Single-step deep reinforcement learning for two- and three-dimensional optimal shape design

H. Ghraieb, J. Viquerat, A. Larcher, P. Meliga, E. Hachem

Single-step deep reinforcement learning for two- and three-dimensional optimal shape design

H. Ghraieb[a], J. Viquerat[a], A. Larcher[a], P. Meliga[a], E. Hachem[a,*]

[a]*Mines Paris, PSL Research University, Centre de mise en forme des matériaux (CEMEF), CNRS UMR 7635, 06904 Sophia Antipolis Cedex, France*

Abstract

This research gauges the capabilities of deep reinforcement learning (DRL) techniques for direct optimal shape design in computational fluid dynamics (CFD) systems. It uses Policy Based Optimization, a single-step DRL algorithm intended for situations where the optimal policy to be learnt by a neural network does not depend on state. The numerical reward fed to the neural network is computed with an in-house stabilized finite elements environment combining variational multiscale (VMS) modeling of the governing equations, immerse volume method, and multi-component anisotropic mesh adaptation. Several cases are tackled in two and three dimensions, for which shapes with fixed camber line, angle of attack and cross-sectional area are generated by varying a chord length and a symmetric thickness distribution (and possibly extruding in the off-body direction). At zero incidence, the proposed DRL-CFD framework successfully reduces the drag of the equivalent cylinder (i.e., the cylinder of same cross-sectional area) by 48% at a Reynolds numbers in the range of a few hundreds. At an incidence of 30°, it increases the lift to drag ratio of the equivalent ellipse by 13% in two dimensions and 5% in three dimensions at a chord Reynolds numbers in the range of a few thousands. Although the low number of degrees of freedom inevitably constrains the range of attainable shapes, the optimal is systematically found to perform just as well as a conventional airfoil, despite DRL starting from the ground up and having no priori knowledge of aerodynamic concepts. Such results showcase the potential of the method for black-box shape optimization of practically meaningful CFD systems. Since the resolution process is agnostic to details of the underlying fluid dynamics, they also pave the way for a general evolution of reference shape optimization strategies for fluid mechanics and any other domain where a relevant reward function can be defined.

Keywords: Deep Reinforcement Learning; Artificial Neural Networks; Shape optimization; Computational fluid dynamics; Policy Based Optimization.

1. Introduction

Shape optimization is ubiquitous in engineering applications ranging from magnetostatics [1], acoustics [2], image restoration and segmentation [3], composite material identification [4] to nano-optics [5], just to name a few. Shape optimization in fluid mechanics dates back to the pioneering work of Pironneau on the minimization of energy loss in Stokes and Navier–Stokes flows [6, 7]. Since then, it has become an increasingly important research topic in the attempt to enhance drag reduction capabilities, which is due to the ever growing concerns on aerodynamic energy efficiency (to give a taste, reducing the overall drag by just a few percent while maintaining lift can help reducing fossil fuel consumption and CO2 emission while saving several billion dollars annually in ocean shipping or airline traffic [8]). In the following, the focus is essentially on airfoil shape optimization, a key component of aircraft flight mechanics that has come into prominence in a variety of other applications such as acoustic noise reduction [9] or energy harvesting [10]. One of the major challenges in the field is that the majority of flows of engineering interest are time-dependent

*Corresponding author
Email address: elie.hachem[@]mines-paristech.fr (E. Hachem)

and even turbulent (e.g., fluttering, buffeting, dynamic stall), and therefore require sophisticated unsteady methods and optimization techniques, thus drastically increasing the computational cost.

Shape optimization has historically been tackled by two main classes of approaches, namely gradient-based and gradient-free methods. Gradient-based methods rely on the evaluation of the gradient of the objective function with respect to the design parameters. They have proven effective in large optimization spaces when said gradient is computed by the adjoint method [11–13], whose cost is comparable to that of solving the governing equation (unlike more computationally expensive alternatives such as variance-based and regression-based methods, in which the governing equations need to be solved repeatedly, up to a hundred times). Nonetheless, gradient-based algorithms are easily trapped in local optima, meaning that the solution optimality can be very sensitive to the initial guess, all the more so when applied to stiff nonlinear problems [14]. Gradient-free methods are better equipped in this regard, but can be more complex to implement and to use. Among the available methods, genetic algorithms [15], particle swarm optimization [16] or metropolis algorithms [17] feature good global optimization capabilities, but they can be highly sensitive to heuristically chosen meta-parameters, plus their cost is usually higher and can easily exceed the available computational budget, thus limiting the number of design parameters [18]. It should be noted that both classes of methods can make use of cheap-to-evaluate surrogate models to approximate expensive objective and constraint functions without resorting systematically to numerical simulations [19]. Several approaches exist for building such surrogate models, e.g., polynomial response surfaces, radial basis functions, kriging, or supervised artificial neural networks [20], for which geometric parametrization plays a determinant role, in terms of both the attainable geometries and the tractability of the optimization process [21].

The premise of this research is that the related task of selecting an optimal subset of design parameters can alternatively be assisted using deep reinforcement learning (DRL). DRL is the advanced branch of machine learning that couples deep neural networks (DNNs, a family of versatile tools that can learn how to hierarchically extract informative features from data, and have gained traction as efficient computational processors for performing a variety of tasks, from exploratory data analysis to qualitative and quantitative predictive modeling) and reinforcement learning, a class of decision-making algorithms that can autonomously learn effective policies for sequential decision problems. In practice, DRL involves DNNs learning how to behave in an environment so as to maximize some notion of long-term reward, a task compounded by the fact that each action taken affects both immediate and future rewards. The feature extraction capabilities of DNNs, as well as their ability to handle quasi-arbitrary nonlinear input/output mappings, have lifted several major obstacles that hindered classical reinforcement learning and has led unprecedented efficiency in the context of nonlinear optimal control problems with high-dimensional state spaces. Several notable works using DRL in mastering games (e.g., Go, Poker) have stood out for attaining superhuman level [22, 23], but the approach has also breakthrough potential for practical applications such as robotics [24, 25], computer vision [26], finance [27], autonomous cars [28, 29], or data center cooling [30].

The efforts for applying DRL to fluid mechanics are ongoing but still at an early stage, as recently reviewed in [31]. Nonetheless, the domain has undergone a large inflow of contributions with clear focus on drag reduction problems [32–44]. This enthusiasm is likely due to the increasing number of open-source initiatives [32, 45, 46], that has led to an accelerated diffusion of the methods in the community, and to the sustained commitment from the machine learning community, that has allowed concurrently expanding the scope from computationally inexpensive, low-dimensional reductions of the underlying fluid dynamics to complex Navier–Stokes systems [47, 48], all the way to experimental set-ups [49]. A handful of studies have recently provided insight into the performance improvements to be delivered in shape optimization, but it is worth emphasizing that figuring out a fixed shape that best meets a set of required criteria (e.g., high lift-to-drag ratio, low pressure loss) requires optimizing state-independent parameters, which is not *per se* the original purpose of DRL. Nonetheless, two main classes of methods have emerged in the community, namely the direct and incremental approaches. The incremental approach uses the state-to-action mapping as a way to incrementally modify an initial shape into an optimal one [50–53], which exploits the capabilities of the DRL paradigm (in which network updates are performed after multi-step episodes) in performing active flow control. The direct approach [46] conversely relies on single-step DRL, a subset of DRL in which network updates are performed after one-step episodes (hence the

stateless moniker), and builds on recent efforts to assess the relevance of DRL in the context of open-loop control [41, 54].

This research introduces a novel framework combining single-step reinforcement learning with immersed methods for fluid flow shape optimization, that exploits both the ability of neural networks to learn to approximate arbitrarily well the mapping function between input and output spaces, and the dynamic programming built in the reinforcement learning algorithm. It is a follow-up on to our contribution showcasing the first application of DRL to direct shape optimization [46]. It uses Policy Based Optimization (PBO [55]), a novel single-step algorithm developed in-house, that improves the convergence rate of the previously used single-step Proximal Policy Optimization (PPO [24]) algorithm by adopting several key heuristics from the covariance matrix adaptation evolution strategy (CMA-ES). In short, PBO learns the mean, variance and correlation parameters of a multivariate normal search distribution from three separate neural networks, while single-step PPO updates the mean and variance (the same for all variables) from a single network, which can prematurely shrink the exploration variance. The objective is twofold: first, to further shape the capabilities of PBO for fluid mechanics applications (as it has so far been limited to textbook problems of analytic functions minimization), to help narrow the gap between DRL and advanced numerical methods for multi-scale, multi-physics computational fluid dynamics (CFD). Second, to gauge the feasibility of learning optimal designs from a low, yet suitable number of design parameters, for which Bézier curves, B-splines and NURBS are good candidates. We believe this is chief to mitigate the computational burden without deteriorating the geometric accuracy, since the parametrization in the direct approach provides a complete description of the shape itself, not that of a perturbation to a reference shape. The PBO agent is trained on high-fidelity CFD simulations, in contrast to most aforementioned studies about incremental shape optimization, in which a pre-trained surrogate or a simplified model is used for full agent training, or to perform an initial learning phase before re-training on a CFD environment using transfer learning. This is because the uncertainty of surrogate models cannot be quantified during optimization, which may misguide policy updating. We insist that it lies out of the scope of this paper to provide exhaustive performance comparison data against state-of-the art optimization techniques (e.g., evolution strategies or genetic algorithms). This would indeed require a tremendous amount of time and resources even though the efforts for developing the method remain at an early stage. Nonetheless, it is worth mentioning that PBO is shown in [55] to compare well against standard CMA-ES and to significantly outperform our previous PPO-based single-step algorithm, even though new algorithms cannot be expected to reach right away the level of performance of their more established counterparts.

The organization is as follows: section 2 introduces PBO (together with the baseline principles of DRL and single-step DRL), and outlines the main features of the finite element CFD environment used to compute the numerical reward fed to the neural networks. Section 3 revisits the classical problem of finding the two-dimensional shapes minimizing drag in a uniform flow for the purpose of validation and assessment part of the method capabilities. In section 4, PBO is applied to more meaningful aerodynamic optimization problems consisting of finding the two-dimensional shapes maximizing the lift to drag ratio in the context of turbulent flows at moderately large Reynolds number (in the range of a few thousands). Finally, an extension to three-dimensional shapes is proposed in section 5.

2. Methodology

2.1. Deep reinforcement learning

Reinforcement learning (RL) is a process by which an agent learns to earn rewards through trial-and-error interaction with its environment. At each turn, the agent observes the state s_t of the environment and takes an action a_t, that prompts both the transition to the next state s_{t+1} and the reward received r_t. This repeats until some termination state is reached, the core objective of the agent being to learn the succession of actions maximizing its cumulative reward over an episode (this is the reference unit for agent update, best understood as one instance of the scenario in which it takes actions). In a deep reinforcement learning context (deep RL or DRL), the agent is a deep neural network (DNN) patterned after the neural circuits formed by neurons in human brains.

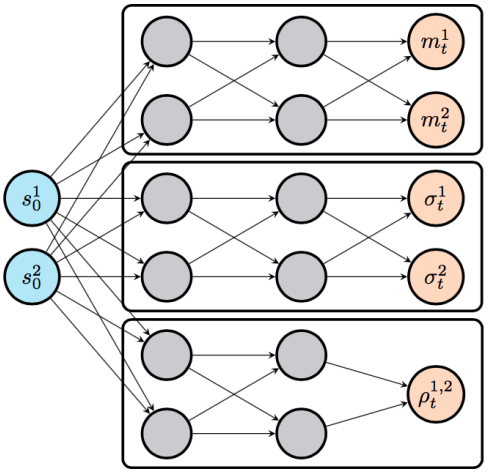

Figure 1: Policy networks used in PBO to map states to policy. Three networks trained separately are used for the prediction of mean, standard deviation, and correlation parameters. Orthogonal weights initialization is used throughout the networks, with a unit gain for all layers except the output layers, for which the gain is set to 10^{-2}.

The most general form of neural network architecture is the fully connected DNN, in which the processing units (the artificial neurons) are stacked in layers and information propagates forward from the input layer to the output layer via "hidden" layers. Each neuron performs a weighted sum of its inputs to assign significance with regard to the task the algorithm is trying to learn, adds a bias to figure out the part of the output independent of the input, and feeds an activation function that determines whether and to what extent the computed value should affect the outcome. The neural network learns to represent the relation between input (action) and output (reward) data by repeatedly adjusting the weights and biases by back-propagation, from the output layer back through the hidden layers to the input layer (a process known as training).

2.2. Single-step deep reinforcement learning

Single-step DRL is a subset of DRL that has recently emerged from the premise that tweaked versions of regular DRL algorithms can be used as black-box optimizers. The underlying idea is that it may be enough for the agent to interact only once per episode with its environment (hence, single-step episodes, and by extension, single-step DRL) if the optimal behavior to be learnt is independent of state, as is notably the case in optimization and open-loop control problems. The novelty of the approach can be summed up as follows: in DRL, a DNN learns the optimal set of observation-based actions $a^\star$ yielding the largest possible reward. In single-step DRL, it learns instead the optimal mapping $f_{\theta^\star}$ such that $a^\star = f_{\theta^\star}(s_0)$, where s_0 is some input state (usually a constant vector) repeatedly fed to the agent for the optimal policy to eventually embody the transformation from s_0 to $a^\star$. A direct consequence is that single-step DRL algorithms can use much smaller networks (compared to the usual agent architecture used in other DRL contributions), because the agent is not required to learn a complex state-action relation, but only a transformation from a constant input state to a given action.

