Black-Box Online Aerodynamic Performance Optimization for a Seamless Wing with Distributed Morphing

Ruland, Oscar; Mkhoyan, Tigran; De Breuker, Roeland; Wang, Xuerui

DOI
10.2514/1.G006573

Publication date
2023

Document Version
Final published version

Published in
Journal of Guidance, Control, and Dynamics

Citation (APA)
Ruland, O., Mkhoyan, T., De Breuker, R., & Wang, X. (2023). Black-Box Online Aerodynamic Performance Optimization for a Seamless Wing with Distributed Morphing. *Journal of Guidance, Control, and Dynamics*, *46*(3), 560-570. https://doi.org/10.2514/1.G006573

Journal of Guidance, Control, and Dynamics
Vol. 46, No. 3, March 2023

Black-Box Online Aerodynamic Performance Optimization for a Seamless Wing with Distributed Morphing

Oscar Ruland,* Tigran Mkhoyan,† Roeland De Breuker,‡ and Xuerui Wang§
Delft University of Technology, 2629 HS Delft, The Netherlands

https://doi.org/10.2514/1.G006573

I. Introduction

OVER the past century, aircraft have become increasingly more efficient. During the 1960s, improvements in engine technology and wing design lead to significant improvements in aircraft fuel economy. In recent years, this trend of increasing efficiency has started to stagnate. To further reduce both the cost of flying and environmental pollution, more radical departures from the conventional aircraft design are needed. One promising technology is active morphing, which enables shape transformation in-flight [1,2]. The Wright Flyer, the first successful heavier-than-air powered aircraft, relied on twist morphing of its fabric-wrapped flexible wings to achieve roll control [3]. However, as aircraft flew with ever-increasing speeds, higher wing rigidity was required, which made morphing fade out in the 1940s to 2000s. In recent years, morphing has again been made possible by advanced developments in material science such as shape memory alloys, compliant mechanisms, and piezoelectrics [1,4].

The ability to reshape the wing in flight introduces the problem of determining what that shape should be for a wide range of operational conditions. The current method for cruise drag minimization is the scheduling of configuration settings through lookup tables as a function of gross weight, airspeed, and altitude. These lookup tables generally depend on analytical models, validated with wind tunnel or test flight data. However, different operating conditions, aircraft production variances, and repairs can result in uncertainties in the table-lookup method.

Online optimization has the potential to tailor the wing shape to any specific flight condition for achieving the best aerodynamic performance based on in-flight measurements. Much like birds, a smart morphing-wing aircraft could sense its environment and adapt its wings' shape to achieve the best performance in any condition, making it fully mission-adaptive. However, many challenges remain on the path toward operational smart morphing aircraft wings. To begin with, any online optimization method is reliant on the ability to accurately evaluate the aircraft's performance using on-board sensors. Furthermore, only a very limited amount of search space exploration could realistically be afforded on a typical commercial flight. Ideally, a global optimum in the optimization landscape should be found with limited and local explorations.

A real-time adaptive least-squares drag minimization approach has been proposed for the variable camber continuous trailing edge flap (VCCTEF) described in [5,6]. This strategy uses a recursive least-squares algorithm to estimate the derivatives of the aerodynamic coefficients with respect to the system inputs. The optimal wing shape and elevator deflection are then calculated from a constrained optimization problem using the Newton–Raphson method. Improvements to the model excitation method, on-board model, and optimization methods were demonstrated in wind tunnel experiments to achieve up to 9.4% drag reduction on the common research model (CRM) with the VCCTEF at off-design conditions at low subsonic speeds [7]. Simulations have also indicated that a 3.37% drag reduction is achievable on the CRM with a distributed mini-plain flap system at Mach 0.85 [8].

While the coefficients of the linear-in-the-parameters multivariate polynomial model adopted in [7,8] can be estimated with relatively low computational cost, the model is only valid in the local region around the trim condition. This means that in order to perform real-time drag minimization across the entire flight envelope, the model parameters need to be re-identified at every operational point. Moreover, the required model excitation maneuvers that comprise both angle-of-attack and flap deflection inputs would induce undesirable bumpiness, structural loads, and increased fuel consumption. Last but not least, the use of a local model together with a gradient-based optimization method makes the solution prone to converge onto a local optimum. By contrast, a global on-board model, while more difficult to identify online, could allow for continuous drag minimization throughout the flight envelope. Additionally, when paired with a global optimization method, global optima with even better performance could potentially be found.

The main contributions of this paper are the first presentation and demonstration of a novel adaptable in-flight black-box performance optimization strategy for morphing wings. The proposed strategy integrates an online trained global artificial neural network (ANN) surrogate model [9], also referred to as the on-board model, with an evolutionary optimization algorithm [10,11]. The covariance matrix adaptation–evolutionary strategy (CMA–ES) black-box optimization method was adopted because of its robustness to noise, ability to optimize nonconvex and multimodal problems, and desirable global performance [12]. To reduce the time required for optimization and to effectively retain the knowledge gained from historical measurements, an on-board model is adapted online. For the online identification of this on-board model, radial basis function neural networks (RBFNNs) were employed because of their local sensitivity, robustness to noise, and effectiveness on scattered data [13,14]. The integration of these methods allows for the optimization of the morphing wing's shape based on scattered and noisy flight data in real time.

This data-driven approach is more adaptable and potentially able to realize higher performance than conventional shape scheduling by look-up tables. The morphed wing shape could be tailored in-flight to

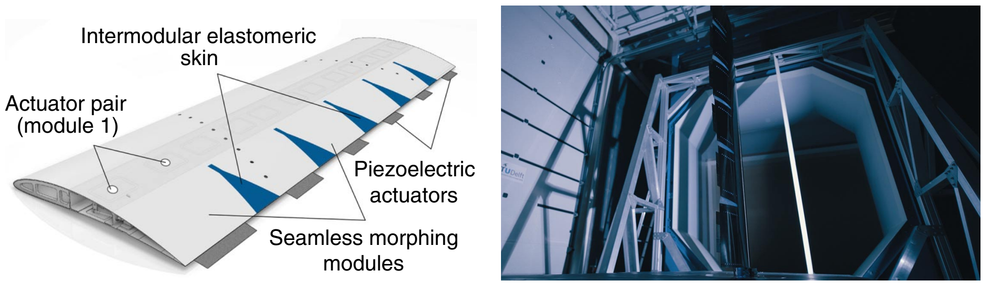
Presented as Paper 2022-1840 at the AIAA Scitech 2022 Forum, San Diego, CA, and Virtual, January 3–7, 2022; received 16 November 2021; revision received 14 September 2022; accepted for publication 19 September 2022; published online 17 October 2022. Copyright © 2022 by Xuerui Wang and Delft University of Technology. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission. All requests for copying and permission to reprint should be submitted to CCC at www.copyright.com; employ the eISSN 1533-3884 to initiate your request. See also AIAA Rights and Permissions www.aiaa.org/randp.

*M.Sc. Student, Department of Control and Operations, Faculty of Aerospace Engineering, Kluyverweg 1; o.l.ruland@student.tudelft.nl.

†Ph.D. Candidate, Department of Aerospace Structures and Materials, Faculty of Aerospace Engineering, Kluyverweg 1; t.mkhoyan@tudelft.nl. Student Member AIAA.

‡Associate Professor, Department of Aerospace Structures and Materials, Faculty of Aerospace Engineering, Kluyverweg 1; r.debreuker@tudelft.nl. Associate Fellow AIAA.

§Assistant Professor, Department of Aerospace Structures and Materials, and Department of Control and Operations, Faculty of Aerospace Engineering, Kluyverweg 1; x.wang-6@tudelft.nl. Member AIAA.



a) Overview of wing components b) Wind tunnel setup

Fig. 1 Overview of the SmartX-Alpha wing demonstrator (0.5 m × 1.8 m).

