doi: 10.82124/CEAS-GNC-2026-051 CEAS-GNC-2026-051

AI-based Design of Low Reynolds Number Propellers

Gabriele Salomone PhD Candidate, University of Naples "Federico II", Naples, Italy.
gabriele.salomone@unina.it

Gerardo Paolillo Researcher, University of Naples "Federico II", Naples, Italy.
gerardo.paolillo@unina.it

Tommaso Astarita Full Professor, University of Naples "Federico II", Naples, Italy.
astarita@unina.it

Gennaro Cardone Full Professor, University of Naples "Federico II", Naples, Italy.
gennaro.cardone@unina.it

Carlo Salvatore Greco Associate Professor, University of Naples "Federico II", Naples, Italy.
carlosalvatore.greco@unina.it

ABSTRACT

The present work investigates the application of Artificial Intelligence-based optimization techniques to the aerodynamic design of small-scale propellers operating at low Reynolds numbers. A Single-Step Deep Reinforcement Learning (SDRL) algorithm is implemented to optimize propeller geometry in terms of chord and twist distributions and airfoil shape. The performance of the generated geometries is assessed using Blade Element Momentum Theory (BEMT), coupled with NeuralFoil for fast and reliable aerodynamic coefficients evaluation. In the two optimization runs, the rewards identifying the quality metric are set equal to thrust and efficiency, respectively. The results prove that the intelligent agent is capable of autonomously identifying valid design solutions in a complex design space, without prior aerodynamic knowledge.

Keywords: Deep Reinforcement Learning, Aerodynamic Optimization, Low-Reynolds-Number Propellers, Blade Element Momentum Theory, Airfoil Design

Nomenclature

α	=	angle of attack [rad]
ϕ	=	inflow angle [rad]
μ	=	weighting factor in SDRL optimization [-]
Ω	=	angular velocity [rad/s]
π_Θ	=	policy function
ρ	=	air density [kg m^{-3}]
Θ	=	policy parameters
θ	=	pitch angle [rad]
a, a'	=	axial and tangential induction factors [-]

$\mathbf{a_t}$ = action at time step t
B = number of blades [-]
c = local chord length [m]
C_d = drag coefficient [-]
C_l = lift coefficient [-]
C_n = normal force coefficient [-]
C_P = power coefficient [-]
C_T = thrust coefficient [-]
C_t = tangential force coefficient [-]
D = propeller diameter [m]
F = Prandtl loss factor including tip-loss and hub-loss [-]
J = advance ratio [-]
K = residual of BEMT equation [-]
N = number of discretized spanwise stations [-]
P = power [W]
Q = torque [N m]
r = radial position [m]
r_t = reward at time step t
R = propeller radius [m]
Re_c = chord-based Reynolds number [-]
$\mathbf{s_0}$ = initial state (SDRL)
$\mathbf{s_t}$ = state at time step t
T = thrust [N]
V_∞ = freestream velocity [m/s]
V_x = axial velocity, $V_x = V_\infty(1 + a)$ [m/s]
V_θ = tangential velocity, $V_\theta = \Omega r(1 - a')$ [m/s]
W = relative velocity magnitude, $W = \sqrt{V_x^2 + V_\theta^2}$ [m/s]
BEMT = Blade Element Momentum Theory
LSB = laminar separation bubble [-]
PPO = Proximal Policy Optimization
RL = Reinforcement Learning
(S)DRL = (Single-Step) Deep Reinforcement Learning

1 Background

Unmanned Aerial Vehicles (UAVs), often referred to as drones, have rapidly evolved into versatile platforms with applications spanning civilian, industrial, and defense sectors. Floreano and Wood [1] state that recent advances in miniaturized electronics, low-cost sensors, and efficient control algorithms have contributed to the rise of small-scale autonomous drones capable of operating in both natural and urban environments for surveillance, inspection, logistics, and infrastructure monitoring. In smart city contexts, drones can integrate with Internet of Things (IoT) infrastructures to enhance security and provide real-time situational awareness [2]. Alternatively, these systems can offer flexibility and the ability to operate in areas that may be dangerous or inaccessible for humans [3, 4].

In order to fulfill these mission requirements, UAVs mount propellers that typically operate in chord-based Reynolds number regimes of $Re_c = 10^4 - 10^5$. In this low-Reynolds range, phenomena , such as Laminar Separation Bubbles (LSB), greatly influence aerodynamic behavior and performance. For this reason, a fair evaluation of propeller performance in small UAVs is strongly conditioned by the solver capability of predicting the fluid dynamic phenomena. At the same time, a trade-off between

computational cost and fidelity must be taken into account. In the low-Re regime, RANS solvers have the severe limitation of assuming fully turbulent flow. While transition-sensitive turbulence models, such as $k - k_l - \omega$ or $\gamma - Re_\theta$, have been developed [5–7], the quality of their predictions seems to be linked to the calibration of their parameters, which need to be tuned for the specific flow problem.

These challenges and the underlying RANS assumptions can lead to significant errors in estimating thrust, torque, and efficiency, primarily due to the neglect of LSBs, which makes low-fidelity tools, such as Blade Element Momentum Theory (BEMT) sometimes preferred in low-Re range [8, 9]. To address these limitations, the present study proposes an optimization framework based on Single-Step Deep Reinforcement Learning (SDRL), in which propeller designs are evaluated using a BEMT model.

Inoltre aggiungerei una breve introduzione a ciò che verrà fatto nel lavoro (buttata giù con ChatGPT, se ritieni opportuno modifica)

In this framework, the agent modifies chord and twist distributions as well as airfoil geometry, aiming to optimize propeller performance under low-Reynolds-number conditions. Two optimization scenarios are investigated, where the reward function is defined either in terms of thrust or propulsive efficiency. The resulting optimal configurations are then analyzed and validated a posteriori using the QBlade software to assess the reliability of the proposed framework.

2 Methodology

The BEMT approach provides a practical framework for analyzing the performance of propellers and rotors, combining the actuator disk approach of Rankine–Froude momentum theory with the sectional analysis of blade element theory. For the present implementation, a formulation based on the root-finding of the inflow angle ϕ is employed.