2.3. Policy based optimization

The present research relies on policy-based optimization (PBO) a single-step, model free, off-policy gradient RL algorithm whose key features are summarized as follows:

- the agent interacts with the environment itself, not a surrogate model of the environment (model free, hence no assumptions about the fluid dynamics of the problems to be solved),
- its behavior is modeled after a parametrized probability distribution of actions $\pi_\theta(a)$, optimized by gradient ascent (policy gradient),
- the agent is not required to sample the training data with the current policy (off-policy),

PBO draws actions from a d-dimensional multivariate normal distribution (with d the dimension of the action required by the environment). A full co-variance matrix is used to improve the balance between exploration and exploitation (the single-step PPO algorithm used in [46] conversely assumes all variables to have the same variance and to be uncorrelated, which can prematurely shrink the exploration variance). The co-variance matrix also accelerates convergence to the optimum by aligning the contour of the sampling distribution with the contour lines of the objective function and thereby the direction of steepest ascent.

As shown in figure 1, three independent neural networks output the necessary mean, standard deviation, and correlation information, using hypersphere decomposition [56, 57] to generate valid symmetric, positive semidefinite covariance matrices. Different meta-parameters and architectures can be used for each network, which is shown in [55] to substantially impact the convergence rate. Actions drawn in $[-1;1]^d$ are then mapped into relevant physical ranges, a step deferred to the environment as being problem-specific. Finally, the Adam algorithm [58] runs stochastic gradient ascent by computing adaptive learning rates (i.e., the step sizes to be taken in the gradient direction) for each policy parameter, using the gradient of the loss function

$$L(\theta) = \mathop{\mathbb{E}}_{a\sim\pi_\theta}\Big[\max(\tilde{r},0)\log\pi_\theta(a)\Big]. \tag{1}$$

In 1, $\tilde{r}$ is the whitened reward normalized to zero mean and unit variance, considered a suitable advantage estimator. The rationale for this choice is as follows: as is customary in DRL, the discounted cumulative reward is approximated by the advantage function, that measures the improvement (if positive, otherwise the lack thereof) associated with taking action a in state s compared to taking the average over all possible actions. Because a single-step trajectory consists of a unique state-action pair, the discount factor adjusting the trade-off between immediate and future rewards can be set to unity, in which case the advantage reduces to the reward; see [41]. Substituting the whitened reward for r introduces bias but reduces variance, and thus the number of actions needed to estimate the expected value. Finally, the max allows discarding negative-advantage actions, that may destabilize learning when performing multiple mini-batch gradient steps using the same data (as each step drives the policy further away from the sampled actions).

2.4. Computational fluid dynamics environment

At the core of the CFD resolution framework is the in-house, CimLIB_CFD parallel finite element library [59], whose main ingredients are as follows:

- the variational multiscale approach (VMS) is used to solve a stabilized weak form of the governing equations using linear approximations ($\mathbb{P}_1$ elements) for all variables, which otherwise breaks the Babuska–Brezzi condition. The approach relies on an a priori decomposition of the solution into coarse and fine scale components [60–62]. Only the large scales are fully represented and resolved at the discrete level. The effect of the small scales is encompassed by consistently derived source terms proportional to the residual of the resolved scale solution, hence ad-hoc stabilization parameters comparable to local coefficients of proportionality.

- in laminar regimes, velocity and pressure come as solutions to the Navier–Stokes equations. In turbulent regimes, the focus is on phase-averaged velocity and pressure modeled after the unsteady Reynolds averaged Navier–Stokes (uRANS) equations. In order to avoid transient negative turbulent viscosities, negative Spalart–Allmaras [63] is used as turbulence model, whose stabilization proceeds from that of the convection-diffusion-reaction equation [64, 65].

- two-dimensional airfoil sections with fixed camber line are generated by varying a chord length and a thickness distribution. The chord direction is constant, just as the angle of attack measuring the incidence relative to the oncoming flow. The upper (suction/leeward) and lower (pressure/windward) sides are discretized into n_p control points equally spaced in the camber line direction. All shapes are closed and symmetrical with respect to the chord line, as achieved forcing zero thickness at the edges and identical (half)-thicknesses at each forward and rearward facing points. Consecutive points are connected by a cubic Bézier curve using local position and curvature

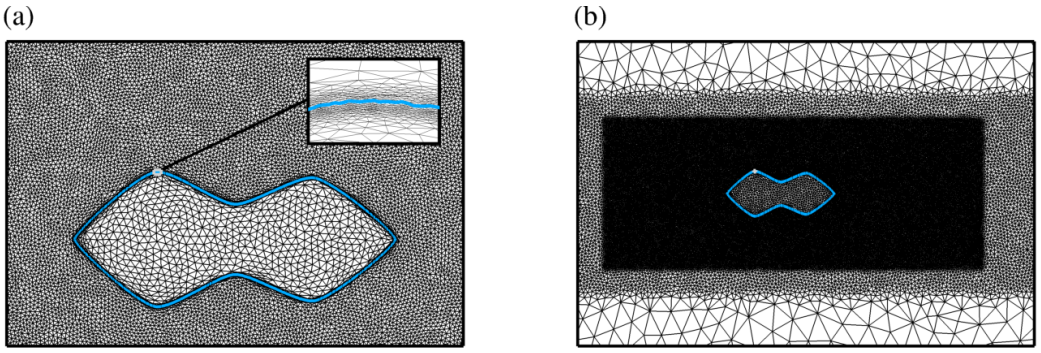

Figure 2: Details of (a) an anisotropic adapted mesh and (b) successive refinement steps of the background mesh. The blue line in (a) indicates the zero iso-contour of the level set function.

information. The final step consists of sampling all Bézier curves and in exporting a closed loop, to be either used as an immersed mesh in a two-dimensional (2-D) environment, or extruded in the off-body direction to serve as an immersed mesh in a three-dimensional (3-D) environment.

-. the immersed volume method (IVM) is used to immerse and represent all geometries inside a unique mesh. The approach combines level-set functions, using the zero-iso value of a signed distance function to localize the solid/fluid interface, and anisotropic mesh adaptation, to align the mesh element edges with the interface and refine the mesh interface under the constraint of a fixed, number of edges. This ensures that the quality of all actions taken over the course of a DRL optimization is equally assessed, even though the interface is action-dependent.

Substantial evidence of the flexibility, accuracy and reliability of the numerical framework for the intended ammplication is documented in several papers to which the reader is referred for exhaustive details regarding the shape generation using Bézier curves [46, 66], the level-set and mesh adaptation algorithms [67, 68], the VMS formulations, stabilization parameters and discretization schemes used in laminar and turbulent regimes [69–72], and the mathematical formulation of the IVM in the context of finite element VMS methods [73, 74].

2.5. Numerical implementation

At each episode, actions drawn from the current policy are distributed to n_{env} environments running in parallel, each of which executes a self-contained MPI-parallel numerical simulation (here, all simulations are performed on a few tens of cores on a workstation of Intel Xeon E5-2640 processors) and feeds the reward associated to its input action to the DRL algorithm. There are thus two levels of parallelism related to the environment and the computing architecture. This simple parallelization technique is key to use DRL in the context of CFD applications, as a large number of actions drawn from the current policy must be evaluated to accurately compute the expected value of the policy loss (1). Even though, the high CPU cost of performing massive, unsteady numerical simulations involving hundreds of thousands (even millions) of degrees of freedom caps the number of environments that can efficiently run in parallel, and thus the number of state-action-reward triplets that can be sampled from the current policy (which also makes intractable the common practice in DRL studies to gain insight into the performances of the selected algorithm by averaging results over multiple independent training runs with different random seeds, as it would trigger a prohibitively large computational burden. The same random seeds are thus used for all computations to ensure a minimal level of performance comparison between cases.) PBO therefore improves the reliability of the loss evaluation by incorporating the reward data available from several previous episodes, using an empirical decay parameter that exponentially decreases the advantage history (to give recent episodes more weight) while retaining a longer memory of the previous episodes as the problem dimensionality increases (in accordance with the idea that more state-action-triplets are then needed to build a coherent covariance matrix). The remainder of the practical implementation details are as follows:

Mean	Variance	Correlation	Neural network
5×10^{-3}	»	10^{-3}	Learning rate
128	8	»	Nb. epochs
1	8	16	Nb. learning episodes
1	4	8	Nb. mini-batches
[2,2,2]	»	»	Architecture

Table 1: Details of the PBO meta-parameters and network architectures. For the architectures, only the sizes of the hidden layers are provided.

-. the environment consists of CFD simulations of incompressible flows described in a Cartesian coordinate system with drag (resp. lift) positive in the $+x$ (resp. $+y$) direction. All equations are discretized on 2-D and 3-D rectangular grids whose side lengths documented in the coming sections have been checked to be large enough not to have a discernible influence on the results (with the exception of the 3-D case in section 5, for which we favor computing all numerical solutions at affordable CPU cost using a limited transverse dimension). Open flow conditions are used, that consist of a uniform inflow in the x direction, together with symmetric lateral, advective outflow and no-slip interface conditions. In turbulent regime, the ambient value of the Spalart–Allmaras variable is three times the molecular viscosity, as recommended to lead to immediate transition. Typical adapted meshes of the interface and wake regions are shown in figure 2, the latter also being accurately captured via successive refinement of the background elements.

-. optimal surface shapes subject to a target cross-sectional area S_{ref} are determined by maximizing a compound reward function

$$r = \overline{J} - \beta |S - S_{ref}| \,, \tag{2}$$

where J is the objective function associated to performance, S is the cross-sectional area (also abbreviated as CSA in the following) of the shape, the overline indicates time-averaging, and β is a weighting coefficient that increasingly penalizes the shape when its area strays away from the target value. In practice, the cross-sectional area is computed as

$$S = \frac{1}{L} \int_\Omega H_\epsilon(\phi) \, \mathrm{d}\Omega \tag{3}$$

where H_ϵ is the smoothed Heaviside function introduced in [75], and L is the extrusion length in the off-body direction (hence equal to unity in 2-D). Moving average rewards and actions are also computed as the sliding average over the 50 latest values (or the whole sample if it has insufficient size). Time averages are performed over an interval $[t_i; t_f]$ with edges large enough to dismiss the initial transient and achieve convergence to statistical equilibrium. In the following, we take J to be a function of the drag and lift coefficients per unit span in the transverse direction, denoted by D and L, respectively, whose instantaneous values are computed with a variational approach featuring only volume integral terms, reportedly less sensitive to the approximation of the body interface than their surface counterparts [76, 77].

-. the agent consists of three identical fully connected networks with 3 hidden layers, each of which holds 2 neurons (this is by design, as we recall that the PBO networks can theoretically use different architectures). The only difference lies in the activation function applied to the output layer, namely the first network uses hyperbolic tangent to output the mean of the d-dimensional multivariate normal distribution in $[-1; 1]^d$, the second network uses sigmoid to output the standard deviations in $[0; 1]^d$, and the third network also uses sigmoid to output a set of coefficients in $[0, 1]^d$, eventually assembled into a full correlation matrix by hypersphere decomposition [56, 57]. As to the meta-parameters, the number of parallel environments used to collect rewards before performing the network updates is set from the well-established heuristics of CMA-ES (that similarly relies on

full co-variance matrices and uses an evolution path to add information about correlations across consecutive generations [78] to

$$n_{env} = \lfloor 4 + 3 \ln d \rfloor \,, \tag{4}$$

where $\lfloor \cdot \rfloor$ denotes the floor function. Each network is updated for n_e epochs (the number of full passes of the algorithm over the entire data set) using a learning rate λ (the size of the step taken in the gradient direction for policy update) and a history of n_{ep} episodes, shuffled and organized in n_b mini-batches (whose sizes are in multiples of n_{env}). The values used in this study are documented in table 1 to ease reproducibility.

3. Validation

3.1. Test case description

We assess first the relevance of the proposed numerical framework by revisiting the classical problem of finding the 2-D shape minimizing the drag force induced by a surrounding uniform flow at zero incidence. A sketch of the configuration is provided in figure 3. The origin of the coordinate system is at the half chord length. Several laminar cases at Reynolds number $\mathrm{Re} = U_\infty \sqrt{S_{ref}}/\nu$ are modeled after the Navier–Stokes equations, where U_∞ is the inflow velocity, ν the kinematic viscosity, and we have used the square root of the target cross-sectional area (set to $S_{ref} = 1$ in our implementation) as reference length. The objective function is simply

$$J = -D \,, \tag{5}$$

and the weighing coefficient is set empirically to $\beta = 8$. All CFD environments use the simulation parameters documented in table 2, found to offer a good compromise between numerical accuracy and computational effort since numerical tests carried out at two other grid resolutions and spatial extents yield limited variations within $2\% - 3\%$.

The control points used to parametrize the shape are labeled clockwise from 0 at the leading edge. All inner (i.e., non-end) curvature radii are set to 0.4 to provide sufficient smoothness (as this is a tad below the value 0.5 required for maximal smoothness). This leaves $n_p + 1$ independent design variables, the chord length c, two end curvature radii $\rho_{j\in\{0,n_p-1\}}$ and $n_p - 2$ inner thicknesses $e_{k\in\{1,\ldots,n_p-2\}}$. The network action output consists accordingly of values $(\hat{c}, \hat{\rho}_j, \hat{e}_k)$ in $[-1;1]^{n_p+1}$, mapped into the actual physical quantities using

$$c = \frac{1-\hat{c}}{2} c_{min} + \frac{1+\hat{c}}{2} c_{max} \,, \qquad \rho_j = \frac{1-\hat{\rho}_j}{2} \rho_{min} + \frac{1+\hat{\rho}_j}{2} \rho_{max} \,, \qquad e_k = e_{k,max} - \frac{1-\hat{e}}{2} \delta e \,, \tag{6}$$

for the chord to vary in $[c_{min}; c_{max}]$ with $c_{min} = 1$ and $c_{max} = 4$, the curvature radii to vary in $[\rho_{min}; \rho_{max}]$ with $\rho_{min} = 0.1$ and $\rho_{max} = 0.4$, and the thickness to vary in $[e_{k,max} - \delta e; e_{k,max}]$ with $\delta e = 0.4$ and $e_{k,max}$ a maximum value tuned locally for each problem. At each episode, the position of the inner points is adjusted to the current chord length to maintain equal spacing. Unless specified otherwise, all results documented hereafter are for $n_p = 5$, for which DRL evolves six design parameters, the chord length, two end curvature radii and three inner thicknesses.