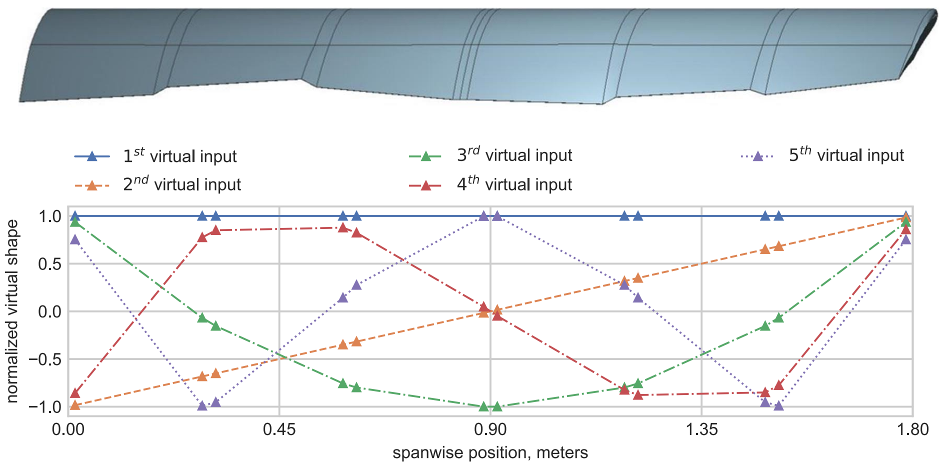


Fig. 2 Virtual shape functions that dictate the amount of camber morphing at each actuator location.

maximize the performance of the particular aircraft under consideration, rather than the performance of a model, built from previous test fight data on a similar aircraft. Moreover, compared to the state-of-the-art local gray-box methods that require additional model excitation maneuvers and re-identifications at each operational condition, the proposed approach retains the information learned in a global radial basis function neural network on-board model such that smooth and direct transitions to well-performing wing shapes can be achieved throughout the entire flight envelope. Furthermore, by integrating a derivative-free evolutionary optimization strategy with a global on-board model, global optima can be found. The proposed method has been validated on a model of a seamless active distributed morphing wing: SmartX-Alpha [15,16] (Fig. 1).¶

The structure of this paper is as follows. The morphing wing system is modeled in Sec. II. The optimization architecture is proposed in Sec. III. In Sec. IV, the simulation results are presented and discussed. Finally, the main conclusions are drawn in Sec. V.

II. System Modeling

A. Virtual Inputs

The morphing wing system consists of 13 inputs: the deflections of the 12 actuators and the wing angle of attack α. However, instead of using the actuator angles θ directly as system inputs, the optimizer and on-board model use a total of five virtual shape functions to describe the wing's shape. The virtual inputs $u_1, \ldots, u_5$ scale the five basis shapes described by the first five Chebyshev polynomials of the first kind, rescaled onto the [0, 1.80] m domain, where 1.8 m is the half-wing span. The spanwise distribution of the local actuator deflection is a linear combination of the virtual inputs and the Chebyshev polynomials T_i as a function of the spanwise location y as stated in Eq. (1).

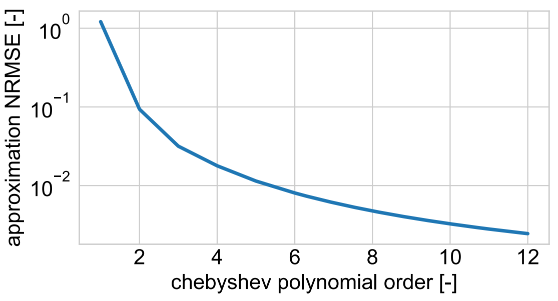
¶The project video can be found via https://www.youtube.com/watch?v=SdagIiYRWyA&t=319s.

$$\theta(y) = \sum_{i=1}^{5} u_i T_i(y) \tag{1}$$

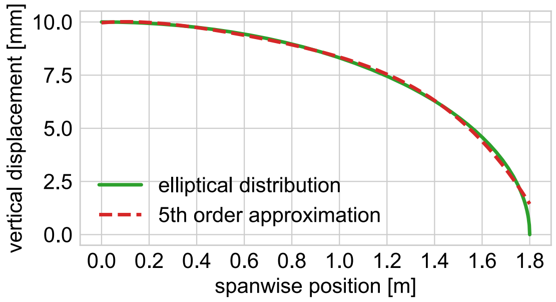
The virtual inputs and their contributions to the actuator deflection at each actuator location are shown in Fig. 2, where the triangular markers indicate the actuator positions. The translation-induced camber morphing mechanisms are modeled as a series of twistable plain flaps, whose local deflections vary linearly between the actuators. The deflection of each actuator is in turn dictated by the virtual inputs.

The virtual shapes reduce the 13-dimensional optimization domain for the real system to a 6-dimensional one for the model, which significantly reduces the computational cost. Moreover, the basis shape functions enforce a certain degree of smoothness in the final morphed wing shape. Their use generally leads to smoother shapes than those resulting from 12 independent actuator deflections as they avoid shapes with large and frequent jumps in spanwise camber.

The choice of the number of virtual functions is determined by the tradeoff between reducing the shape approximation error and reducing the number of measurements required to identify an on-board model. As we show in Fig. 3, increasing this number can indeed reduce the shape approximation normalized root mean square error (NRMSE). However, on the one hand, the reduction rate (slope) decreases as this number increases; on the other hand, the computational load significantly increases as this number increases because 1) the size of the on-board neural network model increases, which required a longer time to train, evaluate, and update, and 2) the search space of the optimization algorithm also increases, requiring longer searching and optimization time as well as more measurements. Based on these reasons, we choose five virtual shape functions in this research, which results in reduced computational loads without compromising the shape smoothness. As shown in Fig. 3, this number can approximate an elliptical distribution well while making the NRMSE below 1.15%.



a) Relation between the NRMSE of a Chebyshev polynomial and the model order



b) Comparison of an elliptical distribution function and its 5th order Chebyshev approximation

Fig. 3 The impacts of the number of virtual functions on the shape approximation error.

B. Aerodynamic Model

The actuator deflections described by the virtual inputs are transformed to local flap deflections to produce the geometry that is to be evaluated by the aerodynamic model. First, the local vertical displacement of the trailing edge z_{te} is computed as $z_{\mathrm{te}} = \theta k_\theta$. Using the digital image correlation measurements of symmetric morphing on SmartX-Alpha [15], k_θ is estimated as 5.6×10^{-4}. The local plain flap deflection angle δ_f is then computed using Eq. (2), where x_{hinge} is the location of the flap hinge as a fraction of the chord length c. Between the actuator locations, where the local flap angle is specified by the virtual inputs, the local flap angle varies linearly.

$$\delta_f = \sin^{-1}\left(\frac{z_{\mathrm{te}}}{c \cdot (1 - x_{\mathrm{hinge}})}\right) \tag{2}$$

The aerodynamic performances of wing shape and angle-of-attack combinations are evaluated using a vortex lattice method (VLM) [17] model implemented in the Aerosandbox python package [18]. This method is used because of its high computational efficiency and scriptability. Since Aerosandbox is a relatively new open-source aerodynamic solver, and only one publication using this package exists in literature [19], its VLM implementation is verified against that of XFLR5 using the geometry of SmartX-Alpha. Figure 4 shows results from the Aerosandbox and XFLR5 VLM solvers, and wind tunnel measurements for constant spanwise actuator angle of −22 deg. It can be observed from Fig. 4 that the outputs of Aerosandbox and XFLR5 VLM have a high consistency.

However, VLM neglects the effects of viscosity and thickness, and can only be used to estimate lift and induced drag. As a result, the models slightly overestimate the lift slope, although their lift predictions remain close to the wind tunnel measurements for the linear part of the lift curve. The drag on the other hand is consistently underestimated due to the lack of viscous drag effects in the model. Furthermore, while asymmetric flap deflections affect the lift-to-induced-drag ratio L/D_i through reshaping of the spanwise lift distribution, constant flap deflections along the wingspan do not affect L/D_i at all. However, in order to optimize the morphing wings aerodynamic efficiency, both the total drag and the effects of flap deflections on the lift-to-drag ratio L/D should be modeled. Therefore, the model is augmented with an estimation of the zero-lift-drag coefficient C_{D_0} and a correction to the Oswald efficiency factor e based on data from a previous wind tunnel campaign with SmartX-Alpha. Furthermore, the use of the corrected model is restricted to the linear part of the lift curve, i.e., $-5.0 < \alpha < 10.0$ deg. Wind tunnel measurements from seven angle-of-attack sweeps at different spanwise constant actuator angles were used to estimate C_{D_0} and e using the least-squares method and Eq. (3).

$$C_D = C_{D0} + C_L^2/(\pi Æ e) \tag{3}$$

where C_L and C_D denote the lift and drag coefficients, respectively. The estimates for C_{D_0} and e were interpolated by first- and second-order polynomials, respectively (Fig. 5). With these corrections and the induced drag from the Aerosandbox model, the total drag is then estimated with Eq. (4):