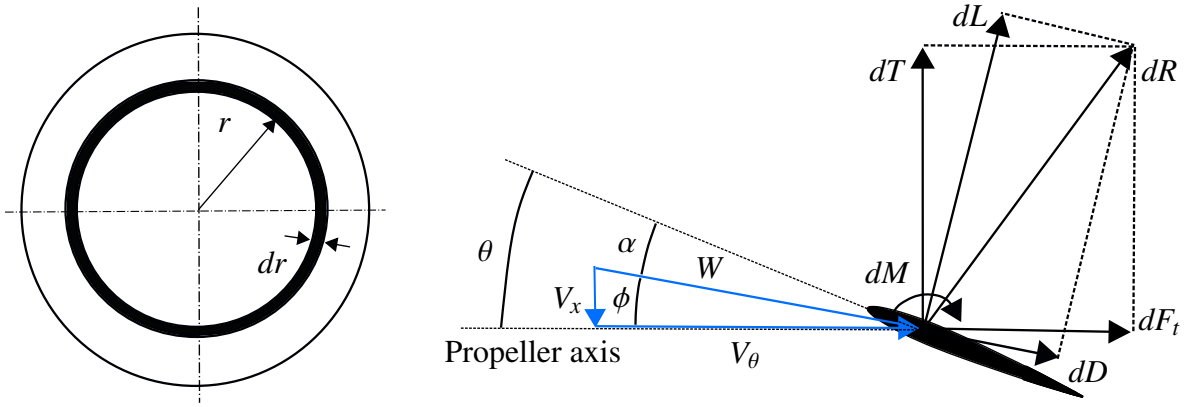

Fig. 1 Scheme of the velocity field induced by an actuator disk in hovering conditions.

In the momentum formulation, the rotor is modeled as an actuator disk (left panel in Figure 1), and the axial and tangential loads are obtained by applying conservation of linear and angular momentum to annular control volumes. For a rotor annulus of radius r and thickness dr, the corresponding differential thrust and torque are written as

$$\mathrm{d}T = 4\pi\rho\, F\, r\, V_\infty^2\, a(1+a)\,\mathrm{d}r, \tag{1}$$

$$\mathrm{d}Q = 4\pi\rho\, F\, r^3\, V_\infty\, \Omega\, a'(1+a)\,\mathrm{d}r, \tag{2}$$

where a and a' denote the axial and tangential induction factors, respectively, and F is the Prandtl loss factor [10]. In parallel, blade element theory provides the sectional aerodynamic forces by discretizing the blade into radial stations and evaluating lift and drag from two-dimensional airfoil data and by projecting these components onto the axial and tangential directions to obtain the contribution to thrust and torque (right panel in Figure 1):

$$\mathrm{d}T = B\left(\frac{1}{2}\rho W^2 c C_n\right)\mathrm{d}r, \tag{3}$$

$$\mathrm{d}Q = B\left(\frac{1}{2}\rho W^2 c C_t\right) r\,\mathrm{d}r, \tag{4}$$

where B is the number of blades, W is the local relative velocity, c the local chord, and C_n and C_t are the non-dimensional coefficients of the force components dT and dF_t, obtained from the lift and drag coefficients as a function of local inflow angle:

$$C_n = C_l \cos\phi - C_d \sin\phi, \tag{5}$$

$$C_t = C_l \sin\phi + C_d \cos\phi. \tag{6}$$

The coupling between momentum theory and blade element theory is enforced by requiring the two descriptions to provide the same axial and tangential loads at each radial station.

In the present implementation, instead of solving directly for the axial and tangential induction factors, the problem is reformulated in terms of the inflow angle ϕ. Starting from the velocity triangle in Figure 1, the kinematic relation between axial and tangential components of the relative velocity can be written as

$$\tan\phi = \frac{V_x}{V_\theta} = \frac{V_\infty(1+a)}{\Omega r(1-a')}. \tag{7}$$

Rearranging this expression leads to a compatibility condition between the local flow angle and the induction factors from which a residual function can be defined:

$$K(\phi) = \frac{\sin\phi}{1+a(\phi)} - \frac{V_\infty}{\Omega r}\frac{\cos\phi}{1-a'(\phi)}. \tag{8}$$

where the induction factors are expressed as a function of the inflow angle from equating thrust and torque contributions from the two theories. The solutions consists in finding the root-ϕ^* for which the condition $K(\phi^*) = 0$ is met.

The elemental lift and drag are evaluated, in the present work, through NeuralFoil [11], a Neural Network-based model that has been trained on a large database of XFoil [12] simulations and other CFD results. The way it works is by computing a statistically approximated prediction of the aerodynamic coefficients of a given profile, instead of computing the two-dimensional solution with a panel method formulation, differently from other common methods, such as XFoil. The use of NeuralFoil is considered more appropriate as it ensures fast outputs, since its computational speed is much higher (on the order of milliseconds per query) than XFoil, making it much suitable for optimization contexts.

Reinforcement Learning (RL) is a subset of machine learning in which an intelligent agent learns to solve optimization problems by earning rewards through trial and-error interaction with an environment. RL consists of training a software agent to pursue actions that maximize a previously established metric of cumulative reward [13]: the RL agent selects an action based on a certain policy and state, and sends the action to an environment, that is, the design space of the propellers. After the action is taken, the RL agent receives a reward from the environment based on the quality of the individual according to the metric and the agent updates the policy parameters based on the taken action and the reward received.

Deep reinforcement learning (DRL) combines RL and deep neural networks, i.e., collections of connected units or artificial neurons, that can be trained to arbitrarily well approximate the mapping function between an input and an output (in this case, an optimal policy) [14]. SDRL is a "lightweight" degenerate version of DRL in which single-step episodes are observed by the intelligent agent, meaning that rewards are awarded after "displacing" the state from a constant baseline one $\mathbf{s_0}$. The agent observes the state $\mathbf{s_{t-1}}$ and uses a policy network π_Θ parameterized with Θ_t to represent a density function from which the action $\mathbf{a_t}$ is extracted. The state $\mathbf{s_t}$ is obtained by scaling the action by an array of multipliers μ, whose components may be also tuned to different values for different components of the state $\mathbf{s_t}$:

$$\mathbf{s_t} = \mathbf{s_0} + \mu \cdot \mathbf{I} \cdot \mathbf{a_t^T} \tag{9}$$

The state $\mathbf{s_t}$ is then mapped onto a physical range in order to obtain a corresponding propeller geometry, which is subsequently awarded with the reward r_t. The intelligent agent iteratively updates the parameters Θ in order to maximize the single-step reward. In literature studies, SDRL found applications in airfoil shape optimization [15, 16]. The present implementation is carried out using the *Proximal Policy Optimization (PPO)* model from the Python library *Stable-Baselines3*, a set of reliable, easy-to-use and consistent Reinforcement Learning (RL) algorithms in PyTorch [17]. A compatible custom environment for the intelligent agent was built to modify a set of design variables that a `geometry` class, created for this purpose, would use to reconstruct a three-dimensional propeller blade.

In the present work, the propeller blade is discretized in N stations. The design variables are a set of PARSEC parameters for the definition of a single airfoil shape, kept constant along the span, and the distributions of pitch angle and chord length along the span. In order to reduce the dimensionality of the design variables, while maintaining a natural planform with no geometric discontinuity, the twist angle and chord length along the blade span are parametrized using Bézier curves, as previously proposed by Wu et al. [18]. The geometry parametrization is summarized in Fig 2.

