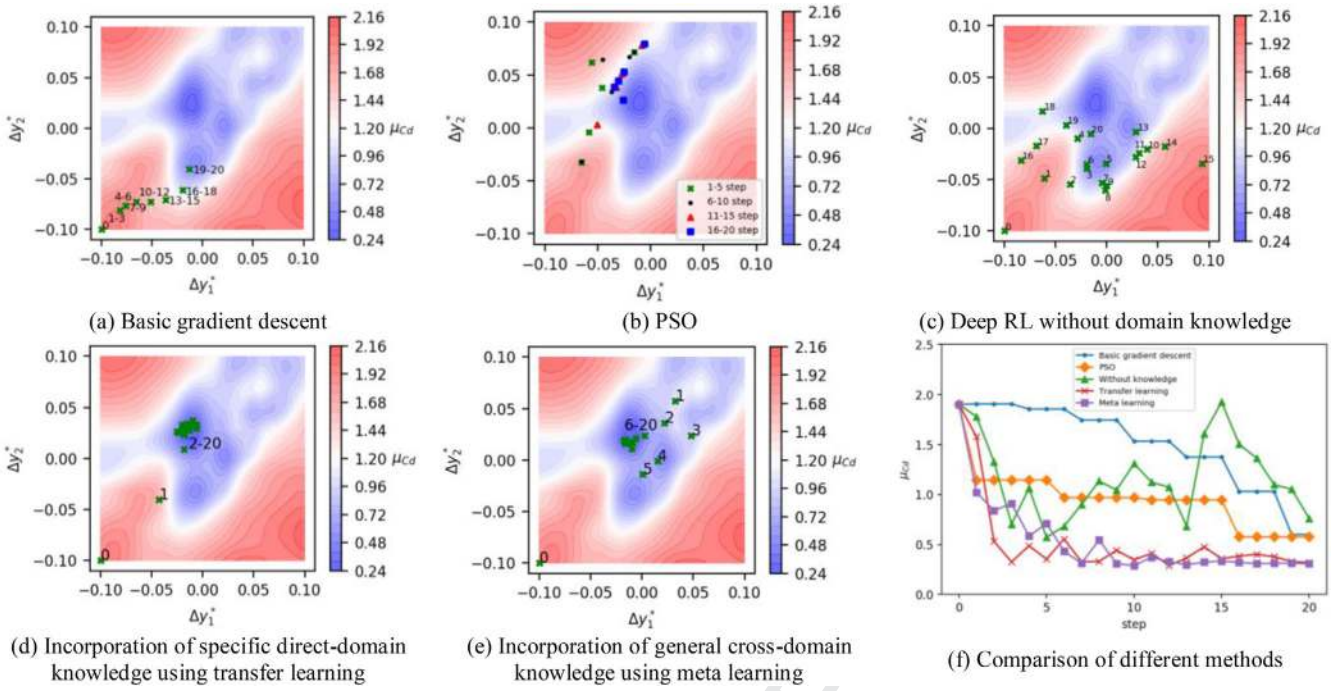

FIGURE 8 Optimization of μ_{Cd} using different schemes with the starting point (–0.1, –0.1)

corresponding effective policy are directly employed in the current aerodynamic shape optimization process without further training.

The proposed designs by the general cross-domain knowledge-enhanced deep RL-based shape optimizer in the current aerodynamic optimization process are shown in Figure 8e. It is shown that the developed optimizer first explores in design space for a few steps and then efficiently march towards the global optimum. The comparison results in Figure 8f indicate that the general cross-domain knowledge from a mixture of Gaussian process functions can greatly facilitate the efficient search of deep RL-based optimizer for the optimal (or near optimal) design in LES-based simulations, which demonstrate that the general cross-domain knowledge-enhanced deep RL-based optimizer outperforms both gradient-based and gradient-free optimization schemes in this case study. In addition to the starting point (–0.1, –0.1), the comparative study with another starting point (–0.1, 0.1) is further carried out to more comprehensively demonstrate the computational advantage of the proposed aerodynamic shape optimizer. The simulation results are given in Figure 9, and it is shown that both specific direct-domain and general cross-domain knowledge-enhanced deep RL-based optimizers still present improved performance compared to conventional methods in the case of a different starting point. Other starting points (e.g., points of (0.1, 0.1) and (0.1, –0.1)) are also investigated, and it is found that the knowledge-enhanced deep RL-based optimizer can always quickly reach to optimum. It is noted that the performance