$$C_D = C_{D_0}(\bar{\delta}_f) + C_{D_i} e_0 / e(\bar{\delta}_f) \tag{4}$$

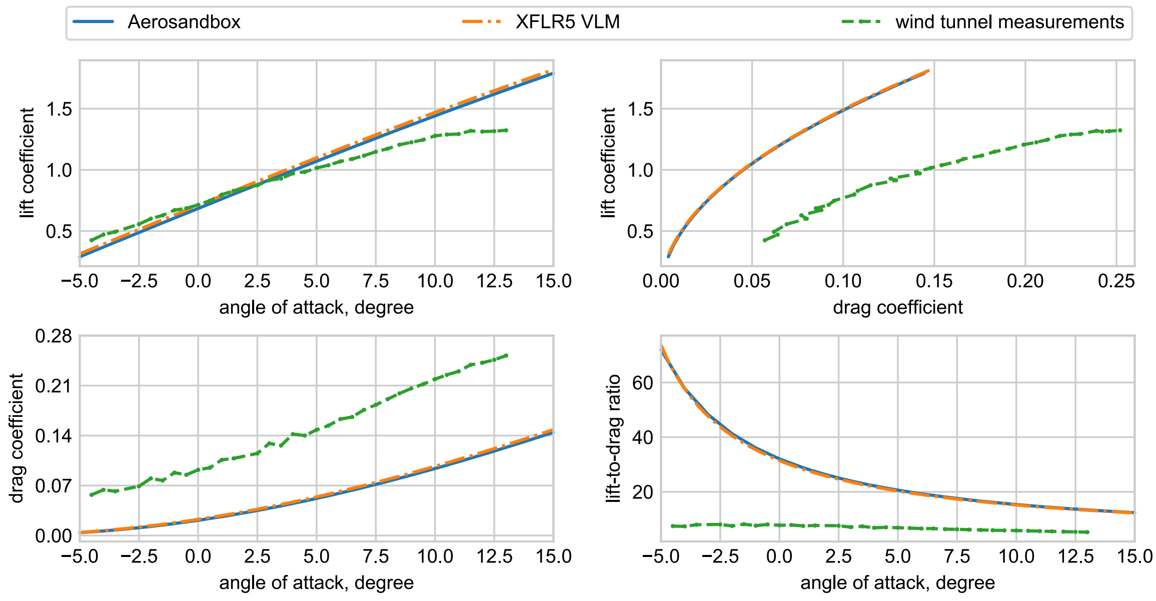


Fig. 4 Comparison of VLM solvers with wind tunnel measurements for a constant actuator angle of −22 deg.

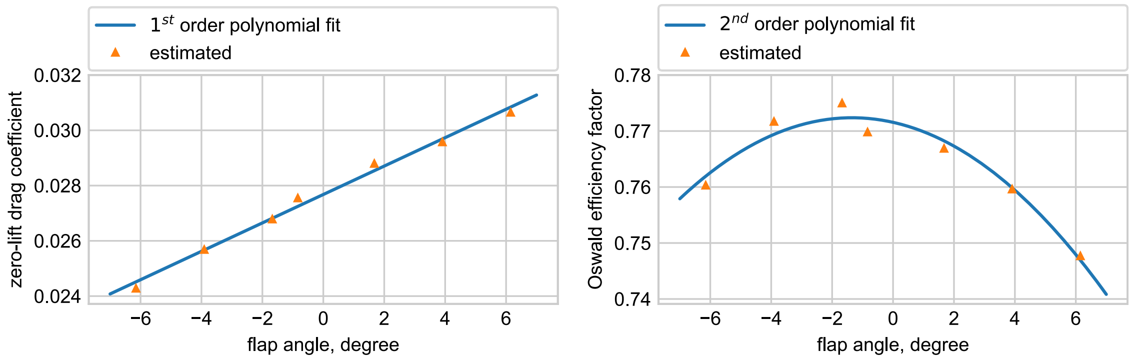


Fig. 5 Two correction functions estimated based on wind tunnel measurements.

in which $\mathcal{R}$ represents the aspect ratio, and $\bar{\delta}_f$ represents the mean flap angle. The Oswald efficiency factor of the constant deflection wing shape from the Aerosandbox model, denoted as e_0, is estimated using the least-squares method with Eq. (3). Using simulated lift and induced drag measurements from an angle-of-attack sweep with the Aerosandbox model, e_0 is estimated as 0.95. For the case of a constant −22 deg actuator angle, the effects of the corrections functions are shown in Fig. 6. Compared with the uncorrected drag polar from Fig. 4, the zero-lift-drag correction yields a result that is much closer to the wind tunnel measurements. However, the drag is still underestimated consistently. After correcting the drag predicted by Aerosandbox with both the zero-lift-drag and the Oswald efficiency corrections, the resulting drag polar closely matches the wind tunnel measurements. Since the corrections were estimated using wind tunnel data, their validity is limited to the wing geometry and flow conditions that these measurements correspond to. Nevertheless, the presented correction method is widely applicable to other cases.

C. Secondary Model

For future real-world operations, the use of white-box aerodynamic models such as the corrected model described above would be limited to training of the on-board model beforehand. In this manner, a priori knowledge about the system is transferred to the on-board model through the network weights. Although these will be adjusted during the online learning process, fewer adjustments are required than would be in the case of learning from scratch. In later stages of the technology, the network weights would hold the knowledge from previous flights, which is superior in quality compared to any model-based predictions.

To demonstrate the ability of the online learning shape optimization procedure to adapt to a change in the system to be optimized, a secondary aerodynamic model is used in the online shape optimization. The secondary model represents a comparable, but yet distinctly different morphing wing system. In this research, this model is comprised of the same wing planform as the nominal model, but with a NACA4312 airfoil instead (the SmartX-Alpha airfoil is NACA6510). As the VLM solver does not model the effects of airfoil thickness, only the maximum camber and location of maximum camber are different between the nominal and secondary models. Because equivalent wind tunnel data for this wing do not exist, the correction function estimation procedure cannot be repeated for the secondary wing model. Instead, the correction functions are altered directly. Therefore, the secondary model does not accurately model the aerodynamics of a known wing anymore. Instead, the secondary model represents the aerodynamics of an unknown wing, which are relatively close to those of the nominal model. The correction functions for both models are shown in Fig. 7.

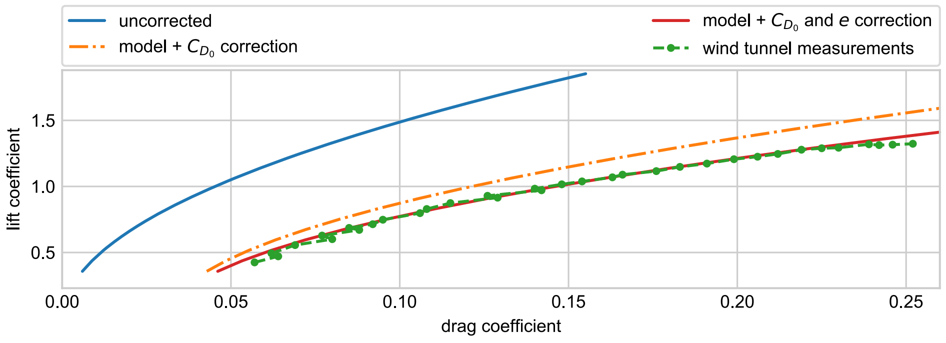


Fig. 6 Drag polar of the corrected aerodynamic model for a constant actuator angle of −22 deg.

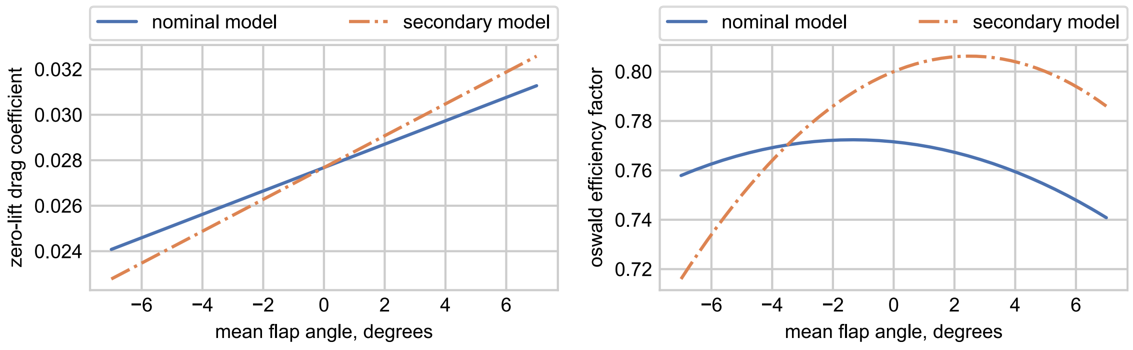


Fig. 7 Two correction functions for the nominal and secondary model.

D. Noise Simulation and Filtering

Real-world measurements were simulated by adding noise to the aerodynamic model outputs. The noise realizations used were derived from noise measurements from a previous wind tunnel experiment. The power spectral density (PSD) of the original noise signal, sampled at 1000 Hz, was approximated by its periodogram. The PSD $\boldsymbol{S}(\boldsymbol{f}_n)$ is sampled at n positive frequencies $\boldsymbol{f}_n = [\Delta_f \quad 2\Delta_f \quad \dots \quad n\Delta_f]^T$.

First, these power spectral densities are converted to amplitudes using $\boldsymbol{A}(\boldsymbol{f}_n) = \sqrt{2\boldsymbol{S}(\boldsymbol{f}_n)}$, where $\boldsymbol{A}(\boldsymbol{f}_n)$ is the $n \times 1$ amplitude vector. Subsequently, the $n \times 1$ phase vector $\boldsymbol{\phi}(\boldsymbol{f}_n)$ is built by assigning each spectral component a random phase between 0 and 2π radians. Next, a frequency domain signal $\boldsymbol{Z}(\boldsymbol{f}_n)$ is constructed as $\boldsymbol{Z}(\boldsymbol{f}_n) = \boldsymbol{A}(\boldsymbol{f}_n) \cdot e^{i\boldsymbol{\phi}(\boldsymbol{f}_n)}$.