The pitch angle distribution is described by a third-order Bézier curve, and the chord length distribution is described by a seventh-order curve, as follows:

$$\begin{cases} \theta(r/R) = \theta_1(1-r/R)^3 + 3\theta_2\, r/R\,(1-r/R)^2 + 3\theta_3\,(r/R)^2(1-r/R) + \theta_4\,(r/R)^3, \\ c(r/R) = c_1(1-r/R)^7 + 7c_2\, r/R\,(1-r/R)^6 + 21c_3\,(r/R)^2\,(1-r/R)^5 + 35c_4\,(r/R)^3\,(1-r/R)^4 \\ \qquad\quad + 35c_5\,(r/R)^4\,(1-r/R)^3 + 21c_6\,(r/R)^5\,(1-r/R)^2 + 7c_7\,(r/R)^6\,(1-r/R) + c_8(r/R)^7, \end{cases} \tag{10}$$

Here θ_i $(i = 1, 2, \ldots, 4)$ and c_i $(i = 1, 2, \ldots, 8)$ are the design variables that control the pitch angle and chord length distributions.

Regarding airfoil design, the PARSEC parameterization is a way to fully describe airfoil shapes with a limited set of geometrical parameters that have a precise physical meaning [19], as shown in Figure 2.

The environment with which the agent interacts is characterized by an observation space and an action space:

- The **observation space** defines what the agent perceives about the environment at each step: it contains the state information which serves as input to the current policy. In this work, the state of the environment consists of 11 PARSEC parameters, 4 variables for the twist Bézier curve and 8 variables for the chord Bézier curve, so the observation space dimension is set to 23.
- The **action space**, on the other hand, defines what the agent can do to modify the environment: it represents the set of possible shifts $\mathbf{\Delta s}$ from the initial state $\mathbf{s_0}$. In this case, the action space

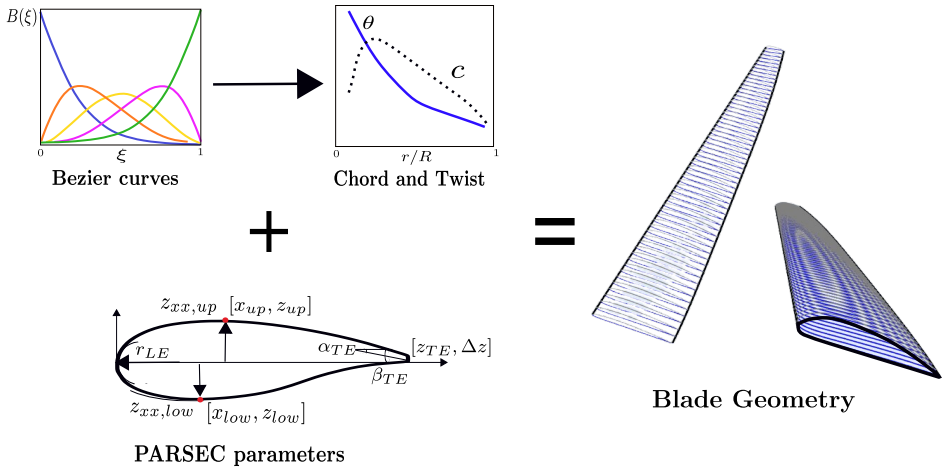

Fig. 2 Schematic of the blade parameterization framework.

dimension is lower than the observation space, as some of the PARSEC parameters are excluded from the optimization and kept constant.

The state $\mathbf{s_t}$, belonging to the **observation space**, is normalized within the interval $[0, 1]$, as each group of components is scaled between given minimum and maximum values. The bounds for the PARSEC parameters are determined by analyzing their scatter in a database of NACA airfoils so as to ensure physical feasibility of the generated airfoils. The twist and chord parameters are normalized by using prescribed $[\theta_{\min}, \theta_{\max}]$ and $[c_{\min}, c_{\max}]$, respectively.

The action space is defined over the interval [-9,9], such that each component of the action at in Eq. 9 lies within this range. The action is mapped to the corresponding physical variation through the multipliers μ. Following the state update, $\mathbf{s_t}$ is clipped to remain within the admissible bounds and subsequently de-normalized to recover the corresponding physical parameters.

To ensure geometric feasibility, penalty terms are introduced whenever at least one radial station violates the prescribed bounds in terms of local chord or twist. In addition, the airfoil shape is subjected to consistency checks, such as the absence of intersections between the upper and lower surfaces. Any violation of these constraints results in a penalized reward, set to $r_t = -2$, independently of the aerodynamic performance predicted by the BEMT solver.

The initial PARSEC parameter set corresponds to a NACA 4412 airfoil, while the initial chord and twist distributions are defined according to the specific optimization setup and may vary depending on the problem under consideration.

3 Results

The results of optimizations for thrust T and efficiency T/P are here presented. The performance of the generated individuals is displayed in terms of thrust coefficient C_T, the power coefficient C_P, and their ratio C_T/C_P as a function of the number of episodes. The coefficients were defined as follows:

$$C_T = \frac{T}{\rho\, n^2\, D^4} \tag{11}$$

$$C_P = \frac{2\pi\, n\, Q}{\rho\, n^3\, D^5} = \frac{P}{\rho\, n^3\, D^5} \tag{12}$$

V_∞	RPM	J	c_{min}	c_{max}	θ_{min}	θ_{max}	μ_θ	μ_c
$8\,\mathrm{m\,s^{-1}}$	3000	0.32	0.002 m	0.05 m	0°	40°	0.05	0.1

Table 1 Thrust-based optimization parameters.

V_∞	RPM	J	c_{min}	c_{max}	θ_{min}	θ_{max}	μ
$8\,\mathrm{m\,s^{-1}}$	3000	0.26	0.002 m	0.026 m	0°	40°	0.11

Table 2 Efficiency-based optimization parameters.

where T is the thrust, Q is the torque, P is the power spent to sustain the rotation, D is the propeller disk diameter and $n = \mathrm{RPM}/60$ is the rotational velocity of the propeller expressed in revolutions-per-second.

4 Thrust-based Optimization

The first optimization is performed by defining the reward as the thrust coefficient C_T, scaled by a factor of 0.15 in order to obtain values of order $O(1)$, as commonly recommended when employing *(PPO)* [17]. No additional constraints on power consumption or propulsive efficiency are imposed in this case. The main flow conditions and optimization parameters are summarized in Table 1.

The initial state $\mathbf{s_0}$ corresponds to a propeller characterized by a monotonically decreasing twist distribution, defined by the parameters $\theta_1 = 30$, $\theta_2 = 25$, $\theta_3 = 10$, $\theta_4 = 5$, and a uniform chord distribution with parameters $c_1 = c_2 = \ldots c_8 = 0.025$. Throughout the optimization, the airfoil geometry is kept fixed and corresponds to a NACA 4412 profile, represented through a PARSEC parametrization. Moreover, the multipliers were set to $\mu_c = 0.1$ for the chord distribution parameters and to $\mu_\theta = 0.05$ for the twist distribution parameters. The purpose of this first test case is to assess and validate the capability of the proposed framework to converge to an optimum.