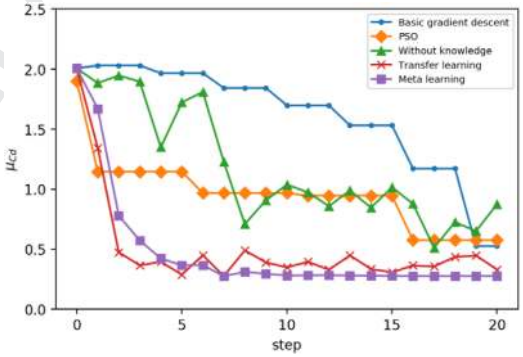

FIGURE 9 Comparison of various methods with the starting point (–0.1, 0.1)

of each optimization scheme used in this numerical example may vary with model parameters, wind conditions (e.g., angles of attack), and other factors. Hence, a comprehensive parametric study is needed before a more general conclusion in terms of optimization efficiency can be obtained.

5 | CONCLUSION

This study developed a novel aerodynamic shape optimizer for wind-sensitive structures using knowledge-enhanced deep RL. The RL approach with a DNN-based policy (for shape search and update) is utilized as a data-driven shape optimization scheme for aerodynamic mitigation of wind-sensitive structures, and the equation-free domain

knowledge is leveraged to remarkably enhance the training efficiency. It is shown that the specific direct-domain knowledge learnt from low-fidelity computational fluid dynamics simulations (with RANS equations) and general cross-domain knowledge learned from a mixture of Gaussian process functions can be incorporated into the deep RL-based aerodynamic shape optimizer via the transfer-learning and meta-learning techniques, respectively, and both greatly facilitated the efficient search of an effective RL policy to obtain the optimal (or near optimal) design in expensive high-fidelity CFD simulations (i.e., large-eddy simulations). Numerical examples for aerodynamic shape optimization of the cross section of a typical tall building demonstrated the improved performance of both specific direct-domain and general cross-domain knowledge-enhanced deep RL-based shape optimizers compared to conventional gradient-based and gradient-free optimization algorithms. To simultaneously incorporate both specific direct-domain and general cross-domain knowledge into the same deep RL-based shape optimization for a further improved performance would be an interesting research topic to explore. Other advanced deep learning schemes (e.g., enhanced probabilistic neural network, Ahmadlou & Adeli, 2010; neural dynamic classification, Rafiei & Adeli, 2017b; dynamic ensemble learning, Alam, Siddique, & Adeli, 2020; and finite element machine, Pereira, Piteri, Souza, Papa, & Adeli, 2020) should be also explored to enhance simulation accuracy and efficiency of the aerodynamic shape optimization. Another direction of future work is to extend current simple 2D aerodynamic shape optimization to more complex nonlinear, high-dimensional and nonconvex problems for aerodynamic mitigation of large-scale wind-sensitive structures.

ACKNOWLEDGMENTS

The support from Institute of Bridge Engineering at University at Buffalo is gratefully acknowledged. The authors are also thankful to Ms. Fei Ding from University of Notre Dame for the valuable assistance on CFD simulations.

FUNDING INFORMATION

Institute of Bridge Engineering at University at Buffalo

REFERENCES

Ahmadlou, M., & Adeli, H. (2010). Enhanced probabilistic neural network with local decision circles: A robust classifier. *Integrated Computer-Aided Engineering*, *17*(3), 197–210.

Alam, K. M. R., Siddique, N., & Adeli, H. (2020). A dynamic ensemble learning algorithm for neural networks. *Neural Computing and Applications*, *32*(12), 8675–8690.

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., … De Freitas, N. (2016). Learning to learn by gradient descent by gradient descent. Paper presented at the Proceedings of 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

Benito-Picazo, J., Domínguez, E., Palomo, E. J., & López-Rubio, E. (2020). Deep learning-based video surveillance system managed by low cost hardware and panoramic cameras. *Integrated Computer-Aided Engineering*, *27*(4), 373–387.

Bernardini, E., Spence, S. M., Wei, D., & Kareem, A. (2015). Aerodynamic shape optimization of civil structures: A CFD-enabled Kriging-based approach. *Journal of Wind Engineering and Industrial Aerodynamics*, *144*, 154–164.