Second, the frequency domain signal is transformed to a time-domain signal using the inverse fast Fourier transform, with the results shown in Fig. 8. These noise realizations, although unique in the time domain, all are made of the same spectral components. As such, the power spectral densities of both signals are nearly identical.

Finally, the measured outputs are simulated by averaging over the 50 s noise realization for noise attenuation.

III. Optimization Architecture

In this section, the online shape optimization strategy and framework are proposed. First, an overview of the complete optimization architecture is presented. Then each of the individual components is elaborated upon in the following subsections. The architecture of the proposed online shape optimization framework is shown in Fig. 9.

The optimization procedure involves a fast and a slow loop. The optimizer, on-board model, and cost function work together in the fast loop, marked by the shaded arrows. The optimizer evaluates angle-of-attack (α) and wing shape combinations ($\boldsymbol{u}$) on the on-board model with a high frequency. The resulting lift and drag coefficients from the on-board model are valued with a cost function (J), which is also based on the target lift coefficient (C_{L_t}). These cost values are in turn used by the optimizer to produce a more promising set of inputs for the next iteration of the optimization loop. Once the optimizer has converged onto the most promising set of inputs, they are evaluated on the system itself in the outer loop.

The on-board model and optimizer describe the wing shape in terms of five virtual inputs $u_1, \dots, u_5$ for wing shape smoothness and computational load reduction. However, since the shape of the morphing wing is controlled by 12 actuators $\theta_1, \dots, \theta_{12}$, the virtual input vector $\boldsymbol{u} \in \mathbb{R}^5$ needs to be mapped to the actuator input vector $\boldsymbol{\theta} \in \mathbb{R}^{12}$. Next, the actuator inputs are limited to their saturation limits of ± 25 deg. Subsequently, the wing shape and angle of attack are actuated on the system. In this study, the camber-morphing wing was simulated with an aerodynamic model of a wing with continuously distributed flaps. The resulting lift and drag coefficients are then contaminated with noise to simulate real-world measurements C_{L_m}, C_{D_m}. The inputs and outputs of the latest evaluation are added to the replay buffer, with a replacement strategy aimed at maintaining a global coverage of the input domain in memory. The model inputs $\boldsymbol{X}_i$ and model outputs $\boldsymbol{Y}_i$ in the buffer make up the training set that is used to train the on-board model. The training of the artificial neural networks that make up the on-board model results in new network weights $\boldsymbol{W}_{i+1}$. The process of the model update when new measurements become available is indicated by the diagonal gray arrow laying behind the on-board model block shown in Fig. 9. From here on, a new optimization cycle is initiated with an improved on-board model.

To evaluate the adaptability of the method, weights from previous training on a different wing, and no initial buffer data were used on the first iteration. To partly fill the empty buffer with data spread out over the input domain, the first 100 iterations (wandering phase) were performed with quasi-random inputs instead of the optimizer-computed optima.

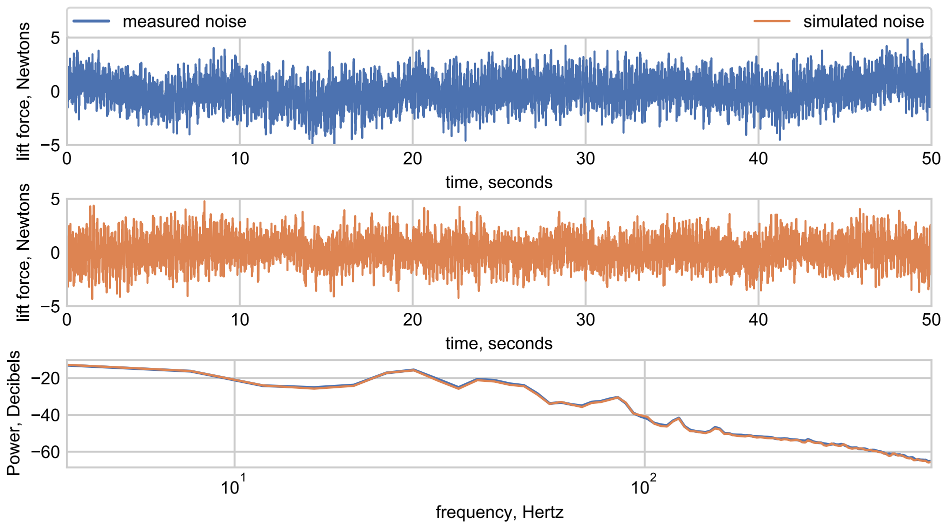


Fig. 8 Measured and simulated lift force noise signals.

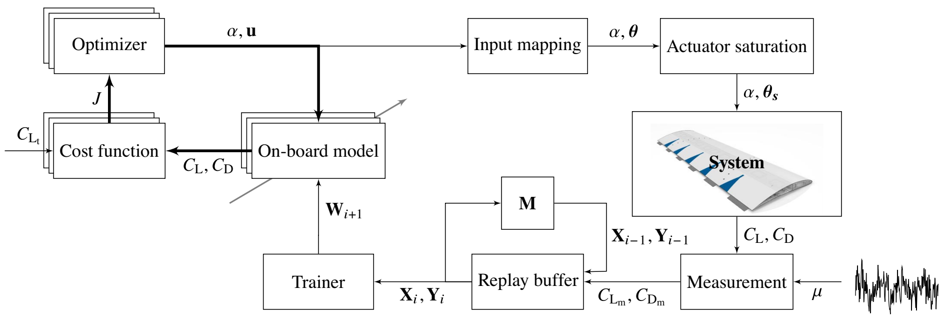


Fig. 9 Online shape optimization architecture.

As depicted in Fig. 9, the optimizer does not work with the system directly, but rather on the on-board model, which can be evaluated with much lower computational costs. The genetic optimization algorithm queries the on-board model with a population of inputs to be evaluated. The quality of these inputs is then determined from the model's outputs using a cost function. The optimizer in turn uses this information to generate a new group of candidate solutions. This loop is continued until the optimizer converges, after which this most promising input can be tested on the actual system.

The objective of the optimizer is to find the set of inputs $\alpha, u_1, \ldots, u_5$ that maximizes C_L/C_D while meeting the target lift coefficient C_{L_t} without violating the angle-of-attack or actuator limits. The mathematical representation is

$$\max_{\alpha,\boldsymbol{u}} \frac{C_L(\alpha,\boldsymbol{u})}{C_D(\alpha,\boldsymbol{u})}, \quad \text{subject to } \alpha \in [\alpha_{\min}, \alpha_{\max}],$$
$$\boldsymbol{\theta}_{\min} < \boldsymbol{\theta}(\boldsymbol{u}) < \boldsymbol{\theta}_{\max}, \quad C_L(\alpha,\boldsymbol{u}) = C_{L_t} \tag{5}$$

This problem is nonlinear and nonconvex because C_L and C_D are nonlinear and nonconvex functions of α and $\boldsymbol{u}$.

A. Cost Function

As the optimizer queries the system with certain inputs, the corresponding outputs from the system need to be valued to in turn inform the optimizer how well the input performed. The inputs cannot simply be scored on their associated drag, as this would tempt the optimizer into minimizing the drag, by minimizing the lift produced. Instead, a promising angle-of-attack and wing shape combination should result in both a low drag coefficient and a lift coefficient that is very close to the target lift coefficient. This is achieved with the cost function shown in Eq. (6).

$$J(C_L, C_D, C_{L_t}) = \underbrace{-\frac{C_L}{C_D}}_{\text{efficiency}} \cdot \underbrace{\frac{k_2}{k_1 + (C_L - C_{L_t})^2}}_{\text{deviation from lift target}} \tag{6}$$

The cost of any set of system outputs is dependent on the lift and drag coefficients, as well as on the target lift coefficient. The cost varies linearly with the aerodynamic efficiency C_L/C_D and is inverse-quadratically related to the difference between the target and actual lift coefficients. A small quantity $k_1 = 1 \times 10^{-4}$ is added to prevent singularities for small error values. The parameter $k_2 = 2 \times 10^{-5}$ is used to scale the output to $[-1, 0]$. Two- and three-dimensional plots of the cost function for $C_{L_t} = 0.50$ are shown in Fig. 10. Note that the cost increases rapidly for any deviation from the target lift coefficient, while steps in the drag-coefficient axis generally result in smaller cost variations as indicated by the isolines on the right side of this figure. In other words, this cost function prioritizes matching the target lift coefficient over reducing the drag coefficient.

This is important because the lift and drag coefficients are not independent variables, but are related through lift-induced drag. Hence, the optimizer needs to be discouraged from minimizing drag by reducing the amount of lift produced.