The evolution of C_T is reported in Figure 3, where a clear convergence trend is observed after approximately 10 000 episodes. Beyond this point, the thrust coefficient stabilizes at a value roughly 7 times the initial one. Figure 4 reports the history of the geometric modification through twist and chord distributions parameters. The first two twist parameters rapidly reach their upper bounds, indicating that higher pitch angles are consistently favored to increase thrust. The remaining parameters, θ_3 and θ_4, exhibit an initial exploration phase before converging to intermediate values of approximately 0.75 and 0.35, respectively.

The most notable outcome concerns the chord distribution, as all eight chord parameters converge to their upper bounds. This behavior is expected, as increasing the blade chord directly increases the generated thrust. In the absence of constraints on efficiency or power, the optimization naturally saturates the chord distribution.

5 Efficiency-based Optimization

In the second optimization run, the reward function is defined to be proportional to the efficiency ratio C_T/C_P. In addition, a minimum thrust constraint is imposed, requiring $C_{T,\mathrm{min}} = 0.08$. This condition is introduced to prevent the algorithm from converging toward configurations characterized by very low thrust and torque levels, resulting in little practical relevance. The main flow conditions and optimization parameters are reported in Table 2, while the initial state $\mathbf{s_0}$ is identical to that adopted in the thrust-based optimization. In this optimization, the multipliers are constant for all the components of $\mathbf{s_t}$ and set equal to $\mu = 0.11$.

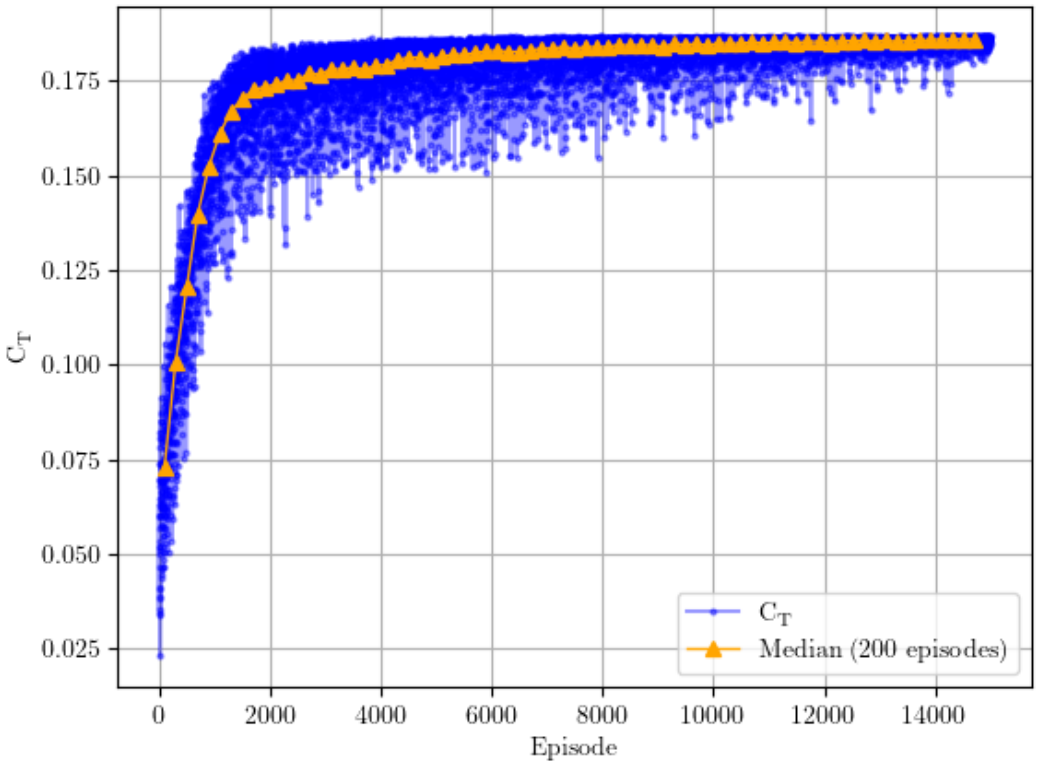

Fig. 3 Thrust-based optimization: training history of C_T showing episode values (blue) and 200-episode median (orange).

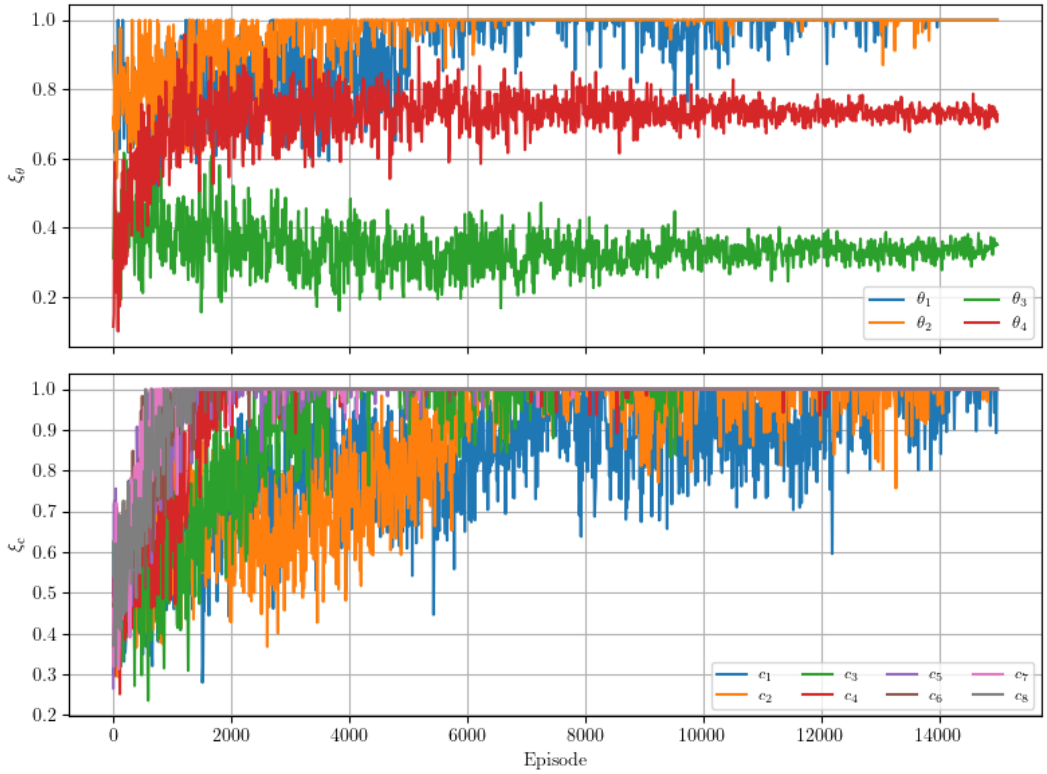

Fig. 4 Thrust-based optimization: training history of the Bézier control parameters. Top panel: twist parameters θ_1-θ_4. Bottom panel: chord parameters c_1-c_8.