Chen, Y., Hoffman, M. W., Colmenarejo, S. G., Denil, M., Lillicrap, T. P., Botvinick, M., & de Freitas, N. (2017). Learning to learn without gradient descent by gradient descent. Paper presented at the Proceedings of the 34th International Conference on Machine Learning(ICML 2017), Sydney, Australia: .

Davenport, A. G. (1962). Buffeting of a suspension bridge by storm winds. *Journal of the Structural Division*, *88*(3), 233–268.

Davenport, A. G. (1971). The response of six building shapes to turbulent wind. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, *269*(1199), 385–394.

Ding, F., & Kareem, A. (2018). A multi-fidelity shape optimization via surrogate modeling for civil structures. *Journal of Wind Engineering and Industrial Aerodynamics*, *178*, 49–56.

Eaton, E., & Lane, T. (2008). Modeling transfer relationships between learning tasks for improved inductive transfer. In W. Daelemans et al. (Eds.), *Joint european conference on machine learning and knowledge discovery in databases* (pp. 317–332). LNAI 5211. Berlin, Germany: Springer.

Kennedy, J., &. Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of the IEEE international conference on neural networks Perth, Australia* (Vol. *4*, pp. 1942–1948). Piscataway, NJ: IEEE Press.

Elshaer, A., & Bitsuamlak, G. (2018). Multiobjective aerodynamic optimization of tall building openings for wind-induced load reduction. *Journal of Structural Engineering*, *144*(10), 04018198.

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of 34th International Conference on Machine Learning (ICML 2017), Sydney, Australia* (pp. 1126–1135). Proceedings of Machine Learning Research (Vol. *70*). Copenhagen, Denmark: MLR Press.

Hao, J., & Wu, T. (2018). Downburst-induced transient response of a long-span bridge: A CFD-CSD-based hybrid approach. *Journal of Wind Engineering and Industrial Aerodynamics*, *179*, 273–286.

Hou, F., & Sarkar, P. P. (2018). A time-domain method for predicting wind-induced buffeting response of tall buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *182*, 61–71.

Kim, H., & Adeli, H. (2005). Wind-induced motion control of 76-story benchmark building using the hybrid damper-TLCD system. *Journal of Structural Engineering*, *131*(12), 1794–1802.

Kim, B. C., & Yhim, S. S. (2014). Buffeting analysis of a cable-stayed bridge using three-dimensional computational fluid dynamics. *Journal of Bridge Engineering*, *19*(11), 04014044.

Kociecki, M., & Adeli, H. (2014). Two-phase genetic algorithm for topology optimization of free-form steel space-frame roof structures with complex curvatures. *Engineering Applications of Artificial Intelligence*, *32*, 218–227.

Li, T., Wu, T., & Liu, Z. (2020). Nonlinear unsteady bridge aerodynamics: Reduced-order modeling based on deep LSTM networks.

Journal of Wind Engineering and Industrial Aerodynamics, *198*, 104116.

Liang, X. (2019). Image based post disaster inspection of reinforced concrete bridge systems using deep learning with Bayesian optimization. *Computer Aided Civil and Infrastructure Engineering*, *34*(5), 415–430.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., … Wierstra, D. (2016). Continuous control with deep reinforcement learning. Paper presented at Proceedings of 4th International Conference on Learning Representations (ICLR 2016), San Juan, Puerto Rico.

Min, A. T. W., Sagarna, R., Gupta, A., Ong, Y. S., & Goh, C. K. (2017). Knowledge transfer through machine learning in aircraft design. *IEEE Computational Intelligence Magazine*, *12*(4), 48–60.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., … Petersen, S. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.

Nagao, F., Utsunomiya, H., Oryu, T., & Manabe, S. (1993). Aerodynamic efficiency of triangular fairing on box girder bridge. *Journal of Wind Engineering and Industrial Aerodynamics*, *49*(1–3), 565–574.

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359.

Park, H. S., & Adeli, H. (1997). Data parallel neural dynamics model for integrated design of large steel structures, *Computer-Aided Civil and Infrastructure Engineering*, *12*(5), 311–326.

Peherstorfer, B., Willcox, K., & Gunzburger, M. (2018). Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review*, *60*(3), 550–591.

Pereira, D. R., Piteri, M. A., Souza, A. N., Papa, J. P., & Adeli, H. (2020). FEMa: A finite element machine for fast learning. *Neural Computing and Applications*, *32*(10), 6393–6404.

Psichogios, D. C., & Ungar, L. H. (1992). A hybrid neural network-first principles approach to process modeling. *AIChE Journal*, *38*(10), 1499–1511.