Additionally, the angle-of-attack and actuator constraints are also handled by the cost function. Since the actuator angles are intermediate variables that are unknown to the black-box optimizer, their constraints can only be enforced indirectly through penalties to the cost function. In the case of the angle-of-attack limits, direct min/max constraints on the input variable may be used instead, which has been found to lead to comparable results in this study.

$$J = (\alpha_i - \alpha^\star)^2 + C_J, \quad J = (\theta_i - \theta^\star)^2 + C_J \tag{7}$$

If a set of inputs violates any constraint, then its cost becomes as shown by Eq. (7). In the case that a set of inputs α to be evaluated is outside the bounds $[-2.5, 10.0]$ deg, the associated cost will be the square of the difference between the angle of attack α and the middle of the domain $\alpha^\star = 3.75$ deg plus a large constant C_J. The valid range of θ is $[-25, 25]$ deg so $\theta^\star = 0$ and Eq. (7) reduces to $J = (\theta_i)^2 + C_J$. This cost penalty constant is set to $C_J = 10$ to ensure that the cost will always be higher than that of an input set that is not in violation of these constraints. The square term serves to provide a gradient toward the middle of the parameter domain to direct the optimizer back to the feasible region of the input space. Note that this cost penalty is only incurred by solution candidates that breach the actuator or angle-of-attack limits, and thus does not bias the optimizer to favor the middle of the input domain when these limits are not reached.

B. Optimizer

The optimizer's goal is to find inputs to the on-board model that minimize the cost of the model outputs as determined by the cost function. This optimization is performed with the covariance matrix adaptation–evolutionary strategy (CMA-ES) algorithm [12]. CMA-ES is an evolutionary strategy for black-box optimization of nonlinear, nonconvex, and continuous problems. It can handle multimodality and discontinuities in the function to be optimized and has desirable global performance.

In the proposed framework, CMA-ES operates by iteratively generating populations of inputs that are subsequently evaluated on the on-board model. Based on the returned costs of these candidate solutions, the mean and covariance matrix of the next generation's population are adapted. This process is repeated until the variation of the cost function converges to a threshold, selected as 1×10^{-6}. In the online shape optimization procedure, a population size of 150 was used. The middle of the input domain was used as the initial solution point $\boldsymbol{x}_0$. Furthermore, the initial standard deviation $\boldsymbol{\sigma}_0$ and the scaling of the inputs were selected such that $\boldsymbol{x}_0 \pm 2\boldsymbol{\sigma}_0$ spanned the width of the inputs domains.

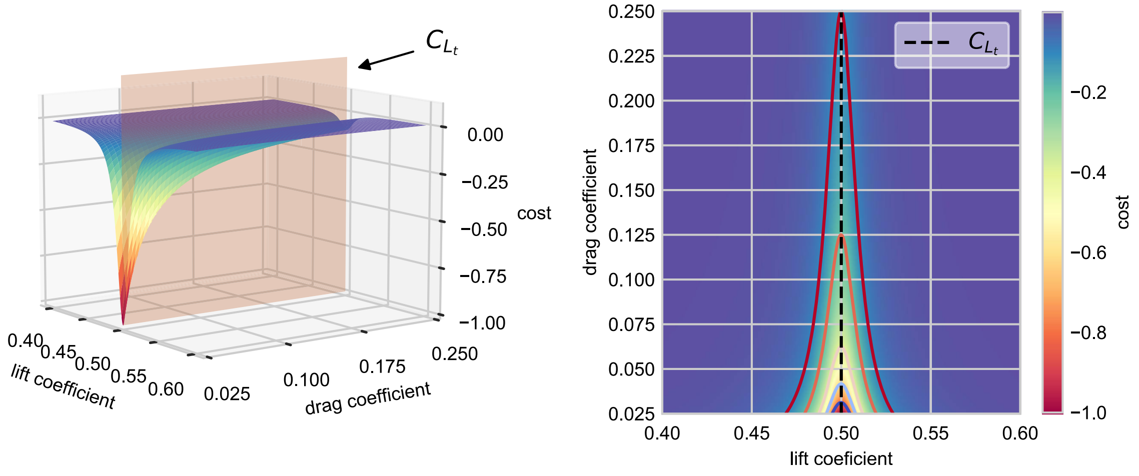


Fig. 10 Isometric (left) and top-down (right) view of the cost function for $C_{L_t} = 0.50$.

The total required number of function evaluations is dependent on the population size used, and also varies naturally due to the stochastic nature of the evolutionary strategy. Optimization with higher population sizes generally requires fewer optimizer iterations, but also requires more system evaluations per iteration. With a population size of 150, on average 180 optimizer iterations were needed with a total number of system evaluations of 27,000.

C. On-Board Model

The on-board model consists of two radial basis function artificial neural networks (RBFNNs) that model the mapping of the system inputs $\alpha, u_1, \ldots, u_5$ to the lift and drag coefficients. The C_L and C_D networks consist of a single hidden layer with 500 and 940 centers, respectively. More approximation power is needed for the C_D network than for the C_L network because of the higher degree of nonlinearity of the drag relation compared to the lift relation.

The training of the neural networks is done with mini-batch online training, with a batch size of 32. During training, the network weights are updated using the Adagrad algorithm proposed by Duchi et al. [20], with an initial learning rate of 0.01 and a mean squared error loss function.

The neural network models are not initialized with random weights, but rather with stored weights from a previous training session. In future applications, such a previous training session would be the online training performed during the most recent flight. For the simulations in this study, the starting weights for the online shape optimization will be weights from offline training on the nominal aerodynamic model. It is noteworthy that the simulated online optimization operates with the secondary model in the loop. Therefore, the initial weights serve only as a starting point and do not yet constitute a model that is representative of the system to be optimized.

For the initial offline training of the on-board model, a data set consisting of 261,360 wing shape and angle-of-attack combinations and their resulting lift and drag coefficients on the nominal model was used, with 10% of the data being reserved for validation. Both neural nets were trained from scratch for 2000 epochs, which equated to roughly 23 h of training time on a laptop (Intel® Core™ i7-4510U CPU, 8.00 GB RAM). Figure 11 shows the corresponding training and validation losses, converted to normalized root mean square errors (NRMSEs) for ease of comparison.

Even with the higher approximation power of the C_D network, the NRMSE of the C_L network is lower because of the lower degree of nonlinearity in the lift relation. The loss curves of both networks still exhibit a decreasing trend toward the end of the training session. The training cutoff at 2000 epochs is a tradeoff between computational cost and starting point quality. The increased computational costs of further training yield an increasingly diminished return in accuracy, and the networks are only to serve as a starting point for the on-board model.

The main benefit of using the on-board model instead of direct system evaluations is the low computational cost. The CMA-ES optimizer typically requires thousands of function evaluations to converge on an optimum. On the neural network models, hundreds of input combinations can be evaluated in less than 1 s, whereas on the aerodynamic model each evaluation takes 1.5 s on average. In other words, the indirect optimization using the on-board model is approximately 2500 times faster than the direct optimization on the aerodynamic model.

On a real-world aircraft, considerably more time would be required because of transients and noise filtering, making direct optimization unfeasible. Both direct optimization using the nominal aerodynamic model, and indirect optimization using the offline-trained on-board model were performed for a number of target lift coefficients. To make the computational time of the direct optimization more feasible, a population size of 9 was used for both. The resulting optimal shapes as computed by the CMA-ES optimizer are shown in Fig. 12.

The optimal shapes computed by indirect optimization are very close to those computed using the system directly. On average, the direct optimization took 44.7 minutes per target lift coefficient, whereas the average computational time of the indirect optimization was only 3.9 s (about 688 times faster).

D. Replay Buffer

During the online mini-batch training, the on-board model is trained on a set of training data kept in memory in the replay buffer. This buffer consists of a history of evaluated inputs and their corresponding lift and drag measurements. Since the on-board model is adjusted to adapt to these data, the contents of the buffer are of critical importance. If the training data set lacks data points in a region of the

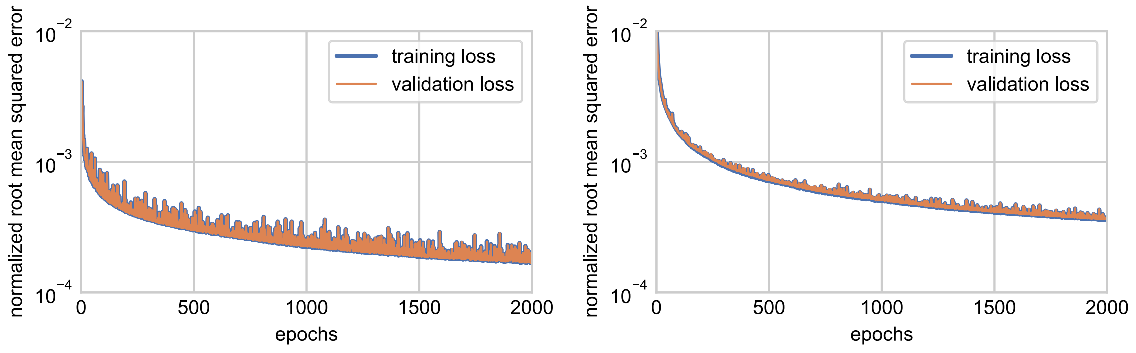


Fig. 11 Training and validation losses for the lift (left) and drag coefficient networks (right) in offline training.