The evolution of the propulsive efficiency, defined as $\eta = J\,C_T/C_P$, is shown in Figure 5. A clear improvement is observed, as the efficiency more than doubles with respect to the initial configuration. At

the same time, as reported in Figure 6, this gain is achieved at the expense of thrust, which stabilizes at significantly lower levels than obtained in the thrust-based optimization, and is approximately 50% lower than the maximum values achieved with the NACA 4412 airfoil.

The resulting geometric adaptation is less straightforward compared to the thrust-driven case. As reported in Figure 7, the twist parameters (except for θ_1) cluster around values close to 0.3, yielding a smooth, monotonically decreasing twist distribution from root to tip (Figure 8). Regarding the chord distribution, the optimization converges toward a non-uniform profile: parameters c_4 and c_5 reach their upper bounds, while the remaining coefficients stabilize within the range 0.5 to 0.8 (Figure 7). As shown in Figure 8, this results in a chord distribution characterized by a pronounced mid-span maximum and a progressive reduction toward the blade tip.

The obtained configuration is consistent with the task required from the optimization, as the estimated spanwise distribution of the angle of attack, displayed in Figure 9, remains rather low throughout the blade length.

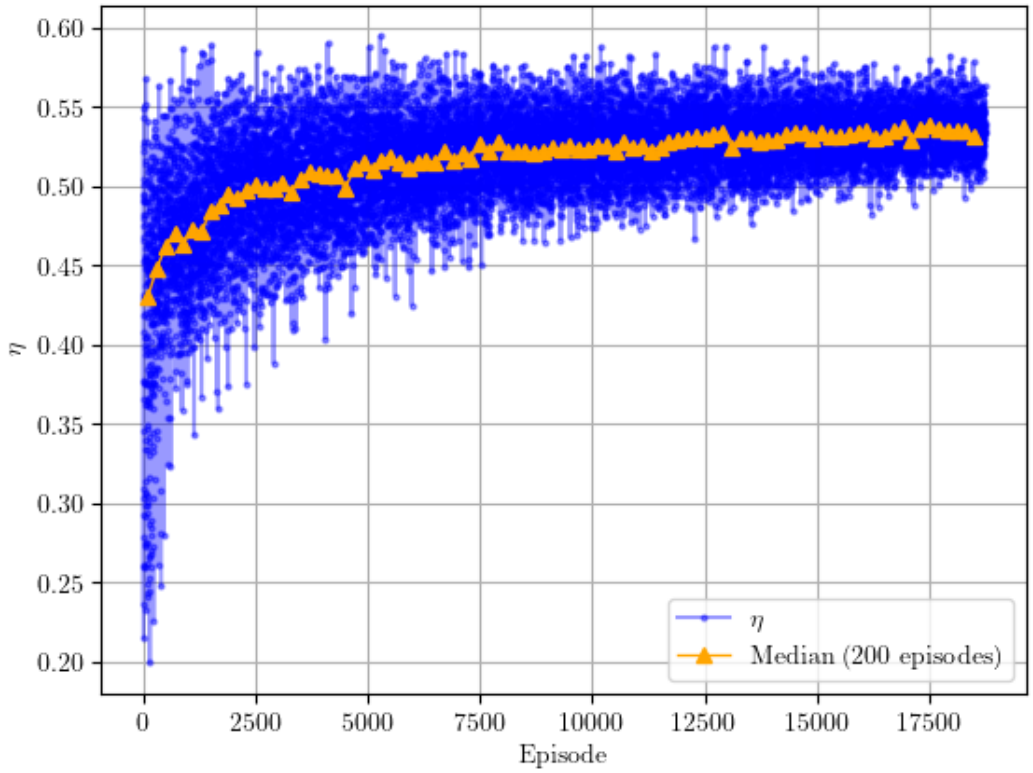

Fig. 5 Efficiency-based optimization: training history of η showing episode values (blue) and 200-episode median (orange)

The optimized airfoil shape is shown in Figure 10. The optimized geometry primarily departs from the baseline NACA 4412 as it appears thinner, generally less cambered and with a streamwise peak location aft-shifted. The two-dimensional aerodynamic polars reported in Figure 11 demonstrate that the optimized airfoil consistently generates lower lift coefficients than the baseline NACA 4412 across the investigated angle-of-attack range. However, in the low-Reynolds numbers regime, a drag reduction can be observed for angles of attack up to approximately 8 °, indicating improved aerodynamic efficiency in this regime.

5.1 Aerodynamic Validation

The aerodynamic performance predicted by the present framework is validated against results obtained by running BEMT simulations on QBlade [20], a free-to-download software widely used for

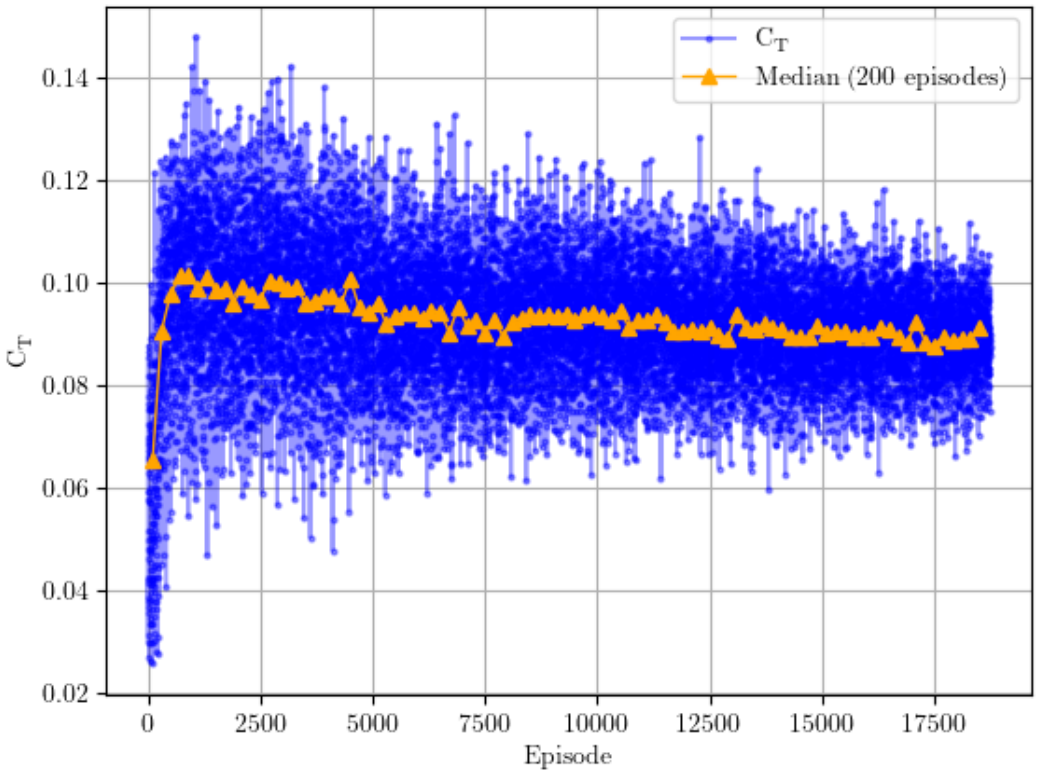

Fig. 6 Efficiency-based optimization: training history of C_T showing episode values (blue) and 200-episode median (orange).