3.2. Results

Several Reynolds numbers have been considered up to $\mathrm{Re} = 100$, for which random shapes collected over the course of the optimization, are presented in figures 4-7, together with their respective iso-contours of vorticity. Because the aspect ratio (as defined from the ratio of the maximum thickness to the chord length) barely exceeds unity, all solutions at $\mathrm{Re} \lesssim 50$ relax to steady state regardless of the DRL action (hence we do not report a proper averaging span for these cases in table 2, as we simply evaluate reward at a final time chosen large enough to flush out the transient behavior). Meanwhile, a small number of shapes with aspect ratio close to unity have been found to exhibit vortex shedding at $\mathrm{Re} = 100$, for which we pay attention to performing the necessary time averages. Figures 4-7 also provide exhaustive convergence history for the reward, the objective function, the ratio of actual to target CSA and the design parameters. The moving average reward especially decreases almost monotonically and reaches a plateau after a few ten

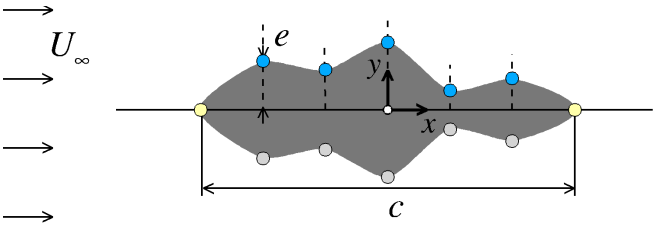

Figure 3: Schematic diagram of the minimum drag test case. The DRL agent optimizes the chord length, the curvature radius at the edge control points marked in yellow, and the thickness at the inner control points marked in blue. The thickness at the inner control points marked in grey deduces by symmetry.

episodes. At this point, the optimal CSA exhibits near-perfect agreement with the target value, hence evidencing the relevance of the reward penalty approach.

At Re = 1, the minimum drag body in figure 4 is that of a perfectly front-rear symmetric rugby ball, with chord length $1.95 \pm 0.6\%$ and aspect ratio $0.369 \pm 1.1\%$. These values have been obtained by averaging over the 10 latest episodes (with associated variance interval computed from the root-mean-square over the same interval, a simple yet robust criterion that will be used systematically to assess convergence for all cases reported in the following). They are close to the creeping flow optimal, whose chord length (relative to a unit target surface) and aspect ratio derived analytically in [80] are 1.88 and 0.40, respectively. The only noticeable difference lies in the fact that the DRL optimal has a pointed rear end with wedge angle about 90°, and a slightly more rounded front end with wedge angle ~ 120°, while the creeping flow optimal has two pointed ends with wedge angle about 100°. As the Reynolds number increases, the optimal chord length increases but the thickness decreases, hence the aspect ratio of the optimal body decreases (likely because the increasing adverse pressure gradient at the front needs to be counterbalance to avoid flow separation). At Re = 20, the optimal shape in figure 5 has chord length $2.40 \pm 0.8\%$ and aspect ratio $0.228 \pm 1.5\%$, and remains almost front-rear symmetric, although the rear section is slightly more streamlined. Similar results are obtained at Re = 50 (figure 6), with chord length $2.65 \pm 0.7\%$ and aspect ratio $0.204 \pm 1.0\%$. At Re = 100, the front-rear symmetry is lost as we obtain a streamlined shape with chord length $2.46 \pm 0.4\%$ and aspect ratio $0.235 \pm 0.8\%$; see figure 7. At Re = 1, the optimal drag ($13.10 \pm 0.02\%$) cuts down that of the equivalent cylinder (i.e., the cylinder of diameter $2\sqrt{S_{ref}/\pi}$, for the area to be equal to S_{ref}) by 6%, which is small but simply reflects that the ratio of drags on any two bodies tends to 1 in the limit where the Reynolds number tends to 0. In comparison, the achieved reduction is by 22% at Re = 20 (optimal drag $1.83 \pm 0.04\%$), 24% at Re = 50 ($1.10 \pm 0.05\%$), and 48% at Re = 100 ($0.71 \pm 4\%$).

Other than that, it is difficult to accurately validate the results because, although the search for optimal profiles of minimum drag in Navier–Stokes flows having received substantial interest in the literature, there is a wide variability in the problem formulation, especially in terms of design constraints (some authors specify a target surface, others impose only a lower bound, plus the values can vary from one reference to another), and the exact geometrical properties of the optimal (e.g., length, aspect ratio) are rarely documented. The closest study to our work is from Kondoh *et al.* [79], who tackle similar drag minimization problems via topology optimization, using a body force to model the effect of classical no-slip boundary conditions at the fluid/solid interface. It has not been possible to assess the convergence rate of DRL in the absence of any reference information in this regard, and it is not entirely clear whether the exact same optimization problem is solved due to inconsistent statements in the study regarding the nature of the design constraint, but even so, the reported optimal shapes and drags turn to be in good agreement with the present DRL results. One minor difference is that the shapes look pointed in [79] (but the blending of the interface makes it difficut to see in the original images), while the present ones are generally rounded at both ends, with little to no effect on the reward. On this point, we note that the end radii can vary substantially even after the reward has converged (as is the case for instance in figure 6(d) at Re = 50), which evidences a general lack of sensitivity to these specific design parameters). At Re = 1, the optimal drags differ by approximately 7%, which may seem large at

				Case setup
1	20	50	100	Reynolds number
5	»	»	»	Nb. points
6	»	»	»	Nb. design variables
				CFD
2	»	»	»	Dimensionality
0.2	»	0.125	»	Time-step
[50;50]	»	»	»	Averaging time span
8	»	»	»	Penalty coeff.
$[-10;20]\times[-10;10]$	»	»	»	Mesh dimensions
100000	»	110000	»	Nb. mesh elements
0.0005	»	»	»	Interface $\perp$ mesh size
				PBO
100	120	115	»	Nb. episodes
10	12	11	»	Nb. environments
3mn	3mn	5mn	10mn	CPU time†‡
5h	6h	10h	18h	Resolution time‡
				Parameter ranges
[1;4]	»	»	»	Chord length
[0.1;0.4]	»	»	»	LE curv. radius
[0.072;0.472]	»	»	»	↓ Thickness
[0.152;0.552]	»	»	»	↓ Thickness
[0.072;0.472]	»	»	»	↓ Thickness
[0.1;0.4]	»	»	»	TE curv. radius
				Optimal
1.95	2.41	2.64	2.46	Chord length
0.309	0.300	0.186	0.344	LE curv. radius
0.297	0.246	0.223	0.290	↓ Thickness
0.362	0.273	0.270	0.267	↓ Thickness
0.299	0.227	0.199	0.166	↓ Thickness
0.115	0.366	0.258	0.395	TE curv. radius
1.000	1.000	1.000	1.001	Ratio of actual to target CSA
13.1	1.83	1.10	0.71	Drag (present)
12.10	1.81	1.10	0.76	Drag [79]

Table 2: Case setup, simulation parameters and convergence data for the drag minimization problem, as computed by averaging over the 10 latest learning episodes. Leading-edge (front end) and trailing edge (rear end) data are labeled LE and TE, respectively. † All CPU times provided per episode and per environment. ‡ All values obtained averaging over 5 independent runs using 12 cores.

first sight but is actually fair given the high sensitivity of drag to small changes in the Reynolds number in this regime. The drags and chord lengths are nearly identical at Re = 20 and 50, as we find the ratio of the chord length at the current Reynolds number to its Re = 1 counterpart to be 1.24 at Re = 20 and 1.35 at Re = 50 using DRL, while extracting data from the reference figures using a graph digitizer software yields values of 1.26 at Re = 20 and 1.33 at Re = 50 (it has not been possible to similarly extract the aspect ratio due to blurred and/or mixed pixels). At Re = 100, the shapes somewhat differ as the optimal in [79] is more elongated and less streamlined in the rear section. Meanwhile the optimal drags differ by only 6%, which raises the possibility that the objective function has either a unique flat minimum, or several nearly equivalent minima. Figure 7 constitutes a favorable presumption in this regard, as the objective function exhibits surprisingly low variations over the course of optimization, and most shapes in figure 7(b) actually are within

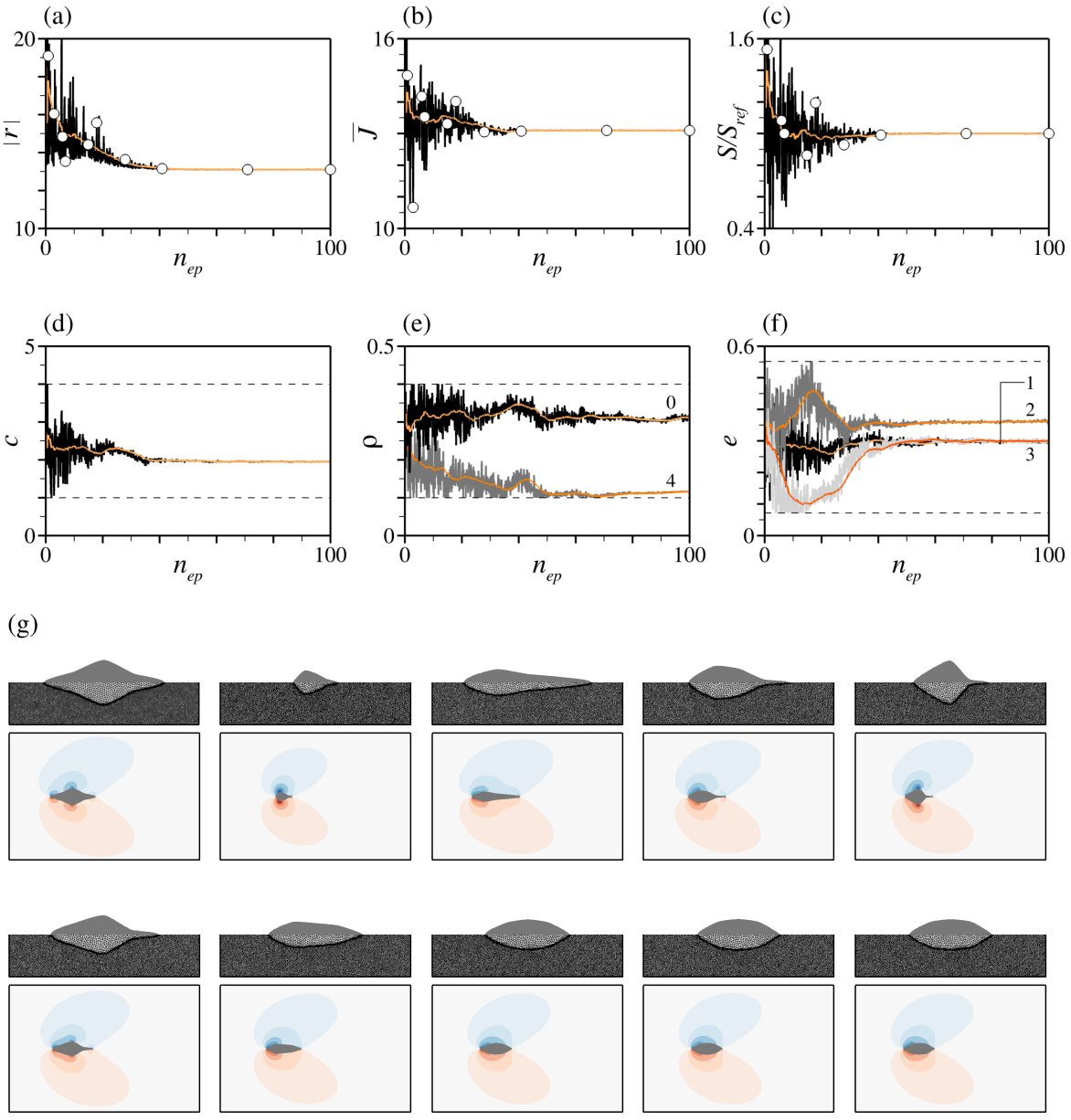

Figure 4: Maximum lift to drag ratio test case at Re = 1 under constant area constraint S_{ref} = 1. (a) Evolution per episode of the instant (black line) and moving average (over episodes, light orange line) reward (in absolute value). (b-f) Same as (a) for the (b) averaged (over time) drag, (c) ratio of the actual to target cross-sectional areas, (d) chord, (e) edge curvature radii and (f) inner thicknesses. All labels in (e-f) are ordered clockwise from the leading edge. The horizontal dashed lines in (d-f) mark the admissible values. (g) Shapes generated over the course of optimization for random episodes marked by the circle symbols in (a-c), together with corresponding iso-contours of vorticity. The last three shapes pertain to episodes 40, 70 and 100, respectively.

the 6% variance interval marked by the grey shade.