Rafiei, M. H., Khushefati, W. H., Demirboga, R., & Adeli, H. (2017). Supervised deep restricted Boltzmann machine for estimation of concrete. *ACI Materials Journal*, *114*(2), 237–244.

Rafiei, M. H., & Adeli, H. (2017a). A novel machine learning-based algorithm to detect damage in high-rise building structures. *The Structural Design of Tall and Special Buildings*, *26*(18), e1400.

Rafiei, M. H., & Adeli, H. (2017b). A new neural dynamic classification algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(12), 3074–3083.

Rosenstein, M. T., Marx, Z., Kaelbling, L. P., & Dietterich, T. G. (2005). To transfer or not to transfer. Paper presented at NIPS 2005 workshop on transfer learning, British Columbia, Canada.

Scanlan, R. H. (1978). The action of flexible bridges under wind, I: Flutter theory. *Journal of Sound and Vibration*, *60*(2), 187–199.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., … Chen, Y. (2017). Mastering the game of go without human knowledge. *Nature*, *550*(7676), 354–359.

Simões, D., Lau, N., & Reis, L. P. (2020). Exploring communication protocols and centralized critics in multi-agent deep learning. *Integrated Computer-Aided Engineering*, *27*(4), 333–351.

Skinner, S. N., & Zare-Behtash, H. (2018). State-of-the-art in aerodynamic shape optimisation methods. *Applied Soft Computing*, *62*, 933–962.

Snaiki, R., & Wu, T. (2019). Knowledge-enhanced deep learning for simulation of tropical cyclone boundary-layer winds. *Journal of Wind Engineering and Industrial Aerodynamics*, *194*, 103983.

Sørensen, R. A., Nielsen, M., & Karstoft, H. (2020). Routing in congested baggage handling systems using deep reinforcement learning. *Integrated Computer-Aided Engineering*, *27*(2), 139–152.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.

Tanaka, H., Tamura, Y., Ohtake, K., Nakai, M., & Kim, Y. C. (2012). Experimental investigation of aerodynamic forces and wind pressures acting on tall buildings with various unconventional configurations. *Journal of Wind Engineering and Industrial Aerodynamics*, *107*, 179–191.

Tominaga, Y., Mochida, A., Yoshie, R., Kataoka, H., Nozu, T., Yoshikawa, M., & Shirasawa, T. (2008). AIJ guidelines for practical applications of CFD to pedestrian wind environment around buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *96*(10-11), 1749–1761.

Topping, B. H. V. (1983). Shape optimization of skeletal structures: A review. *Journal of Structural Engineering*, *109*(8), 1933–1951.

Wang, N., & Adeli, H. (2015). Robust vibration control of wind-excited highrise building structures. *Journal of Civil Engineering and Management*, *21*(8), 967–976.

Wang, H., & Wu, T. (2020). Knowledge-enhanced deep learning for wind-induced nonlinear structural dynamic analysis. *Journal of Structural Engineering*, *146*(11), 04020235.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67–82.

Wu, T., & Kareem, A. (2011). Modeling hysteretic nonlinear behavior of bridge aerodynamics via cellular automata nested neural network. *Journal of Wind Engineering and Industrial Aerodynamics*, *99*(4), 378–388.

Yan, X., Zhu, J., Kuang, M., & Wang, X. (2019). Aerodynamic shape optimization using a novel optimizer based on machine learning techniques. *Aerospace Science and Technology*, *86*, 826–835.

Yang, Y., Wu, T., Ge, Y., & Kareem, A. (2015). Aerodynamic stabilization mechanism of a twin box girder with various slot widths. *Journal of Bridge Engineering*, *20*(3), 04014067.

Yazdi, J., & Neyshabouri, S. S. (2014). Adaptive surrogate modeling for optimization of flood control detention dams. *Environmental Modelling & Software*, *61*, 106–120.

Zhou, Z., Li, X., & Zare, R. N. (2017). Optimizing chemical reactions with deep reinforcement learning. *ACS Central Science*, *3*(12), 1337–1344.

How to cite this article: Li S, Snaiki R, Wu T. A knowledge-enhanced deep reinforcement learning-based shape optimizer for aerodynamic mitigation of wind-sensitive structures. *Comput Aided Civ Inf*. 2020;1–14.
https://doi.org/10.1111/mice.12655