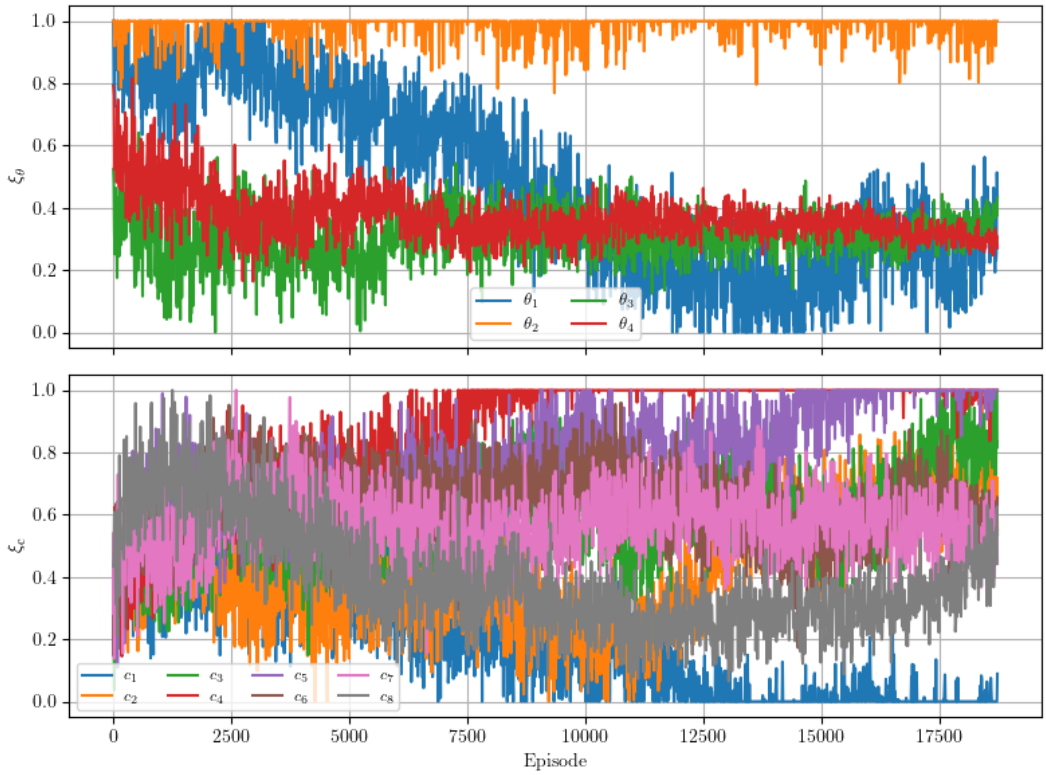

Fig. 7 Efficiency-based optimization: training history of the Bézier control parameters. Top panel: twist parameters θ_1-θ_4. Bottom panel: chord parameters c_1-c_8.

propeller and wind turbines analysis. The comparison is shown in Figure 12. Overall, a satisfactory

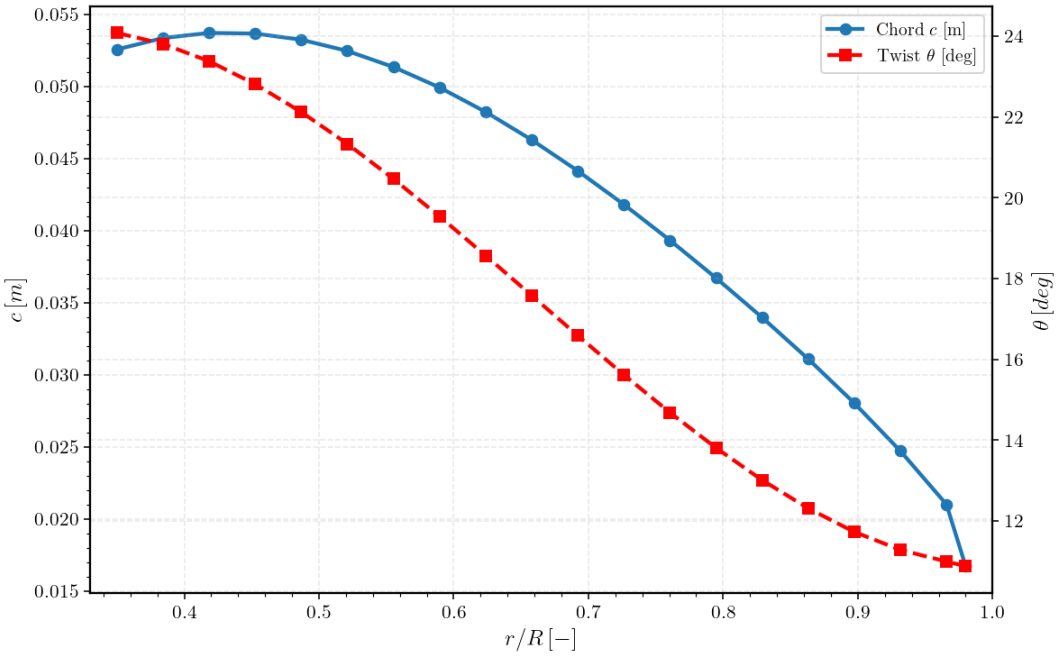

Fig. 8 Efficiency-based optimization: spanwise distributions of chord and twist for the optimal blade configuration.