3.3. Discussion

We believe the above results assess the relevance of the proposed DRL-CFD framework for optimal shape design. Relying on low-dimensional parametrization of the body shape is one key parameter in this regard, as it improves the tractability of the optimization process and avoids the oscillations between points that have been found to occur when using a larger (about 10) number of control points. Nonetheless, we believe important to discuss the impact on robustness, and the extent to which decreasing the number of control points exaggerates (or not) the sensitivity to the curvature radii. This is because using different curvature radii to connect the same set of control

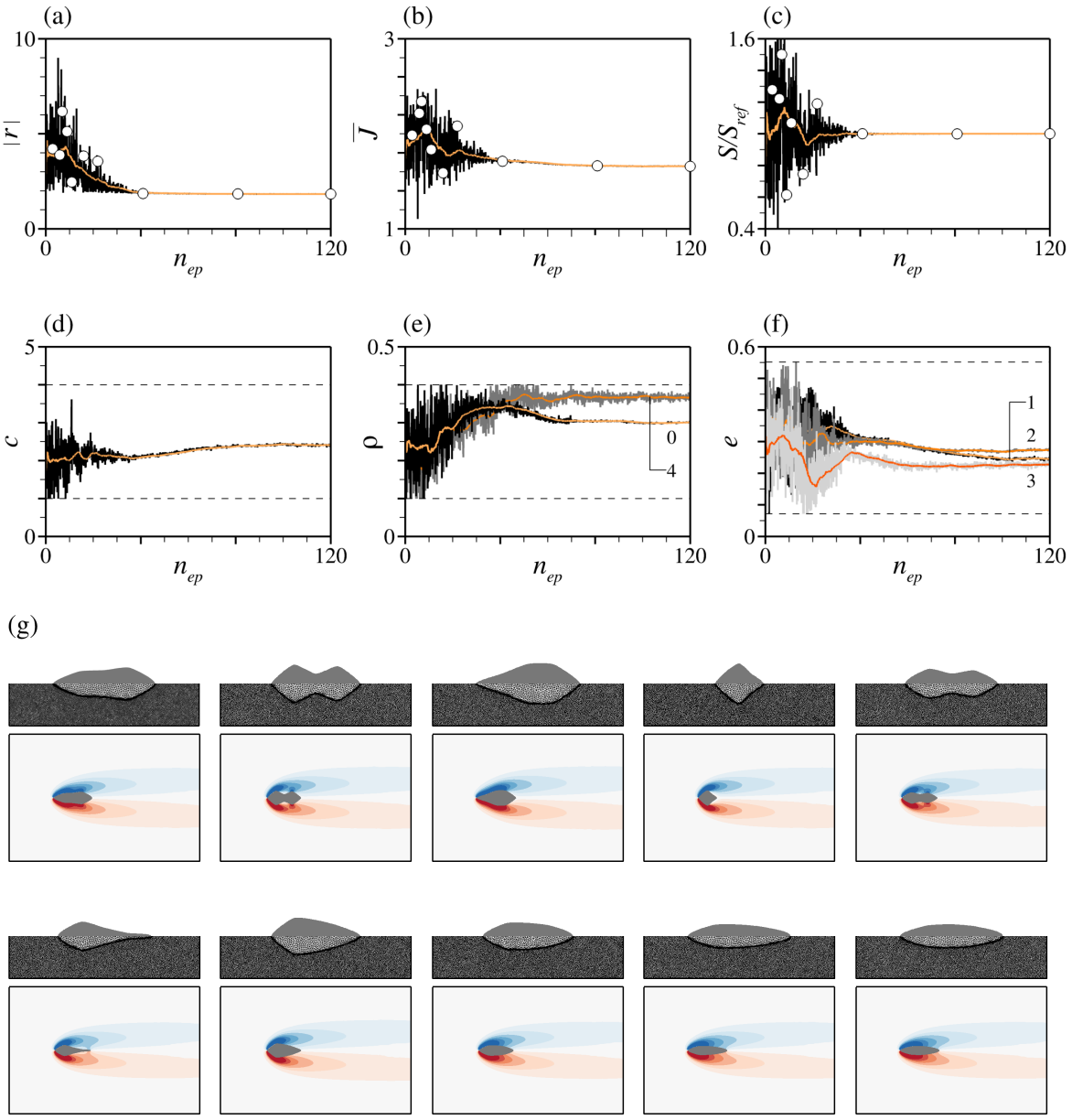

Figure 5: Maximum lift to drag ratio test case at Re = 20 under constant area constraint $S_{ref} = 1$. (a) Evolution per episode of the instant (black line) and moving average (over episodes, light orange line) reward (in absolute value). (b-f) Same as (a) for the (b) averaged (over time) drag, (c) ratio of the actual to target cross-sectional areas, (d) chord, (e) edge curvature radii and (f) inner thicknesses. All labels in (e-f) are ordered clockwise from the leading edge. The horizontal dashed lines in (d-f) mark the admissible values. (g) Shapes generated over the course of optimization for random episodes marked by the circle symbols in (a-c), together with corresponding iso-contours of vorticity. The last three shapes pertain to episodes 40, 80 and 120, respectively.

points can yield two slightly different cross-sectional areas, that in turn can earn two substantially different reward via the penalization term (this is not on the Bézier parametrization itself, though, only on the need to smoothly connect a discrete set of control points. For instance, one must also specify tangency at both endpoints of a spline).

As a first insight into this issue, we report here results obtained at Re = 1 using three alternative parametrizations:

- a case with $n_p = 7$ control points evolving the chord length, five inner thicknesses and two end curvature radii (which amounts to replicating the above reference case, but with two additional inner thicknesses, hence 8 independent design parameters),

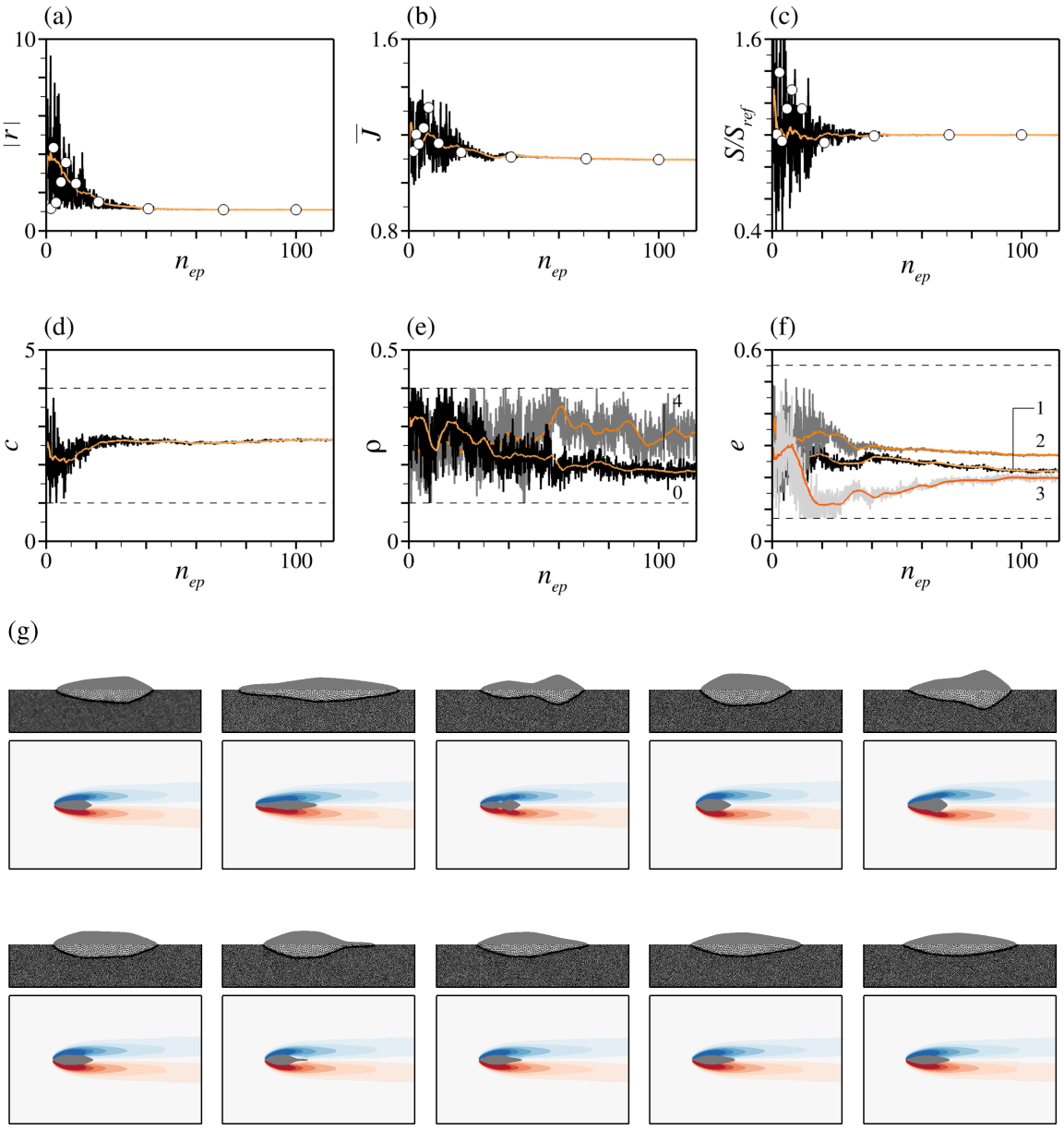

Figure 6: Maximum lift to drag ratio test case at Re = 50 under constant area constraint $S_{ref} = 1$. (a) Evolution per episode of the instant (black line) and moving average (over episodes, light orange line) reward (in absolute value). (b-f) Same as (a) for the (b) averaged (over time) drag, (c) ratio of the actual to target cross-sectional areas, (d) chord, (e) edge curvature radii and (f) inner thicknesses. All labels in (e-f) are ordered clockwise from the leading edge. The horizontal dashed lines in (d-f) mark the admissible values. (g) Shapes generated over the course of optimization for random episodes marked by the circle symbols in (a-c), together with corresponding iso-contours of vorticity. The last three shapes pertain to episodes 40, 70 and 120, respectively.

- a case with $n_p = 5$ points evolving the chord length, three inner thicknesses, two end curvature radii, plus an additional radius common to all inner control points (which amounts to replicating the reference case, but with one additional inner curvature radius, hence 7 independent design parameters),

- a case with $n_p = 7$ points whose thickness distribution is frozen, as obtained interpolating from the reference $n_p = 5$ optimal (for which it suffices to sample the connecting Bezier curves at the relevant positions), after which a dedicated DRL agent restores the proper cross-sectional area by evolving two end curvature radii, plus an additional radius common

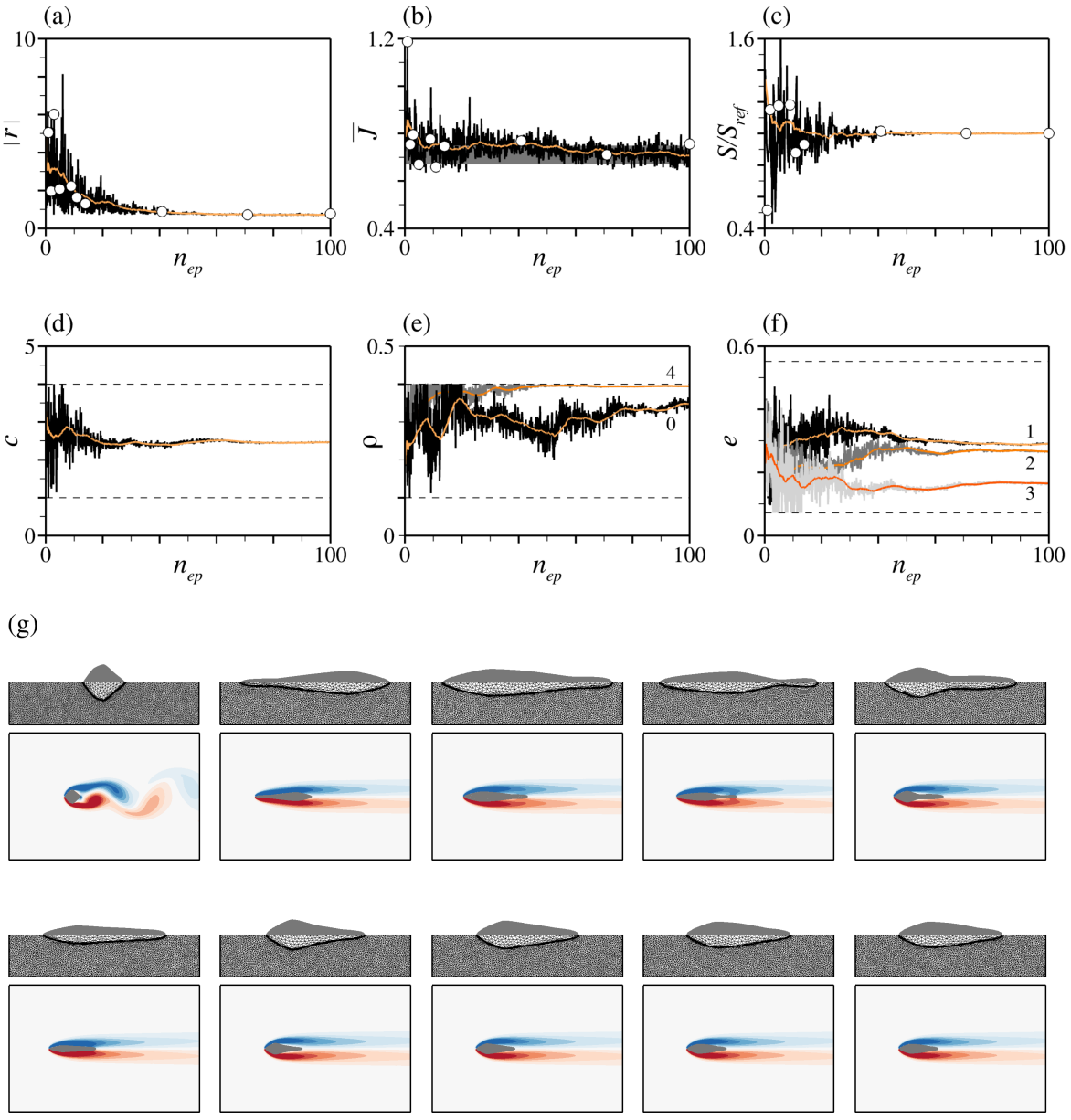

Figure 7: Maximum lift to drag ratio test case at Re = 100 under constant area constraint $S_{ref} = 1$. (a) Evolution per episode of the instant (black line) and moving average (over episodes, light orange line) reward (in absolute value). (b-f) Same as (a) for the (b) averaged (over time) drag, (c) ratio of the actual to target cross-sectional areas, (d) chord, (e) edge curvature radii and (f) inner thicknesses. The grey shade in (b) marks the 6% variance interval with respect to the average over the 10 latest learning episodes. All labels in (e-f) are ordered clockwise from the leading edge. The horizontal dashed lines in (d-f) mark the admissible values. (g) Shapes generated over the course of optimization for random episodes marked by the circle symbols in (a-c), together with corresponding iso-contours of vorticity. The last three shapes pertain to episodes 40, 70 and 120, respectively.

to all inner control points (hence, 3 independent design parameters) with reward

$$r = -|S - S_{ref}|\,, \tag{7}$$

formally identical to (2) with $J = 0$ and $\beta = 1$.