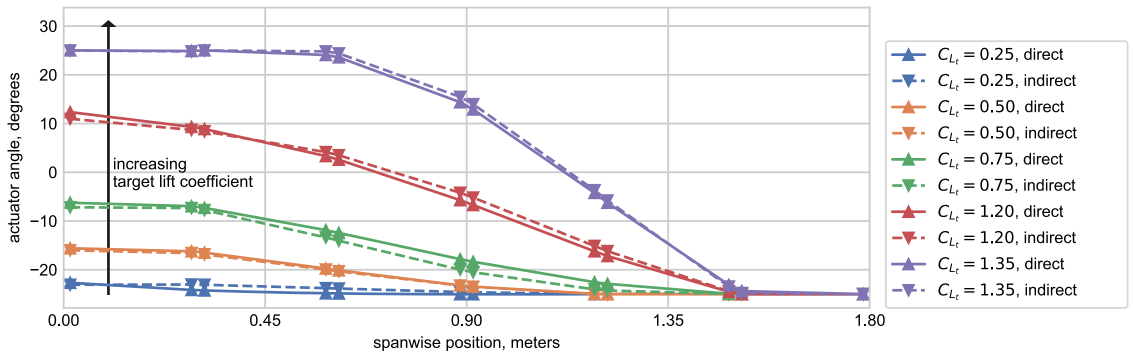


Fig. 12 Optimal wing shapes computed directly and indirectly on the system, for various target lift coefficients.

domain, then the neural nets will unlearn the previously learned from points in this region. This phenomenon, known as catastrophic forgetting, was first described in [21]. Therefore, a simple first-in-first-out training set buffer will not be sufficient to learn and retain a globally accurate on-board model.

Instead, the replacing of old data points when the buffer is full is based on a nearest neighbor search on all points in the buffer inspired by the coverage maximization strategy described in [22]. The data point with the lowest mean Euclidean distance to its 10 closest neighbors is replaced with the latest available data point. This replacement strategy aims to maximize the coverage domain of the training set by replacing the data points in regions of high data density and holding onto samples in data scare regions of the domain.

IV. Results and Discussion

In this section, the results from two simulation experiments are presented. Both experiments start with 100 iterations of input space exploration through pseudo-random actuation in a phase known as the wandering phase. In this phase of the experiment, measurements are collected and the on-board model is trained. This exploratory phase, during which no actuation of the computed optima takes place, is needed to prevent the optimizer from falsely identifying regions of high model error as regions of low drag. In future applications, this data gathering phase would be replaced by measurements from previous flights, or from high-fidelity simulations. During the following phase, the computed optimal inputs are actuated on the system. This phase is referred to as the optimization phase and is the nominal mode of operation of the algorithm. The selection of the length of the wandering phase represents a tradeoff between the time spent in the wandering phase and the accuracy of the on-board model at the start of the optimization phase. The minimum number of wandering phase iterations required is dependent on the number of measurements needed to train an acceptable on-board model across the entire domain. This in turn depends on the complexity of the real mapping, the dimensionality of the input space, and the distribution of the samples over the domain.

During the first experiment, the online optimization algorithm operated in optimization mode for 15 iterations with a fixed target lift coefficient of 0.75. During the second simulation, 275 iterations of online shape optimization were simulated with a target lift coefficient varying between 0.25 and 1.25. The aerodynamic efficiency of the resulting wing shapes was compared to that of the wing jig shape. The wing jig shape is defined as the shape of the wing at rest, with all morphing actuators set to zero deflection. The wing jig shape does not have any pretwist.

A. Single-Target Lift Coefficient

The first online shape optimization experiment was run for 115 iterations, of which the first 100 were performed in wandering mode and the rest in optimization mode. The inputs that were evaluated on the system are shown in Fig. 13, where the optimization phase is marked with a red background. As expected, both the angle of attack and the virtual inputs vary within their bounds with no recognizable pattern during the wandering phase. The cost associated with these pseudo-random inputs is generally high, with one notable exception at iteration 26, where the resulting C_L was relatively close to its target by coincidence. Shortly after the algorithm enters the optimization phase at iteration 101, the inputs plateau. At iteration 102, a shape is tried that results in higher cost than the shape from the previous iteration. Subsequently, the inputs move away from this location and the associated cost falls down and converges.

More insight into the inner mechanisms of the optimization algorithm is provided by the optimal inputs as calculated by the optimizer, shown in Fig. 14. The optimal angle of attack and optimal virtual

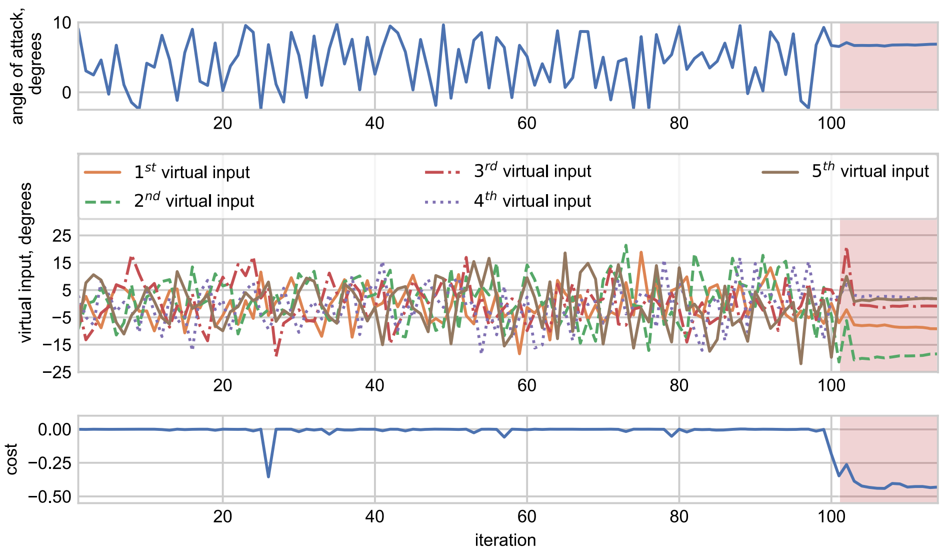


Fig. 13 Input history for wandering and optimization (red background) with $C_{L_t} = 0.75$.

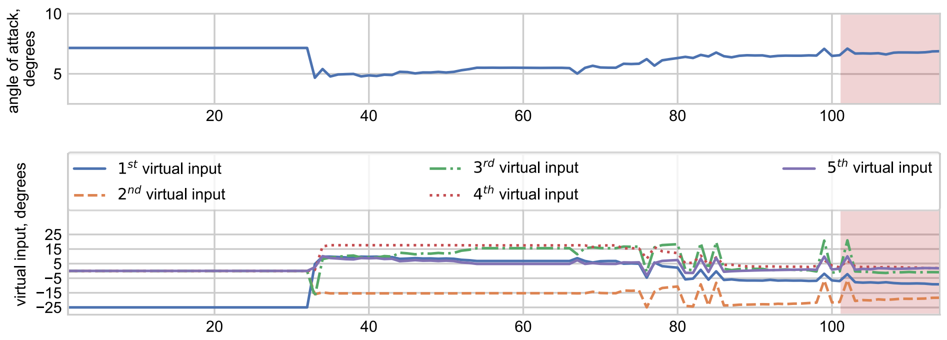


Fig. 14 Optimal inputs as calculated by the optimizer for a target lift coefficient of 0.75.

inputs remain unchanged for the first 32 iterations of the wandering phase. During this period, measurements are collected and the training buffer is partially filled. Training of the on-board model is only started after the training set size exceeds the batch size used for training.

At iteration 32, the online training is started and the algorithm's estimation of the optimal input changes with a sudden jump for the first time as the global minimum of the on-board model has shifted. Subsequently, the estimation of the optimal inputs changes repeatedly as the on-board model keeps training on an increasing number of data points and starts to represent the system more accurately. The fact that the optimal inputs only change slowly during the optimization phase, where estimated optimal inputs are evaluated on the system, indicates that the on-board model has captured the trends in the exploratory data quite well during the wandering phase. Two spikes in estimated optimal input can be observed at iterations 99 and 102, which correspond to inputs that seemed promising based on the on-board model at the end of the wandering phase, but once tested on the system actually yielded a lower performance than expected. After evaluation on the system, this input combination does not show up in the optimal inputs in later iterations.

The wing shapes evaluated on the system during both phases are shown in Fig. 15. The pseudo-random shapes, shown in blue, span the full actuator domain. The optimal wing shape, shown in orange, starts out with only minor changes in camber near the wing root, as compared to the wing's jig shape. Toward the tip of the wing, the camber of the wing is decreased until the actuators in the tip module hit their maximum negative deflection angles of −25 deg. This morphing shape brings the spanwise lift distribution of this zero-twist rectangular planform wing closer to the theoretically ideal elliptic lift distribution and thereby reduces the induced drag. One of the optimization phase shapes looks rather different from its counterparts. This is the shape that was tried on iteration 102 and resulted in an increase in cost compared to the previous iteration. In the following iterations, it was not repeated.