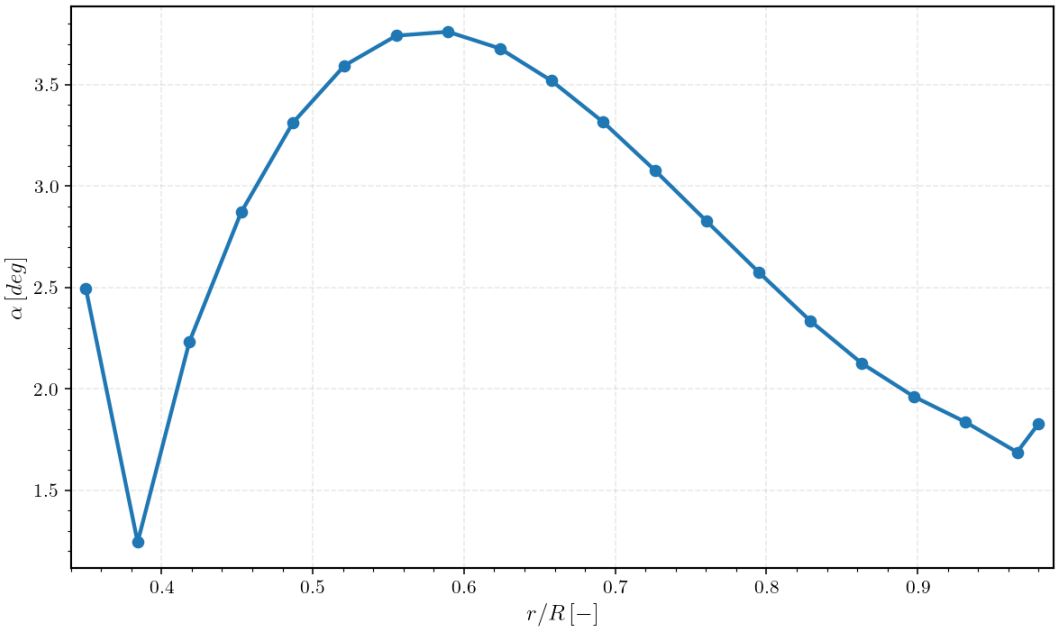

Fig. 9 Efficiency-based optimization: spanwise distribution of the estimated angle of attack for the optimal blade configuration.

agreement is observed across the entire range of advance ratios, both in terms of propulsive efficiency η and thrust coefficient C_T.

The predictions obtained with the developed BEMT solver, whether coupled with NeuralFoil or XFoil aerodinamic data, remain mutually consistent throughout the operating range. When compared to QBlade, both the thrust coefficient and the efficiency exhibit similar trends for $J > 0.25$, with the discrepancies in C_T progressively decreasing as the advance ratio increases.

The remaining differences can be attributed to the automatic implementation of tip-loss and hub-loss corrections, as well as the extrapolation of airfoil polars to 360°, may require further tuning. In contrast,

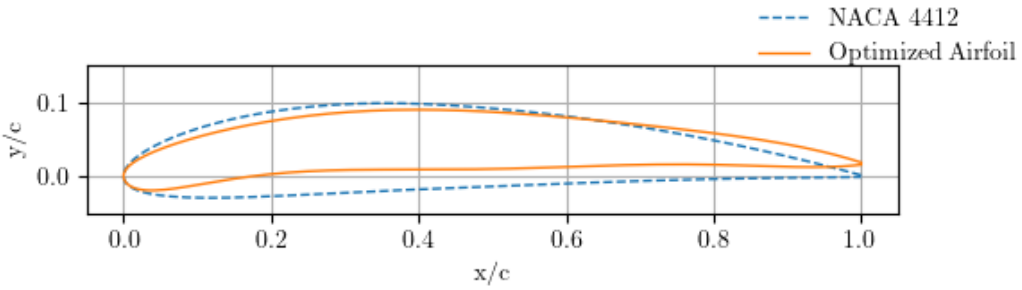

Fig. 10 Efficiency-based optimization: Optimized airfoil vs. NACA 4412.

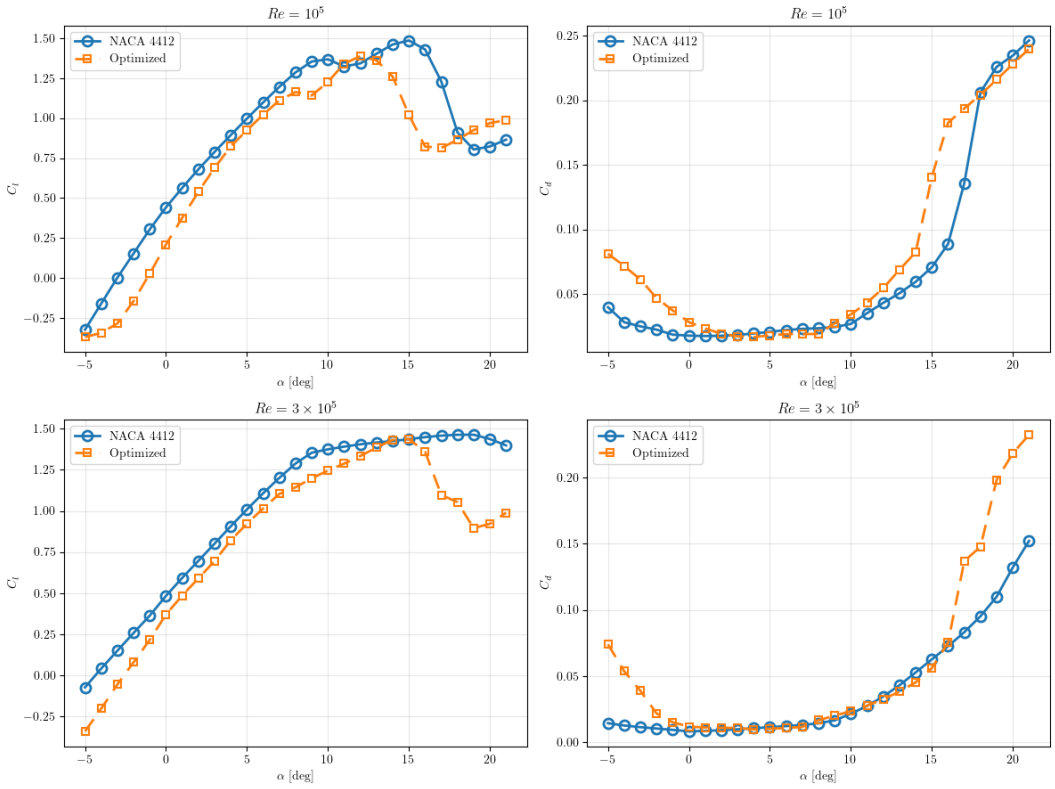

Fig. 11 Efficiency-based optimization: Optimized airfoil polars vs. NACA 4412 at $Re = 1 \times 10^5$ (upper panels) and $Re = 3 \times 10^5$ (lower panels).

QBlade allows for a more refined and often manually adjusted treatment of these effects, which can lead to improved agreement in off-design conditions.

Despite these discrepancies, the comparison demonstrates that the proposed framework provides reliable and consistent predictions of propeller performance, making it suitable for preliminary design and optimization studies. Ongoing efforts are focused on refining the aforementioned modeling aspects to further reduce the gap with higher-fidelity tools.