The results reported in table 3 exhibit limited discrepancy with respect to the reference (reproduced from table 2 in the first column), as the maximum deviation on the chord and the inner thicknesses is by 4%. All runs converge to similar curvature radii at the front. The value at the rear is noticeably different, but with little to no effect on the reward, objective function and shape (as shown in figure 8), which simply reflects the smallness of the reward gradients with respect to the

				Case setup
5	5	7	7	Nb. points
×	✓	×	✓	Inner curv. radius
6	7	8	3	Dimensionality
				Optimal
1.95	1.96	1.87	1.95	Chord length
0.4	0.332	0.4	0.398	Inner curv. radius
0.309	0.359	0.310	0.394	LE curv. radius
0.297	0.284	0.233	0.206	⇓ Thickness
0.362	0.367	0.340	0.324	
0.299	0.303	0.369	0.359	
-	-	0.341	0.331	
-	-	0.258	0.227	
0.115	0.159	0.392	0.389	TE curv. radius
1.00	1.00	1.00	1.00	Ratio of actual to target CSA
13.10	13.09	13.09	13.09	Drag

Table 3: Sensitivity of the drag minimization problem at Re = 1 to the discretization parameters. Leading-edge (front end) and trailing edge (rear end) data are labeled LE and TE, respectively. The first column is reproduced from table 2.

Figure 8: (a) Reference optimal shape for the minimum drag test case at Re = 1 with $n_p = 5$ control points and fixed inner curvature radius. (b) Same as (a) with $n_p = 7$ control points and fixed inner curvature radius. (c) Same as (a) with $n_p = 5$ control points and variable inner curvature radius. (d) Reference optimal shape discretized with $n_p = 7$ control points, after DRL has adjusted the end and inner curvature radii to restore the proper cross-sectional area.

control variables in the vicinity of the optimal. Although the impact needs to be assessed on a case to case basis, this suggests that the method ability to provide robust optima may not be strained by the use of low-end geometrical parametrizations.

4. Application to optimal aerodynamic design

4.1. Test case description

We apply now the method to more meaningful aerodynamic shape optimization problems, as we seek the shape maximizing the lift to drag ratio (used as an indicator of the aerodynamic efficiency) induced by a surrounding uniform flow at angle of attack of $\alpha = 30°$. A sketch of the configuration is provided in figure 9. A Cartesian coordinate system is used with origin at quarter chord length from the leading edge. The target cross-sectional area is set to $S_{ref} = 0.0822$, which corresponds to the CSA of a NACA (National Advisory Committee for Aeronautics) 0012 airfoil. The objective function is

$$J = \frac{L}{D}\,, \tag{8}$$

and the weighing coefficient is set to $\beta = 100$. Two time-dependent flow regimes are modeled after either the Navier–Stokes or the uRANS equations, for which all CFD environments use the numerical simulation parameters provided in table 4.

As has been done for the drag minimization test case, we simplify the parametrization by setting all inner curvature radii to 0.4. Additionally, we fix the chord length to $c = 1$ for the chord Reynolds number $\mathrm{Re} = U_\infty c/\nu$ to remain constant over the course of optimization (which we believe is necessary to meaningfully compare the performances). This leaves n_p independent

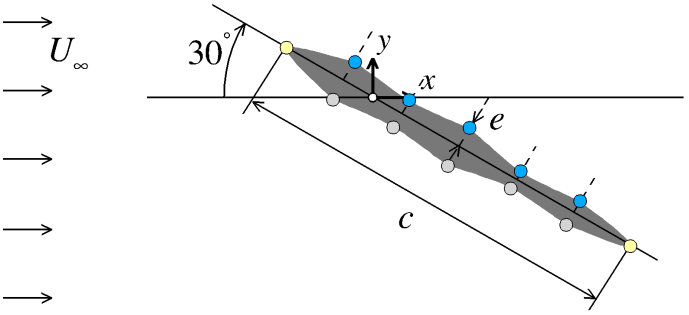

Figure 9: Schematic diagram of the maximum lift to drag ratio test case.

design variables, two end curvature radii $\rho_{j\in\{0,n_p-1\}}$ and $n_p - 2$ inner thicknesses $e_{k\in\{1,\dots,n_p-2\}}$. The network action output consists accordingly of values $(\hat{\rho}_j, \hat{e}_k)$ in $[-1;1]^{n_p}$, converted into the actual physical quantities using the same mapping (6), only we set $\delta e = 0.03$ to account for the smaller target CSA. All results in the following are for $n_p = 5$, for which DRL evolves five design parameters, two end curvature radii and three inner thicknesses.

4.2. Laminar regime at Re = 250

We consider first a 2-D laminar case at Re = 250 modeled after the Navier–Stokes equations, for which the dimensions of the computational domain provided in table 4 yield a blockage ratio of 2.5%. All mesh adaptations are performed under the constraint of a fixed total number of elements $n_{el} = 100000$. A total of 100 episodes has been run for this case, that yield the variety of shapes illustrated in figure 10, together with their respective iso-contours of (instantaneous) vorticity. The general picture is that all shapes exhibit an oscillating pattern of leading- and trailing-edge vortex shedding following the shedding of the initial leading-edge vortex. This stems from the interaction between the (lower) negative vorticity sheet, that separates at the leading edge and then rolls up into a large clockwise vortex, and the (upper) positive vorticity sheet, that remains attached to the windward side and rolls up counter-clockwise from the trailing edge (in average, this yields a massive separation originating at the leading edge and extending on the leeward side, all the way to the trailing edge; not shown here). The Strouhal number for vortex shedding frequency built from the windward width is $S_t = fc\sin\alpha/u_\infty \sim 0.13$ (regardless of the shape), which is identical to experimental measurements performed on a high-aspect ratio NACA 0012 airfoil under the same incidence at Re = 100 [81].

The moving reward in figure 10(a) increases almost monotonically and reaches a plateau after about 40 episodes. The optimal lift to drag ratio computed as the average over the 10 latest episodes is $1.24 \pm 1.0\%$, at which point the cross-sectional area is equal to its target value down to the fifth decimal place. We note that 40 episodes is actually the number of episodes needed for the end radii to converge, as the thickness distribution exhibits excellent convergence after as little as 20 episodes. Interestingly, the agent has generated a wing-like optimal shape representative of a high-lift configuration without any priori knowledge of aerodynamic concepts: the optimal features a rounded leading edge to help maintain a smooth airflow (with curvature radius $0.394 \pm 0.01\%$ close to maximum) and a sharp trailing edge to generate lift (with curvature radius 0.156 ± 0.02 close to minimum). The optimal lift to drag ratio exceeds that of the equivalent ellipse (i.e., of major diameter c and minor diameter $2S_{ref}/\pi c$, for the area to be equal to S_{ref}) by 6% and that of a NACA 0012 airfoil by 1%, as has been estimated from dedicated in-house calculations. This is small but consistent with the overall lack of sensitivity, as the objective function in figure 10(c) actually remains within 3% of the optimal over the course of optimization, as indicated by the grey shade delimiting the related variance interval.

4.3. Turbulent (transitional) regime at Re = 5000

We consider now a case at Re = 5000 corresponding to the ultra-low Reynolds number regime, that has assumed greater significance in the last few decades due to relevance for micro air vehicles and micro-turbines [82, 83]. We believe this constitues a valuable first step towards applying

			Case setup
250	5000	»	Reynolds number
5	»	»	Nb. points
5	»	3	Nb. design variables
			CFD
2	»	3	Dimensionality
-	RANS	»	Turb. model
0.125	0.05	»	Time-step
[100;150]	[150;200]	[100;150]	Averaging time span
100	»	90	Penalty coeff.
$[-10;20]\times[-10;10]$	$[-6;15]\times[-7;7]$	$[-5;10]\times[-5;5]\times[0;5]$	Mesh dimensions
100000	120000	500000	Nb. mesh elements
0.0005	»	0.001	Interface $\perp$ mesh size
			PBO
100	100	80	Nb. episodes
14	14	12	Nb. environments
20mn	2h45mn	9h30mn	CPU time†‡
35h	275h	760h	Resolution time‡
			Parameter ranges
-	-	-	Chord length
[0.1;0.4]	»	-	LE curv. radius
[0.024;0.084]	»	»	↓ Thickness
[0.03;0.09]	»	»	
[0.024;0.084]	»	»	
[0.1;0.4]	»	-	TE curv. radius
			Optimal
1	1	1	Chord length
0.394	0.398	0.3	LE curv. radius
0.0638	0.0549	0.0420	↓ Thickness
0.0514	0.0627	0.0536	
0.0253	0.0252	0.0454	
0.156	0.104	0.1	TE curv. radius
1.00	1.00	0.996	Ratio of actual to target CSA
1.24	1.54	1.34	Lift to drag ratio

Table 4: Case setup, simulation parameters and convergence data for the lift to drag ratio maximization problem, as computed by averaging over the 10 latest learning episodes. † All CPU times provided per episode and per environment. ‡ All values obtained averaging over 5 independent runs using 12 cores.

the method to more prototypal aerodynamic applications in which airfoils operate at Reynolds numbers of $\sim 10^6$ and exhibit some degree of stochastic dynamics (as they carry turbulent energy distributed over a wide range of scales with varying degrees of spatial and temporal coherence), which might lead to high variance gradient estimates and hamper learning. Here, at the high value of angle of attack considered, the flow is expected to be transitional, for instance, transition in the wake of a NACA 0012 has been shown to occur in the separated shear-layer, shortly after the leading edge, at a location strongly dependent on the level of external noise [84]. This has been confirmed vetting preliminary Navier–Stokes simulations for which the built-in small-scale component of the VMS solution acts as an implicit large eddy simulation. While the solutions (not reported here for the sake of conciseness) are dominated by the large-scale component, with small-scale turbulence noticeably absent downstream, intermittent small-scale fluctuations develop on the leeward side, that prompt asymmetric vortex street (at least is the trailing edge is not too