B. Various-Target Lift Coefficients

To investigate the ability of the online shape optimization algorithm to find optimal inputs for different target lift coefficients without repeated exploring, the optimization phase was extended to include two repeated series of steps and a window of gradual changes in the target lift coefficient as depicted in Fig. 16. The quality of the solutions actuated on the system was also evaluated by comparing their lift-to-drag ratios to those of the wing jig shape.

From iterations 100–160, the target lift coefficient is increased by 0.25 every 15 iterations. The steps in target lift coefficient are marked with dashed vertical lines. As a direct result of the steps in target lift coefficient, steps in the computed optimal angle of attack and virtual inputs can be seen at the corresponding iterations. For the duration of the steps, the optimal inputs are stable. The cost associated with the corresponding system outputs is also stable, although it is noisier due to the added measurement noise.

Between iterations 175 and 275 the target lift coefficient is decreased from 1.25 to 0.25 in steps of 0.01. As expected, the optimal angle of attack and mean camber of the optimal shape decrease as the target lift coefficient decreases. The first virtual input, which

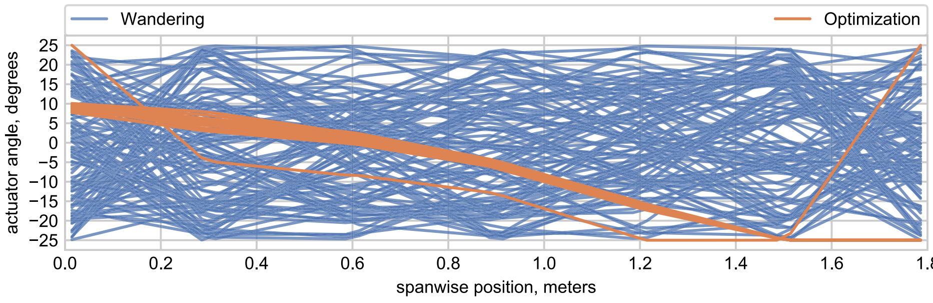


Fig. 15 Morphing shapes evaluated on the system in the wandering and optimization phases.

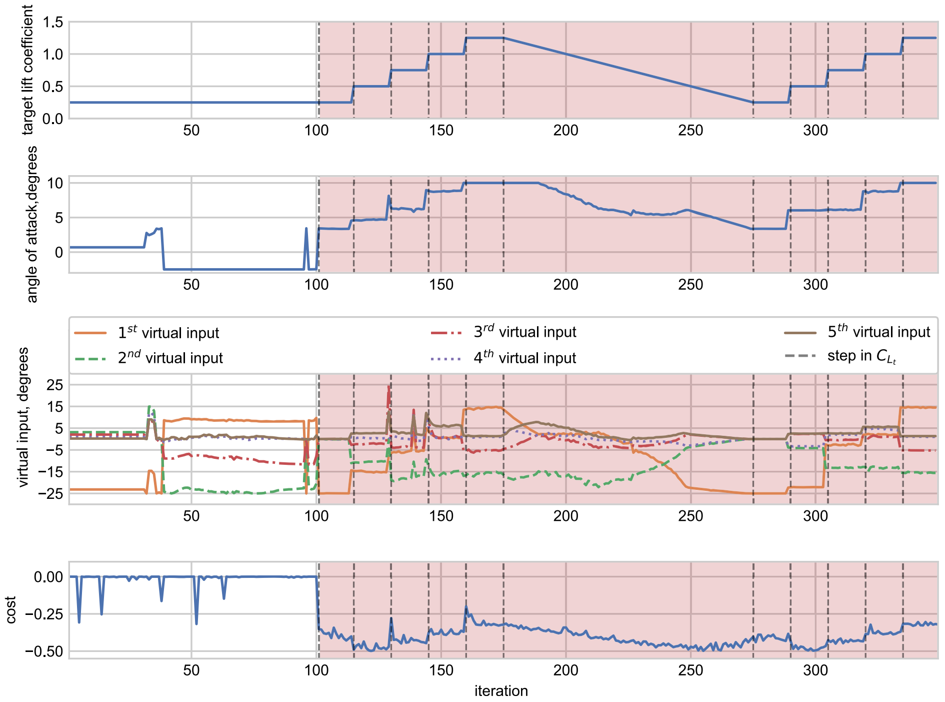


Fig. 16 Optimal inputs computed during the wandering and optimization (red background) phases.

contributes a constant amount of camber morphing along the wingspan, decreases until it nears the negative actuator limit of −25 deg between iterations 175 and 248. Meanwhile, the second virtual input, which represents a linear increase in spanwise camber morphing, becomes less negative. Here the optimizer increases the negative u_2 input because the lower u_1 input leaves less room for spanwise lift reduction before the actuators at the wingtip hit their maximum negative deflections. Between iterations 248 and 275, virtual inputs u_2 through u_5 are decreased to zero so that u_1 can all the way to the −25 deg actuator limit. In other words, for the target lift coefficient of 0.25, the optimizer sacrifices the increased lift induction efficiency of a more elliptical spanwise lift distribution for an overall less cambered airfoil. This makes sense since the airfoil already is relatively highly cambered, which is more efficient for producing higher lift coefficients.

After iteration 275, the same steps in target lift coefficient are repeated. The optimal inputs are almost the same between the runs, with the exception of $C_{L_t} = 0.50$ during iterations 290–305. Even though the inputs are different in this case, the costs are very similar. The average cost during iterations 115–130 is −0.475 with a standard deviation of 0.011, whereas the average cost during iterations 290–305 is −0.481 with a standard deviation of 0.018. Hence, on average the performance of the inputs evaluated during iterations 290–305 was slightly more desirable than those evaluated during iterations 115–130. Nevertheless, this again highlights the importance of accurate lift and drag estimations. Any combination of inputs can only be determined to be more efficient as long as the difference is measurable. In simulations without simulated measurement noise, the revisited target lift coefficients yielded the same inputs.

The lift coefficients and lift-to-drag ratios measured during the wandering and optimization phases are shown together with those of the jig shape in Fig. 17. As shown in Fig. 17, the quasi-random shapes from the wandering phase, shown in blue, produce lower lift-to-drag ratios than the jig shape, shown in green, in roughly 80% of the cases. Many possible shape variations exist that are aerodynamically inefficient, whereas only a smaller subset of shapes yield better aerodynamic performance. By chance, some random inputs perform comparably or even better than the jig shape.

With the exception of only two data points, the optimization points, shown in orange, all outperform the jig shape in terms of aerodynamic efficiency, although, for those two data points, the aerodynamic model output without simulated measurement noise does outperform the jig shape. Another effect of the measurement noise can be observed in the decreasing spread of the optimization point cloud with increasing lift coefficients. Naturally, as the lift and drag coefficients become larger, the lift-to-drag ratio becomes less sensitive to measurement noise. The clustering of optimization points at the target lift coefficients that were repeated for multiple iterations indicates that the optimizer is able to achieve the target lift coefficient very closely while also outperforming the jig shape.

An overview of the improvements in aerodynamic performance at various target lift coefficients achieved is shown in Table 1. As discussed before, the relatively highly cambered airfoil is naturally efficient at inducing higher lift coefficients. This is why the highest performance increases from active wing morphing are observed for low lift coefficients (0.25–0.50). At $C_L = 0.25$ the lift-to-drag ratio is increased with approximately 14.6%. At higher target lift coefficients, less increase in aerodynamic efficiency can be gained from changing the average amount of camber. At $C_L = 1.00$ the lift-to-drag ratio is increased with approximately 2.5%. Due to the rectangular planform, and absence of twist in the jig shape, reshaping of the spanwise lift distribution closer to an elliptical distribution yields an aerodynamic performance increase at all target lift coefficients.

Table 1 Efficiency improvements of the optimized wing shapes compared to the jig shape

C_{L_t}	C_D	L/D	L/D increase, %	C_D reduction, %
0.25	0.02995	8.35	14.6	12.8
0.50	0.05108	9.79	5.6	5.3
0.75	0.08420	8.91	2.9	2.8
1.00	0.12906	7.75	2.5	2.4

V. Conclusions

In this paper, a novel online learning-based black-box approach to active morphing wing shape optimization was presented. Its objective is to maximize the steady-state lift-to-drag ratio for a given target lift coefficient using lift and drag measurements. The presented method integrates an online-trained radial basis function neural network on-board model with an evolutionary optimization algorithm. This optimization strategy was validated on a seamless camber morphing wing, and its performance was compared to the performance of the wing jig shape. Before optimizing, the algorithm was allowed to explore the optimization space with pseudo-random inputs in the wandering phase. Subsequently, in the optimization phase, the on-board model was used by the optimizer to find the optimal wing shape and angle of attack to achieve the target lift coefficient on the surrogate wing model.