6 Conclusions

This work demonstrates the potential of a Single-Step Deep Reinforcement Learning (SDRL) framework coupled with a BEMT solver for the aerodynamic design of propellers operating at low Reynolds numbers. The proposed approach enables efficient exploration of the design space while maintaining a limited computational cost. In the thrust-based optimization the algorithm is reasonably fast at identifying that maximizing chord length is the most effective strategy to achieve higher rewards, regardless

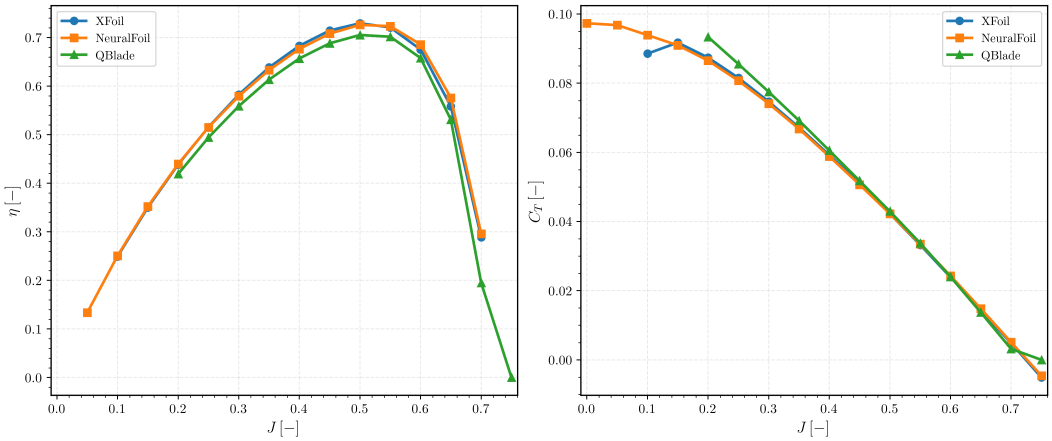

Fig. 12 Aerodynamic validation of the optimal blade design. Efficiency η (left) and thrust coefficient C_T (right) are shown as functions of the advance ratio J, comparing XFoil, NeuralFoil, and QBlade predictions.

of power required to rotate the propeller. The efficiency-based optimization, on the other hand, aimed at minimizing T/P, while respecting the imposed thrust requirement. This is performed by designing an airfoil that results in lower C_d at low incidence and by adjusting the chords and twist distributions so that these favorable features can be exploited. AI-based approaches represent a promising directions in modern engineering and opens up novel possibilities for the design of advanced aerodynamic systems with improved performance. The next steps in the work include a tuning of the BEMT code and experimental verifications by thrust measurements or flow field visualizations that can address the reasons why the optimized configurations result in higher performance. Optimizations that embody high-fidelity aerodynamic methods may also be of great interest.

7 EuroGNC AI policy (new)

Artificial Intelligence tools were **not** used in the development of the concepts covered in the text or for generating images.

References

[1] Dario Floreano and Robert J Wood. Science, technology and the future of small autonomous drones. *Nature*, 521(7553):460–466, 2015.

[2] GN Muchiri and S Kimathi. A review of applications and potential applications of UAV. In *Proceedings of the Sustainable Research and Innovation Conference*, pages 280–283, 2022.

[3] Mostafa Hassanalian and Abdessattar Abdelkefi. Classifications, applications, and design challenges of drones: A review. *Progress in Aerospace Sciences*, 91:99–131, 2017.

[4] Didula Dissanayaka, Thumeera R Wanasinghe, Oscar De Silva, Awantha Jayasiri, and George KI Mann. Review of navigation methods for UAV-based parcel delivery. *IEEE Transactions on Automation Science and Engineering*, 21(1):1068–1082, 2023.

[5] Aniket Aranake, Vinod Lakshminarayan, and Karthik Duraisamy. Assessment of transition model and CFD methodology for wind turbine flows. In *42nd AIAA fluid dynamics conference and exhibit*, page 2720, 2012.

[6] Davide Di Pasquale, Aldo Rona, and S Garrett. A selective review of transition modelling for CFD. In *39th AIAA fluid dynamics conference*, page 3812, 2009.

[7] Florian R Menter, R Langtry, and Stefan Völker. Transition modelling for general purpose CFD codes. *Flow, turbulence and combustion*, 77(1):277–303, 2006.

[8] JCPJ Morgado, R Vizinho, MAR Silvestre, and JC Páscoa. Xfoil vs CFD performance predictions for high lift low reynolds number airfoils. *Aerospace Science and Technology*, 52:207–214, 2016.

[9] Eric Vargas Loureiro, Nicolas Lima Oliveira, Patricia Habib Hallak, Flávia de Souza Bastos, Lucas Machado Rocha, Rafael Grande Pancini Delmonte, and Afonso Celso de Castro Lemonge. Evaluation of low fidelity and CFD methods for the aerodynamic performance of a small propeller. *Aerospace Science and Technology*, 108:106402, 2021.

[10] Gordon J Leishman. *Principles of helicopter aerodynamics with CD extra*. Cambridge university press, 2006.

[11] Peter Sharpe and R John Hansman. Neuralfoil: An airfoil aerodynamics analysis tool using physics-informed machine learning. *arXiv preprint arXiv:2503.16323*, 2025.

[12] Mark Drela. Xfoil: An analysis and design system for low reynolds number airfoils. In *Low Reynolds number aerodynamics: proceedings of the conference notre dame, Indiana, USA, 5–7 June 1989*, pages 1–12. Springer, 1989.

[13] A Lozano-Durán and M Bassenne. Towards model discovery with reinforcement learning. *Annual Research Briefs*, 2019.

[14] Jonathan Viquerat, Régis Duvigneau, Philippe Meliga, Alexander Kuhnle, and Elie Hachem. Policy-based optimization: single-step policy gradient method seen as an evolution strategy. *Neural Computing and Applications*, 35(1):449–467, 2023.

[15] H Ghraieb, J Viquerat, A Larcher, P Meliga, and E Hachem. Single-step deep reinforcement learning for two-and three-dimensional optimal shape design. *AIP Advances*, 12(8), 2022.

[16] P Scavella, G Paolillo, and CS Greco. Deep reinforcement learning-based airfoil design and optimization: An aerodynamic analysis. *Aerospace Science and Technology*, page 110638, 2025.

[17] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[18] Xiaojing Wu, Zijun Zuo, Long Ma, and Weiwei Zhang. Multi-fidelity neural network-based aerodynamic optimization framework for propeller design in electric aircraft. *Aerospace Science and Technology*, 146:108963, 2024.

[19] Pierluigi Della Vecchia, Elia Daniele, and Egidio D Amato. An airfoil shape optimization technique coupling PARSEC parameterization and evolutionary algorithm. *Aerospace Science and Technology*, 32(1):103–110, 2014.

[20] QBlade software website: https://qblade.org/.