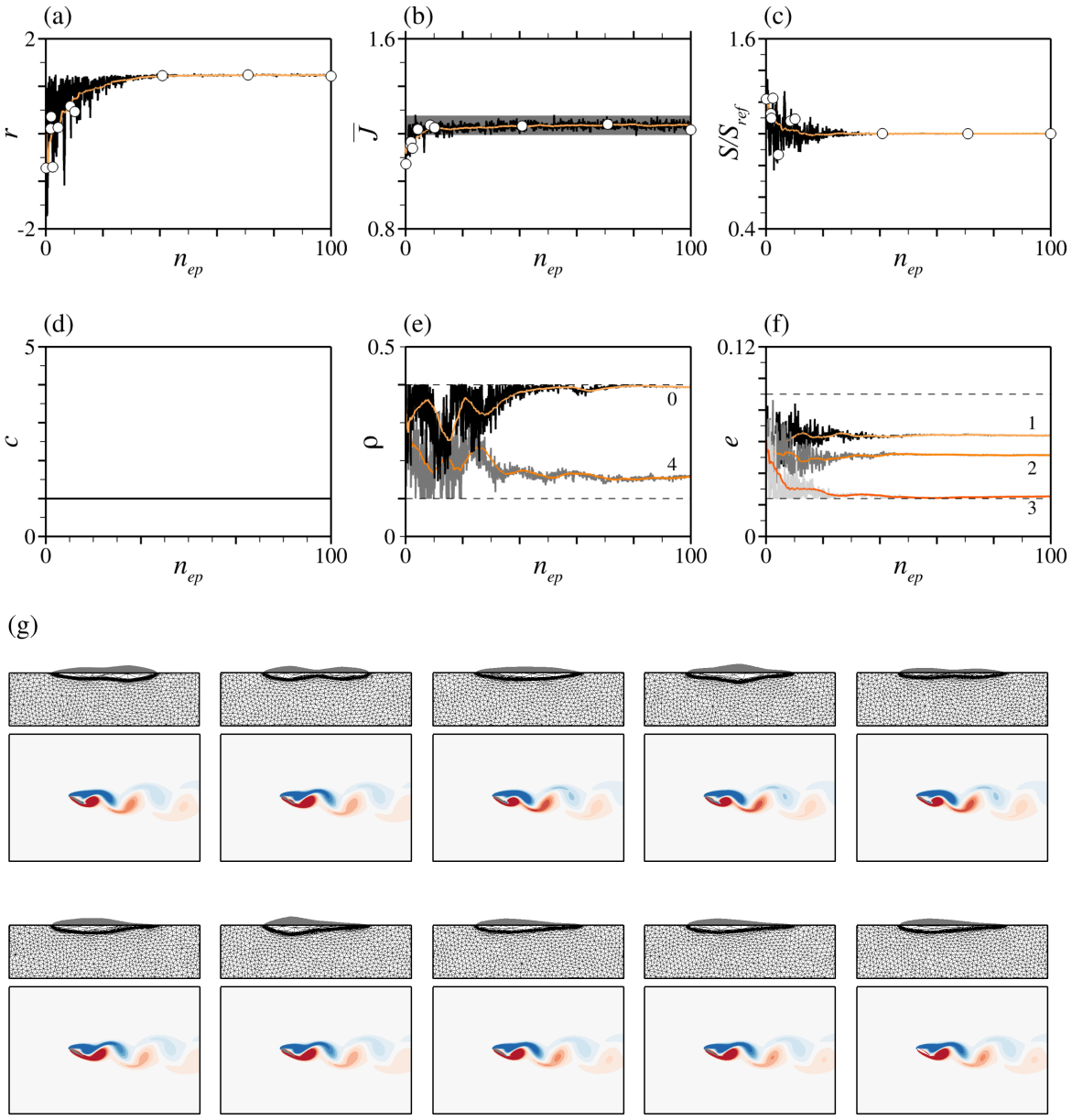

Figure 10: Maximum lift to drag ratio test case at Re = 250 under constant area constraint S_{ref} = 0.0822. (a) Evolution per episode of the instant (black line) and moving average (over episodes, light orange line) reward. (b-f) Same as (a) for the (b) averaged (over time) lift to drag ratio, (c) ratio of the actual to target cross-sectional areas, (d) chord (fixed), (e) edge curvature radii and (f) inner thicknesses. The grey shade in (b) marks the 3% variance interval with respect to the average over the 10 latest learning episodes. All labels in (e-f) are ordered clockwise from the leading edge. The horizontal dashed lines in (e-f) mark the admissible values. (g) Shapes generated over the course of optimization for random episodes marked by the circle symbols in (a-c), together with corresponding iso-contours of vorticity. The last three shapes pertain respectively to episodes 40, 70 and 100.

sharp for the separation point to be free to move) with vortices convected downstream along an axis inclined upward with respect to the streamwise direction, similar to the behavior observed in 2-D LES simulations of the transitional flow past a circular cylinder [85].

Accordingly, the case is modeled here after the uRANS equations, using negative Spalart–Allmaras as turbulence model. Such an approach is not without shortcomings (namely RANS is inherently designed to damp out the small-scales, and Spalart–Allmaras assumes fully turbulent behavior), but given the cost of accurately resolving the complex, unsteady vortex interaction described above, we believe the deficiencies are more than offset by the tremendous gain in computational efficiency derived from the relatively coarse meshes necessary to predict the most important

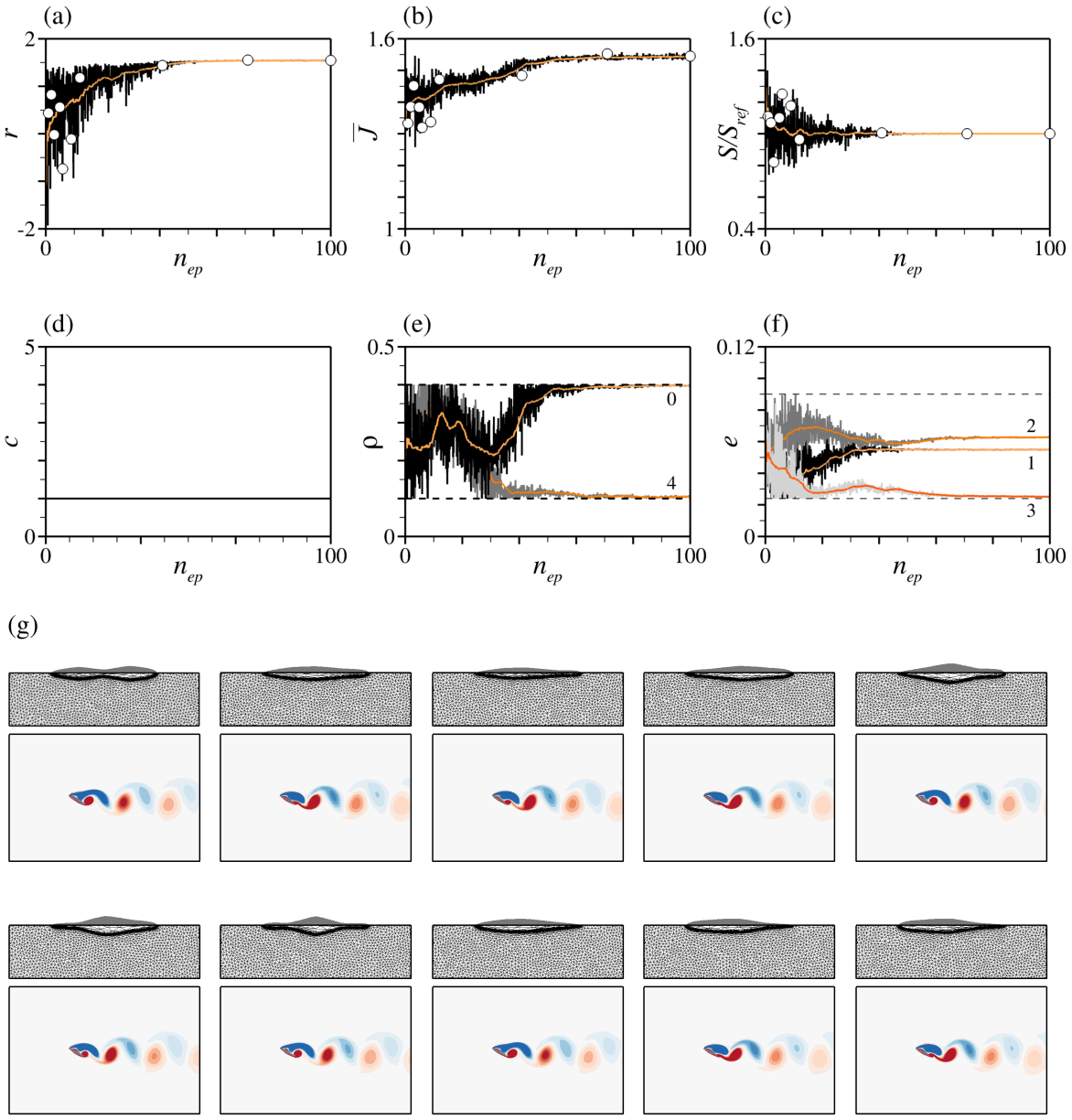

Figure 11: Maximum lift to drag ratio test case at Re = 5000 with negative Spalart–Allmaras turbulence model, under constant area constraint S_{ref} = 0.0822. (a) Evolution per episode of the instant (black line) and moving average (over episodes, light orange line) reward. (b-f) Same as (a) for the (b) averaged (over time) lift to drag ratio, (c) ratio of the actual to target cross-sectional areas, (d) chord (fixed), (e) edge curvature radii and (f) inner thicknesses. All labels in (e-f) are ordered clockwise from the leading edge. The horizontal dashed lines in (d-f) mark the admissible values. (g) Shapes generated over the course of optimization for random episodes marked by the circle symbols in (a-c), together with corresponding iso-contours of vorticity. The last three shapes pertain respectively to episodes 40, 70 and 100.

large scale features of the flow. In practice, a scaled-down computational domain is used, whose dimensions reported in table 4 yield a blockage ratio of 3.5%. All mesh adaptations are performed under the constraint of a fixed total number of elements n_{el} = 120000. A total of 100 episodes has been run, for which the selected iso-contours of vorticity documented in figure 11 are reminiscent of their laminar counterparts, with in-line vortex shedding (since the effect of the intermittent small-scale fluctuations has been lumped into the eddy viscosity model) and robust shedding frequency S_t = 0.15.

The moving average reward in figure 11(a) is seen to converges within about 50 episodes but the thickness distribution again converges faster (within roughly 40 episodes). As was already the

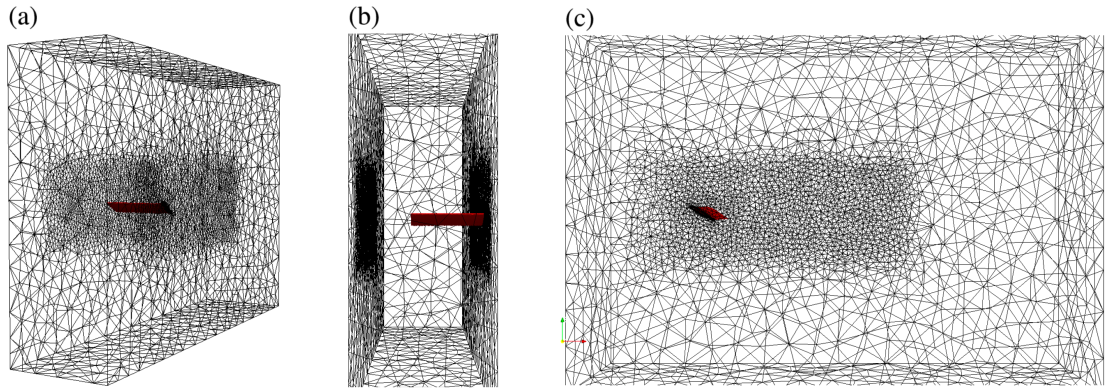

Figure 12: Anisotropic adapted mesh around an immersed three-dimensional unswept, rectangular wing. (a) Three-dimensional view. (b) Front view. (c) Side view.

case at Re = 250, the optimal resembles the airfoil of an airplane wing, with a rounded leading edge and a sharp trailing edge. The end radii are nearly identical to their laminar counterparts, but the shape is streamlined differently, namely it is a tad thinner in the front ($0.0549 \pm 0.2\%$ at Re = 5000 vs. $0.0638 \pm 0.2\%$ at Re = 250) but slightly thicker in the center ($0.0627 \pm 0.4\%$ at Re = 5000 vs. $0.0514 \pm 0.25\%$ at Re = 250). The optimal lift to drag ratio ($1.54 \pm 0.3\%$) exceeds that of the equivalent ellipse by 13% but is ultimately identical to that of a NACA 0012, despite the objective function exhibiting substantial variations in figure 11(b). The inability to outperform a conventional airfoil should not be interpreted as failure of the method, though, as aerodynamic shape design classically requires fine-tuning of the local geometry for a gain that often adds up to a few percent. This is not manageable here because the low number of degrees of freedom inevitably constrains the underlying space of shapes, and the expected gain is comparable to the typical convergence threshold of a DRL run. We believe the results should rather be considered proof that DRL can start from the ground up and generate shapes that perform just as well as a conventional airfoil. Actually, there is ample room for improvement if the optimization is to be tailored to airfoil shape optimization problems (which it is not here for the sake of generality), one may seek for instance to locally refine the DRL optimal by repeating the same analysis, but clustering the control points in specific regions of interest (e.g., the leading-edge, or the rear-end of the leeward side), or to rely on alternative parametrizations better suited to airfoils, such as CST [86].

5. Extension to 3-D shape optimization.

The ultra low-Reynolds number case at Re = 5000 is extended here to 3-D to assess the extent to which the approach carries over to three-dimensional shape optimization. All shapes generated over the course of optimization are unswept, rectangular wings, whose cross-section is set up from the DRL outputs following the exact same process as in sections 3 and 4. The span aspect ratio (relative to the chord length) is set to 3 in our implementation. A Cartesian coordinate system is used with origin in the mid-span plane, at quarter chord length from the leading edge. The number of control points remains set to $n_p = 5$, but we force the leading and trailing edge curvature radii to 0.3 (round edge) and 0.1 (sharp edge) to keep the computational cost manageable, which leaves $n_p - 2 = 3$ independent design variables corresponding to the inner thicknesses. In practice, only a half-span wing body is simulated with symmetry boundary condition prescribed at the mid-span. The computational domain shown in figure 12 is a rectangular prism, whose dimensions reported in table 4 yield a blockage ratio of 5%. All mesh adaptations are performed under the constraint of a fixed total number of elements $n_{el} = 500000$. This is likely insufficient to claim true numerical accuracy, but given the numerical cost (960 3-D simulations total, each of which is performed on 12 cores and lasts about 10h, hence 9600h of total CPU cost), we believe this is a reasonable compromise to assess feasibility while producing qualitative results to build on.