During the wandering phase, the radial basis function neural networks were able to sufficiently learn the mapping between the angle of attack, wing shape, and the resulting aerodynamic forces to facilitate the optimizer to find wing shapes that outperformed the jig shape. Moreover, the presented evolutionary optimization strategy was able to bring the zero-twist rectangular planform wing closer to the theoretically ideal elliptic lift distribution. Furthermore, due to the global character of the neural network on-board model used, the optimizer was able to find wing shape and angle-of-attack combinations with lift-to-drag ratio increases of up to 14.6% for a wide range of target lift coefficients without requiring further exploration.

In the present case, the input space of the on-board model is comprised only of the wing shape and angle of attack. In actuality, the mapping of these parameters to the lift and drag coefficients is also influenced by the Reynolds number and Mach number. Nevertheless, due to the black-box nature of the neural network on-board model, the Reynolds and Mach numbers can be easily incorporated as additional inputs to expand its scope to the full flight envelope of any camber morphing platform.

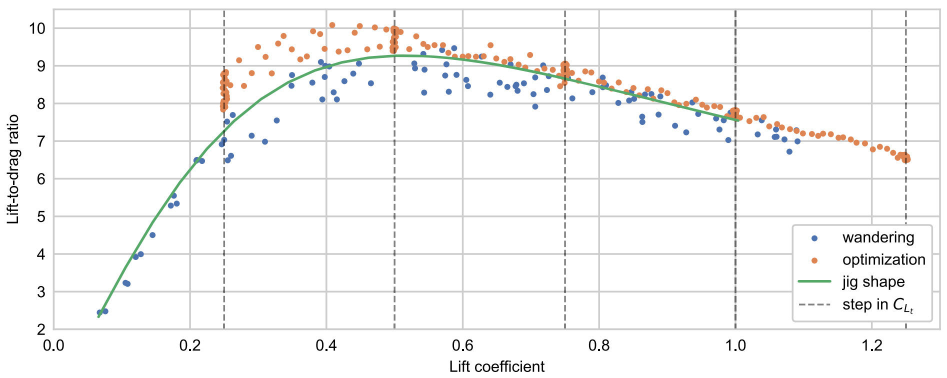


Fig. 17 Performance comparison of the jig shape and the online optimization shapes.

References

[1] Li, D., Zhao, S., Da Ronch, A., Xiang, J., Drofelnik, J., Li, Y., Zhang, L., Wu, Y., Kintscher, M., Monner, H. P., Rudenko, A., Guo, S., Yin, W., Kirn, J., Storm, S., and Breuker, R. D., "A Review of Modelling and Analysis of Morphing Wings," *Progress in Aerospace Sciences*, Vol. 100, June 2018, pp. 46–62.
https://doi.org/10.1016/j.paerosci.2018.06.002

[2] Wu, M., Xiao, T., Ang, H., and Li, H., "Optimal Flight Planning for a Z-Shaped Morphing-Wing Solar-Powered Unmanned Aerial Vehicle," *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 2, 2018, pp. 497–505.
https://doi.org/10.2514/1.G003000

[3] Jex, H. R., and Culick, F. E., "Flight Control Dynamics of the 1903 Wright Flyer," AIAA, Reston, VA, 1985, pp. 534–548; also AIAA Paper 1985-1804, 1985.
https://doi.org/10.2514/6.1985-1804

[4] Hubbard, J., Jr., "Dynamic Shape Control of a Morphing Airfoil Using Spatially Distributed Transducers," *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 3, 2006, pp. 612–616.
https://doi.org/10.2514/1.15196

[5] Nguyen, N., Lebofsky, S., Ting, E., Kaul, U., Chaparro, D., and Urnes, J., "Development of Variable Camber Continuous Trailing Edge Flap for Performance Adaptive Aeroelastic Wing," *SAE Technical Papers*, Vol. 2015, Sept. 2015.
https://doi.org/10.4271/2015-01-2565

[6] Ferrier, Y., Nguyen, N., and Ting, E., "Real-Time Adaptive Least-Squares Drag Minimization for Performance Adaptive Aeroelastic Wing," *34th AIAA Applied Aerodynamics Conference*, AIAA Paper 2016-3567, 2016.
https://doi.org/10.2514/6.2016-3567

[7] Nguyen, N., Cramer, N. B., Hashemi, K. E., Ting, E., Drew, M., Wise, R., Boskovic, J., Precup, N., Mundt, T., and Livne, E., "Real-Time Adaptive Drag Minimization Wind Tunnel Investigation of a Flexible Wing with Variable Camber Continuous Trailing Edge Flap System," *AIAA Aviation 2019 Forum*, AIAA Paper 2019-3156, 2019.
https://doi.org/10.2514/6.2019-3156

[8] Nguyen, N., and Xiong, J., "Real-Time Drag Optimization of Aspect Ratio 13.5 Common Research Model with Distributed Flap System," *AIAA Scitech 2021 Forum*, AIAA Paper 2021-0069, 2021.
https://doi.org/10.2514/6.2021-0069

[9] Horn, J. F., Schmidt, E. M., Geiger, B. R., and DeAngelo, M. P., "Neural Network-Based Trajectory Optimization for Unmanned Aerial Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 2, 2012, pp. 548–562.
https://doi.org/10.2514/1.53889

[10] Huang, A., Luo, Y., and Li, H., "Global Optimization of Multiple-Spacecraft Rendezvous Mission via Decomposition and Dynamics-Guide Evolution Approach," *Journal of Guidance, Control, and Dynamics*, Vol. 45, No. 1, 2021, pp. 1–8.
https://doi.org/10.2514/1.G006101

[11] Igarashi, J., and Spencer, D. B., "Optimal Continuous Thrust Orbit Transfer Using Evolutionary Algorithms," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 3, 2005, pp. 547–549.
https://doi.org/10.2514/1.11135

[12] Hansen, N., and Ostermeier, A., "Completely Derandomized Self-Adaptation in Evolution Strategies," *Evolutionary Computation*, Vol. 9, No. 2, 2001, pp. 159–195.
https://doi.org/10.1162/106365601750190398

[13] Dash, C. S. K., Behera, A. K., Dehuri, S., and Cho, S.-B., "Radial Basis Function Neural Networks: A Topical State-of-the-Art Survey," *Open Computer Science*, Vol. 6, No. 1, 2016, pp. 33–63.
https://doi.org/10.1515/comp-2016-0005

[14] Shankar, P., Yedavalli, R. K., and Burken, J. J., "Self-Organizing Radial Basis Function Networks for Adaptive Flight Control," *Journal of Guidance, Control, and Dynamics*, Vol. 34, No. 3, 2011, pp. 783–794.
https://doi.org/10.2514/1.51135

[15] Mkhoyan, T., Thakrar, N. R., De Breuker, R., and Sodja, J., "Design and Development of a Seamless Smart Morphing Wing Using Distributed Trailing Edge Camber Morphing for Active Control," *AIAA Scitech 2021 Forum*, AIAA Paper 2021-0477, 2021.
https://doi.org/10.2514/6.2021-0477

[16] Wang, X., Mkhoyan, T., Mkhoyan, I., and De Breuker, R., "Seamless Active Morphing Wing Simultaneous Gust and Maneuver Load Alleviation," *Journal of Guidance, Control, and Dynamics*, Vol. 44, No. 9, 2021, pp. 1649–1662.
https://doi.org/10.2514/1.G005870

[17] Löbl, D., Holzapfel, F., Weiss, M., and Shima, T., "Cooperative Docking Guidance and Control with Application to Civil Autonomous Aerial Refueling," *Journal of Guidance, Control, and Dynamics*, Vol. 44, No. 9, 2021, pp. 1–11.
https://doi.org/10.2514/1.G004425

[18] Sharpe, P., "AeroSandbox 2.2.11," PyPi, Sept. 2020, https://pypi.org/project/AeroSandbox/2.2.11/.

[19] Richter, J. S., Woodring, J. B., Fox, S. E., and Agarwal, R. K., "Performance Study of a Tapered Flying Wing with Bell-Shaped Lift Distribution," *AIAA Scitech 2021 Forum*, AIAA Paper 2021-0461, 2021.
https://doi.org/10.2514/6.2021-0461

[20] Duchi, J. C., Bartlett, P. L., and Wainwright, M. J., "Randomized Smoothing for (Parallel) Stochastic Optimization," *Proceedings of the IEEE Conference on Decision and Control*, Inst. of Electrical and Electronics Engineers, New York, 2012, pp. 5442–5444.
https://doi.org/10.1109/CDC.2012.6426698

[21] McCloskey, M., and Cohen, N., "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem," *Psychology of Learning and Motivation—Advances in Research and Theory*, Vol. 24, No. C, 1989, pp. 109–165.
https://doi.org/10.1016/S0079-7421(08)60536-8

[22] Isele, D., and Cosgun, A., "Selective Experience Replay for Lifelong Learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32, Assoc. for the Advancement of Artificial Intelligence, Menlo Park, CA, 2018.
https://doi.org/10.1609/aaai.v32i1.11595