A total of 80 episodes has been run for this case, using a slighly lower weighing coefficient $\beta = 90$ (to take into account that the coarser mesh yields a small loss in accuracy in the computation of

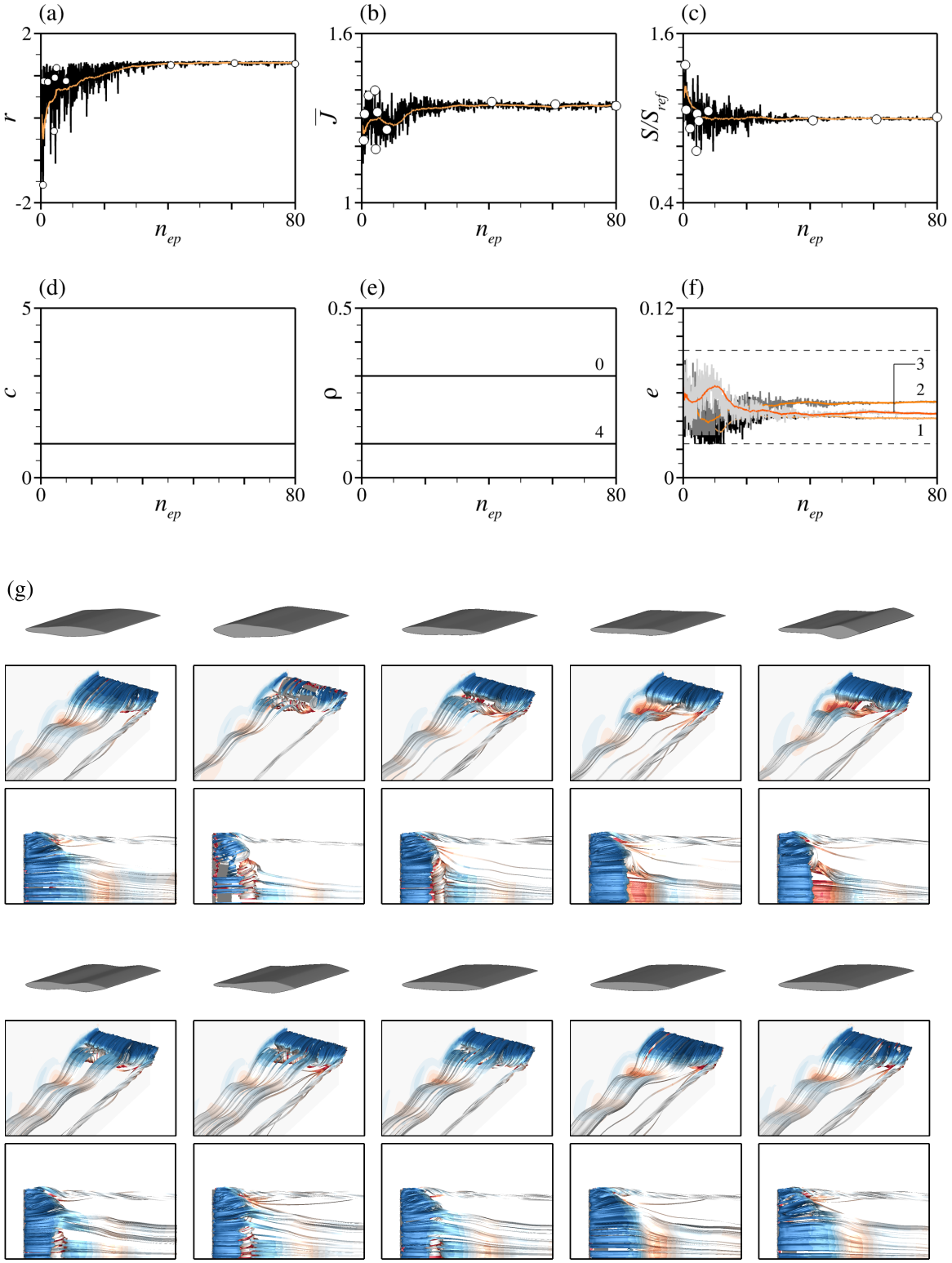

Figure 13: Maximum lift to drag ratio test case in 3-D at Re = 5000 with negative Spalart–Allmaras turbulence model, under constant area constraint S_{ref} = 0.0822. (a) Evolution per episode of the instant (black line) and moving average (over episodes, light orange line) reward. (b-f) Same as (a) for the (b) averaged (over time) lift to drag ratio, (c) ratio of the actual to target cross-sectional areas, (d) chord (fixed over the course of optimization), (e) edge curvature radii (also fixed) and (f) inner thicknesses. (g) Shapes generated over the course of optimization for random episodes marked by the circle symbols in (a-c), together with corresponding iso-contours of vorticity. The last three shapes pertain respectively to episodes 40, 70 and 100.

the cross-sectional area). Several representative flow patterns computed over the course of optimization are illustrated in figure 13 to display the increased degree of complexity due to transverse inhomogeneities. All solutions exhibit vortex shedding, which is because the span aspect ratio is large enough for the tip vortex to remain relatively steady. Conversely, preliminary simulations carried out at lower aspect ratios of order 1 systematically relaxed to steady-state, due to the strong tip-vortex induced downwash over the entire span (the same behavior has been reported in laminar flows at Reynolds numbers of about in the range of a few hundreds [87], and is ascribed here to the RANS damping of the small-scale transverse motion, that should otherwise strengthen the unsteadiness). The moving averager reward in figure 13 plateaus after about 35 episodes. The 3-D distribution is almost front-rear symmetric but the shape itself surprisingly slightly thinner in the front than in the rear, although the rear is ultimately more streamlined due to the smaller trailing edge curvature radius. Compared to its 2-D counterpart, the 3-D optimal is thinner in the front and in the center, but much thicker in the rear. The optimal lift to drag ratio ($1.34 \pm 0.5\%$) exceeds that of the equivalent ellipse by 5% and is identical to that of a NACA 0012. This is consistent with the above findings, in the sense that the DRL optimal performs at the level of a conventional airfoil, and that the limited improvement with respect to the equivalent ellipse should not be taken as an indictment of the method, just a consequence of the flow regime considered (precisely because a similar improvement is achieved using a NACA 0012).

6. Conclusion

Shape optimization in computational fluid dynamics systems is achieved here training fully connected networks with PBO, a recently introduced deep reinforcement algorithm at the crossroad of policy gradient methods and evolution strategies. PBO is single-step, meaning that the DRL agent gets only one attempt per learning episode at finding the optimal. The numerical reward fed to the PBO agent is computed with a finite elements CFD environment solving stabilized weak forms of the governing equations (Navier–Stokes, otherwise uRANS with negative Spalart–Allmaras as turbulence model) with a combination of variational multiscale approach, immersed volume method and anisotropic mesh adaptation.

Several cases are documented, for which shapes with fixed camber line, angle of attack and cross-sectional area are generated by varying a chord length and a symmetric thickness distribution (and possibly extruding in the off-body direction), connecting consecutive points by a cubic Bézier curve using local position and curvature information. The classical problem of finding the 2-D shape of minimum drag in a uniform flow is revisited first to validate and assess the method capabilities. The method is also applied to the more practically meaningful problem of finding the shape of maximum lift to drag ratio (in 2-D or 3-D) at an incidence of 30° and under constant chord Reynolds number. The DRL optimal increases the performance the equivalent ellipse (i.e., the ellipse of same cross-sectional area) by 13% in 2-D and 5% in 3-D. It is systematically found to perform just as well as a conventional airfoil, despite DRL starting from the ground up and having no priori knowledge of aerodynamic concepts. Exhaustive convergence and efficiency data are reported here with the hope to foster future comparisons, but it is worth emphasizing that we did not seek to optimize said effciency, neither by optimizing the PBO meta-parameters, nor by using pre-trained deep learning models (as is done in transfer learning).

Fluid dynamicists have just begun to gauge the relevance of DRL and its application to optimal shape design. This research weighs in on this issue and shows that the proposed single-step method holds a high potential as a reliable, go-to black-box optimizer for complex CFD problems. Moreover, the optimization process is entirely domain-agnostic, meaning that the proposed framework allows for easy application to any domain in which shape optimization may be beneficial. We believe further work should now focus on the challenges specific to fluid mechanics that still prevent DRL capabilities from meeting the requirements for practical deployment, e.g., stochasticity, sampling efficiency (CFD environments are resource expensive as they routinely involve numerical simulations with tens or hundreds of millions of degrees of freedom, while classical RL methods have low sample efficiency, i.e., many trials are required for the agent to learn a purposive behavior), the need to leverage experience from multiple agents learning concurrently (multi-agent DRL) or to train an agent in reasoning about several weighted objectives (multi-objective reward).

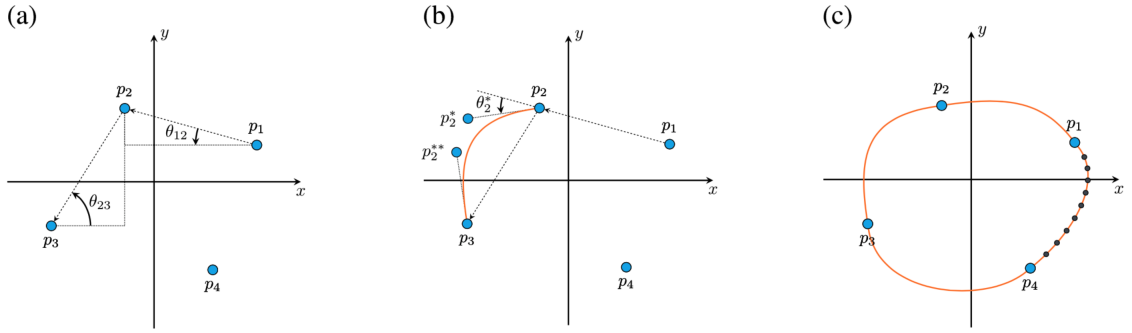

Figure 14: Shape generation using cubic Bézier curves. Each subfigure illustrates one of the consecutive steps used in the process. (a) Compute angles between points and compute an average angle θ_i^* around each point. (b) Compute supplemental control points coordinates from averaged angles and generate cubic Bézier curve. (c) Sample all Bézier lines and export for mesh immersion.

Acknowledgements

This work is supported by the Carnot M.I.N.E.S. Institute through the M.I.N.D.S. project (CFL2022).

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Appendix A. Shape generation using Bézier curves

This section describes the process followed to generate shapes from a set of n_p control points. Once the position has been reconstructed from the agent outputs, the angles between consecutive points are computed. An average angle is then computed around each point (see Fig. 14(a)) as

$$\theta_i^* = r\theta_{i-1,i} + (1-r)\theta_{i,i+1}\,, \tag{A.1}$$

where $r \in [0;1]$ is the curvature radius that control the local sharpness of the curve. Then, each pair of points is joined using a cubic Bézier curve, defined by four points: the first and last points, p_i and p_{i+1} belong to the curve, while the second and third ones, p_i^* and p_i^{**}, are supplemental control points that define the tangent of the curve at p_i and p_{i+1}. The tangents atp_i and p_{i+1} are respectively controlled by $\theta_i$and θ_{i+1} (Fig. 14(b)). A final sampling of the successive Bézier curves leads to a boundary description of the shape (Fig. 14(c)). Using this method, a wide variety of shapes can be attained.

References

[1] P. Gangl, U. Langer, A. Laurain, H. Meftahi, K. Sturm, Shape optimization of an electric motor subject to nonlinear magnetostatics, SIAM J. Sci. Comput. 37 (2015) B1002–B1025.

[2] R. Udawalpola, M. Berggren, Optimization of an acoustic horn with respect to efficiency and directivity, Int. J. Numer. Methods Eng. 73 (2008) 1571–1606.

[3] M. Hintermüller, W. Ring, A second order shape optimization approach for image segmentation, SIAM J. Appl. Math. 64 (2004) 442–467.

[4] J. Pinzon, M. Siebenborn, A. Vogel, Parallel 3d shape optimization for cellular composites on large distributed-memory clusters, Adv. Model. Simul. Eng. Sci. 7 (2020) 117–135.

[5] P.-I. Schneider, X. G. Santiago, V. Soltwisch, M. Hammerschmidt, S. Burger, C. Rockstuhl, Benchmarking five global optimization approaches for nano-optical shape optimization and parameter reconstruction, arXiv preprint arXiv:1809.06674 (2019).

[6] O. Pironneau, On optimum profiles in stokes flow, J. Fluid Mech. 59 (1973) 117–128.

[7] O. Pironneau, On optimum design in fluid mechanics, J. Fluid Mech. 64 (1974) 97–110.

[8] J. J. Corbett, H. W. Koehler, Updated emissions from ocean shipping, J. Geophys. Res. 108 (2003) 4650–64.

[9] A. L. Marsden, M. Wang, B. Mohammadi, P. Moin, Shape optimization for aerodynamic noise control, Center for Turbulence Research Annual Brief (2001).

[10] I. Rodriguez-Eguia, I. Errasti, U. Fernandez-Gamiz, J. M. Blanco, E. Zulueta, A. Saenz-Aguirre, A parametric study of trailing edge flap implementation on three different airfoils through an artificial neuronal network, Symmetry 12 (2020) 828.

[11] M. C. G. Hall, Application of adjoint sensitivity theory to an atmospheric general circulation model, J. Atmos. Sci. 43 (1986) 2644–2651.

[12] A. Jameson, L. Martinelli, N. A. Pierce, Optimum aerodynamic design using the Navier–Stokes equations, Theor. Comput. Fluid Dyn. 10 (1998) 213–237.

[13] M. D. Gunzburger, Perspectives in flow control and optimization, SIAM, Philadelphia, 2002.

[14] S. N. Skinner, H. Zare-Behtash, State-of-the-art in aerodynamic shape optimisation methods, Appl. Soft Comput. 62 (2018) 933–962.

[15] J. H. Holland, Genetic algorithms, Sci. Am. 267 (1992) 66–73.

[16] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Procs. of ICNN'95-international conference on neural networks, 1995, pp. 1942–1948.

[17] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, E. Teller, Equation of state calculations by fast computing machines, J. Chem. Phys 21 (1953) 1087–1092.

[18] R. Hassan, B. Cohanim, O. De Weck, G. Venter, A comparison of particle swarm optimization and the genetic algorithm, AIAA 2005-1897 (2005).

[19] Z. Han, C. Xu, L. Zhang, Y. Zhang, K. Zhang, S. Wenping, Efficient aerodynamic shape optimization using variable-fidelity surrogate models and multilevel computational grids, Chinese J. Aeronaut. 33 (2020) 31–47.

[20] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, P. K. Tucker, Surrogate-based analysis and optimization, Prog. Aerosp. Sci. 41 (2005) 1–28.

[21] O. Chernukhin, D. W. Zingg, Multimodality and global optimization in aerodynamic design, AIAA J. 51 (2013) 1342–1354.

[22] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, D. Hassabis, Mastering the game of go without human knowledge, Nature 550 (2017) 354–359.

[23] M. Moravčik, M. Schmid, N. Burch, V. Lisy, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, M. Bowling, DeepStack: expert-level artificial intelligence in heads-up no-limit poker, Science 356 (2017) 508–513.

[24] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal Policy Optimization Algorithms, arXiv preprint arXiv:1707.06347 (2017).

[25] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, M. Hutter, Learning agile and dynamic motor skills for legged robots, Sci. Robot. 4 (2019) eaau5872.

[26] A. Bernstein, E. Burnaev, Reinforcement learning in computer vision, in: Procs. of the 10th International Conference on Machine Vision, 2018.

[27] Y. Deng, F. Bao, Y. Kong, Z. Ren, Q. Dai, Deep direct reinforcement learning for financial signal representation and trading, IEEE Trans. Neural Netw. Learn. Syst. 28 (2017) 653–664.

[28] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, A. Shah, Learning to drive in a day, arXiv preprint arXiv:1807.00412 (2018).

[29] A. Bewley, J. Rigley, Y. Liu, J. Hawke, R. Shen, V.-D. Lam, A. Kendall, Learning to drive from simulation without real world labels, arXiv preprint arXiv:1812.03823 (2018).

[30] W. Knight, Google just gave control over data center cooling to an AI, `http://www.technologyreview.com/s/611902/google-just-gave-control-over-data-center-cooling-to-an-ai/` (2018).

[31] J. Viquerat, P. Meliga, E. Hachem, A review on deep reinforcement learning for fluid mechanics: an update, arXiv preprint arXiv:2107.12206 (2021).

[32] J. Rabault, M. Kuchta, A. Jensen, U. Réglade, N. Cerardi, Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control, J. Fluid Mech. 865 (2019) 281–302.

[33] J. Rabault, A. Kuhnle, Accelerating deep reinforcement learning strategies of flow control through a multi-environment approach, Phys. Fluids 31 (2019) 094105.

[34] M. A. Elhawary, Deep reinforcement learning for active flow control around a circular cylinder using unsteady-mode plasma actuators, arXiv preprint arXiv:2012.10165 (2020).

[35] M. Holm, Using deep reinforcement learning for active flow control, Ph.D. thesis, Master Thesis University of Oslo (2020).

[36] J. Rabault, F. Ren, W. Zhang, H. Tang, H. Xu, Deep reinforcement learning in fluid mechanics: a promising method for both active flow control and shape optimization, J. Hydrodynam. 32 (2020) 234–246.

[37] F. Ren, H. Hu, H. Tang, Active flow control using machine learning: A brief review, J. Hydrodynam. 32 (2020) 247–253.

[38] H. Tang, J. Rabault, A. Kuhnle, Y. Wang, 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 (2020) 053605.

[39] M. Tokarev, E. Palkin, R. Mullyadzhanov, Deep reinforcement learning control of cylinder flow using rotary oscillations at low Reynolds number, Energies 13 (2020) 5920.

[40] H. Xu, W. Zhang, J. Deng, J. Rabault, Active flow control with rotating cylinders by an artificial neural network trained by deep reinforcement learning, J. Hydrodynam. 32 (2020) 254–258.

[41] H. Ghraieb, J. Viquerat, A. Larcher, P. Meliga, E. Hachem, Single-step deep reinforcement learning for open-loop control of laminar and turbulent flows, Phys. Rev. Fluids 6 (2021, in press) 053902.

[42] R. Paris, R. Beneddine, J. Dandois, Robust flow control and optimal sensor placement using deep reinforcement learning, J. Fluid Mech. 913 (2021).

[43] S. Qin, S. Wang, G. Sun, An application of data driven reward of deep reinforcement learning by dynamic mode decomposition in active flow control, arXiv preprint arXiv:arXiv:2106.06176 (2021).

[44] F. Ren, J. Rabault, H. Tang, Applying deep reinforcement learning to active flow control in weakly turbulent conditions, Phys. Fluids 33 (2021) 037121.

[45] V. Belus, J. Rabault, J. Viquerat, Z. Che, E. Hachem, U. Réglade, Exploiting locality and translational invariance to design effective deep reinforcement learning control of the 1-dimensional unstable falling liquid film, AIP Adv. 9 (2019) 125014.

[46] J. Viquerat, J. Rabault, A. Kuhnle, H. Ghraieb, A. Larcher, E. Hachem, Direct shape optimization through deep reinforcement learning, J. Comput. Phys. 428 (2021) 110080.

[47] G. Novati, S. Verma, D. Alexeev, D. Rossinelli, W. M. van Rees, P. Koumoutsakos, Synchronisation through learning for two self-propelled swimmers, Bioinspir. Biomim. 12 (2017) 036001.

[48] S. Verma, G. Novati, P. Koumoutsakos, Efficient collective swimming by harnessing vortices through deep reinforcement learning, Proc. Natl. Acad. Sci. U.S.A. 115 (2018) 5849–5854.

[49] D. Fan, L. Yang, Z. Wang, M. S. Triantafyllou, G. E. Karniadakis, Reinforcement learning for bluff body active flow control in experiments and simulations, Proc. Natl. Acad. Sci. U.S.A. 117 (2020) 26091–26098.

[50] X. Yan, J. Zhu, M. Kuang, X. Wang, Aerodynamic shape optimization using a novel optimizer based on machine learning techniques, Aerosp. Sci. Technol. 86 (2019) 826–835.

[51] X. Hui, H. Wang, W. Li, J. Bai, F. Qin, G. He, Multi-object aerodynamic design optimization using deep reinforcement learning, AIP Advances 11 (2021) 085311.

[52] R. Li, Y. Zhang, H. Chen, Learning the aerodynamic design of supercritical airfoils through deep reinforcement learning, AIAA J. 59 (2021) 3988–4001.

[53] S. Qin, S. Wang, L. Wang, C. Wang, G. Sun, Y. Zhong, Multi-objective optimization of cascade blade profile based on reinforcement learning, Appl. Sci. 11 (2021) 106.

[54] E. Hachem, H. Ghraieb, J. Viquerat, A. Larcher, P. Meliga, Deep reinforcement learning for the control of conjugate heat transfer, J. Comput. Phys. 436 (2021) 110317.

[55] J. Viquerat, R. Duvigneau, P. Meliga, A. Kuhnle, E. Hachem, Policy-based optimization: single-step policy gradient method seen as an evolution strategy, arXiv preprint arXiv:2104.06175 (2021).

[56] R. Rebonato, P. Jäckel, The most general methodology to create a valid correlation matrix for risk management and option pricing purposes, Available at SSRN 1969689 135 (2011).

[57] K. Numpacharoen, A. Atsawarungruangkit, Generating correlation matrices based on the boundaries of their coefficients, PLoS One 7 (2012) e48902.

[58] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, A. Shah, Adam: a method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[59] T. Coupez, E. Hachem, Solution of high-reynolds incompressible flow with stabilized finite element and adaptive anisotropic meshing, Comput. Methods Appl. Mech. Engrg. 267 (2013) 65–85.

[60] T. J. R. Hughes, G. R. Feijóo, L. Mazzei, J.-B. Quincy, The variational multiscale method - a paradigm for computational mechanics, Comput. Methods Appl. Mech. Engrg. 166 (1998) 3–24.

[61] R. Codina, Stabilization of incompressibility and convection through orthogonal sub-scales in finite element methods, Comput. Methods Appl. Mech. Engrg. 190 (2000) 1579–1599.

[62] Y. Bazilevs, V. M. Calo, J. A. Cottrell, T. J. R. Hughes, A. Reali, G. Scovazzi, Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows, Comput. Methods Appl. Mech. Engrg. 197 (2007) 173–201.

[63] S. R. Allmaras, F. T. Johnson, P. R. Spalart, Modifications and clarifications for the implementation of the Spalart–Allmaras turbulence model, in: Procs. of the 7th International Conference on Computational Fluid Dynamics, 2012.

[64] R. Codina, Comparison of some finite element methods for solving the diffusion-convection-reaction equation, Comput. Methods Appl. Mech. Engrg. 156 (1998) 185–210.

[65] S. Badia, R. Codina, Analysis of a stabilized finite element approximation of the transient convection-diffusion equation using an ALE framework, SIAM J. Numer. Anal. 44 (2006) 2159–2197.

[66] J. Viquerat, E. Hachem, A supervised neural network for drag prediction of arbitrary 2d shapes in laminar flows at low reynolds number, Comp. Fluids 210 (2020) 104645.

[67] J. Bruchon, H. Digonnet, T. Coupez, Using a signed distance function for the simulation of metal forming processes: formulation of the contact condition and mesh adaptation, Int. J. Numer. Meth. Eng. 78 (2004) 980–1008.

[68] C. Gruau, T. Coupez, 3D tetrahedral, unstructured and anisotropic mesh generation with adaptation to natural and multidomain metric, Comput. Methods Appl. Mech. Engrg. 194 (2005) 4951–4976.

[69] E. Hachem, B. Rivaux, T. Kloczko, H. Digonnet, T. Coupez, Stabilized finite element method for incompressible flows with high Reynolds number, J. Comput. Phys. 229 (23) (2010) 8643–8665.

[70] T. Coupez, G. Jannoun, N. Nassif, H. C. Nguyen, H. Digonnet, E. Hachem, Adaptive time-step with anisotropic meshing for incompressible flows, J. Comput. Phys. 241 (2013) 195–211.

[71] J. Sari, F. Cremonesi, M. Khalloufi, F. Cauneau, P. Meliga, Y. Mesri, E. Hachem, Anisotropic adaptive stabilized finite element solver for rans models, Int. J. Numer. Meth. Fl. 86 (2018) 717–736.

[72] G. Guiza, A. Larcher, A. Goetz, L. Billon, P. Meliga, E. Hachem, Anisotropic boundary layer mesh generation for reliable 3D unsteady RANS simulations, Finite Elem. Anal. Des. 170 (2020) 103345.

[73] E. Hachem, H. Digonnet, E. Massoni, T. Coupez, Immersed volume method for solving natural convection, conduction and radiation of a hat-shaped disk inside a 3d enclosure, Int. J. Numer. Method H. 22 (2012) 718–741.

[74] E. Hachem, S. Feghali, R. Codina, T. Coupez, Immersed stress method for fluid-structure interaction using anisotropic mesh adaptation, Int. J. Numer. Meth. Eng. 94 (2013) 805–825.

[75] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.

[76] V. John, Parallele Lösung der inkompressiblen Navier–Stokes Gleichungen auf adaptiv verfeinerten Gittern, Ph.D. thesis, Otto-von-Guericke-Universität Magdeburg, Fakultät für Mathematik (1997).

[77] V. John, Reference values for drag and lift of a two-dimensional time-dependent ow around a cylinder, Int. J. Numer. Meth. Fl. 44 (2004) 777–788.

[78] N. Hansen, The CMA Evolution Strategy: a tutorial, arXiv preprint arXiv:1604.00772 (2016).

[79] T. Kondoh, T. Matsumori, A. Kawamoto, Drag minimization and lift maximization in laminar flows via topology optimization employing simple objective function expressions based on body force integration, Structural and Multidisciplinary Optimization 45 (2012) 693–701.

[80] S. Richardson, Optimum profiles in two-dimensional Stokes flow, Proc. R. Soc. A 450 (1995) 603–622.

[81] J.-Y. Andro, G. Dergham, R. Godoy-Diana, L. Jacquin, D. Sipp, Conditions critiques de déclenchement du lâcher tourbillonnaire au cours du vol des insectes, in: Procs. of the 19ème Congrès Français de Mécanique, 2009.

[82] S. Sunada, T. Yasuda, K. Yasuda, K. Kawachi, Comparison of wing characteristics at an ultralow Reynolds number, J. Aircr. 39 (2002) 331–338.

[83] D. Funda Kurtulus, Vortex flow aerodynamics behind a symmetric airfoil at low angles of attack and reynolds numbers, Int. J. Micro Air Veh. 13 (2021) 17568293211055653.

[84] S. Wang, Y. Zhou, M. Mahbub Alam, H. Yang, Turbulent intensity and reynolds number effects on an airfoil at low reynolds numbers, Phys. Fluids 26 (2014) 115107.

[85] M. Breuer, Large eddy simulation of the subcritical flow past a circular cylinder: Numerical and modeling aspects, Int. J. Numer. Meth. Fl. 28 (1998) 1281–1302.

[86] B. Kulfan, J. Bussoletti, "Fundamental" parameteric geometry representations for aircraft component shapes, in: Procs. of the 11th AIAA/ISSMO multidisciplinary analysis and optimization conference, 2006, p. 6948.

[87] K. Zhang, S. Hayostek, M. Amitay, W. He, V. Theofilis, K. Taira, On the formation of three-dimensional separated flows over wings under tip effects, J. Fluid Mech. 895 (2020).