TIME-EFFICIENT HARDWARE-IN-THE-LOOP AERODYNAMIC OPTIMISATION

Pawel Kekus[1] and David Angland[2]

ABSTRACT

A hardware-in-the-loop system was created for the purpose of aerodynamic optimisation. The system was used for fully independent optimisation of an automotive diffuser model in a wind tunnel, with linear actuators used to modify the geometry. The model had 3 degrees of freedom with 7×10^5 possible configurations. High reliability of the system was demonstrated and its time-efficiency quantified, with up to 400 configurations tested per hour. The system was tested with two implementations of a genetic algorithm, which demonstrated high consistency and resistance to the noise and hysteresis inherent in experimental data. Next, an attempt was made to minimise the main overheads of the optimisation process, that is wind tunnel settling time and measurement sampling time. In the original experiments, these overheads accounted for 35% of the total optimisation time. A series of simulated optimisation runs using a pre-sampled experimental database were carried out, and it was found that the settling time could be eliminated, and the sampling time reduced to as little as 0.01 s, without a detrimental effect on convergence efficiency. Thus, the average duration of a function evaluation was reduced, and the overall efficiency of the optimisation system was improved. The results were validated experimentally and the performance quantified through hardware-in-the-loop, real-time optimisation of a bluff body equipped with a dif-

[1]Department of Aerodynamics, ukasiewicz Research Network — Institute of Aviation, al. Krakowska 110/114, Warsaw, 02-256, Poland.
Faculty of Engineering and Physical Sciences, University of Southampton, Burgess Road, Southampton, SO16 7QF, United Kingdom
E-mail: Pawel.Kekus-Kumor@ilot.lukasiewicz.gov.pl.

[2]Faculty of Engineering and Physical Sciences, University of Southampton, Burgess Road, Southampton, SO16 7QF, United Kingdom
E-mail: D.Angland@soton.ac.uk

fuser. Up to 700 configurations per hour could be tested with the reduced overheads, leading to much faster convergence onto the global optimum.

Keywords: Aerodynamic optimisation, hardware-in-the-loop, diffusers, genetic algorithm, ground effect

INTRODUCTION

The process of aerodynamic optimisation typically involves changing geometric parameters of bodies in motion relative to a fluid. In most cases, the object in question is a vehicle, either terrestrial or aerial, which moves through the atmosphere (Skinner and Zare-Behtash 2018; Klimczyk and Goraj 2018; Klimczyk 2022; Drek et al. 2022; Drek et al. 2023). However, aerodynamic optimisation is also increasingly utilised in the design of structures, for example high-rise buildings or bridges, which have to withstand wind forces in addition to structural loads (Asghari Mooneghi and Kargarmoakhar 2016; Alkhatib et al. 2022; Zhao et al. 2016; He et al. 2019).

Aerodynamic development of all types of vehicles is driven by the need to maximise energy efficiency by minimising aerodynamic drag. Other examples of desirable aerodynamic qualities include low susceptibility to wind, high-speed stability of both aerial and ground vehicles, and high downforce of sports and racing cars. The challenge of maximising the above qualities has led to the development of numerous optimisation algorithms and methods (Skinner and Zare-Behtash 2018; Martins 2022). By formalising and automating the process of optimisation, greater gains can be achieved at the same cost compared to an exhaustive search through the parameter space.

There are three key problems that are of particular significance to the aerodynamic optimisation process, viz. model parametrisation, optimisation method selection, and performance quantification (Skinner and Zare-Behtash 2018). Parametrisation, or describing a geometry with a set of numerical variables, allows the algorithm to establish new configurations or shapes without the aid of a designer. The chosen optimisation method (or a combination of methods) must provide consistent convergence towards the global maximum,

while using minimal time and resources. Finally, every time a candidate configuration is created, its performance must be evaluated in order to feed information back to the optimisation algorithm.

An optimisation process requires hundreds or thousands of function evaluations to determine the optimal or near-optimal solution for complicated problems. In the case of aerodynamic optimisation, a function evaluation corresponds to either a CFD (computational fluid dynamics) simulation, iterating until satisfactory convergence is attained, or an experimental measurement, e.g. time-averaged forces or pressures, which is typically carried out in a wind tunnel, but may also be performed in motion in the open air (Zhang et al. 2020; Droandi and Gibertini 2015; Nabawy et al. 2012). The process between the measurements involves updating the geometry and meshing in the case of numerical methods, and redesigning or manually adjusting the model in the case of wind tunnel testing. When the former is used, it is the computation itself that is the most time-intensive, especially for unsteady problems, which require time-resolved CFD (Martins 2022). In the case of wind tunnel experiments, individual data points take in the order of seconds to acquire, but it is the intermediate process that takes the most time, i.e. stopping the wind tunnel, rebuilding or adjusting the model, and restarting the wind tunnel.

The goal of this research is to combine the two fast processes: quick data acquisition enabled by experimental measurement systems, and rapid geometry modifications. This can be achieved by placing a body capable of changing its shape in a wind tunnel and giving control of the geometry to an optimisation algorithm, similar to the procedure of optimisation using numerical methods. Finally, the algorithm needs to be able to automatically acquire and process measurements, which closes the loop. The whole process, commonly termed hardware-in-the-loop (HIL) aerodynamic optimisation, promises to reduce the total time required to optimise an aerodynamic geometry from days or weeks to hours. Furthermore, sufficiently fast morphing may enable more thorough exploration of the search space, reducing the likelihood of omitting the global optimum. On the other hand, the need for

a physical model restricts the ability to modify the shape to only the degrees of freedom included in model design, which limits the accessible search space relative to CFD-based optimisation, as well as increases the upfront costs of preparing the experimental setup.

Several hardware-in-the-loop aerodynamic optimisation systems have been implemented by researchers to date. Levinsky & Palko (1978, 1979, 1982) optimised the configuration of a flexible wing and tail model with 13 degrees of freedom (DoFs) using a gradient-based algorithm. A major challenge was shape deformation under aerodynamic loads, and the algorithm was incapable of sufficient exploration of the multi-dimensional search space. Rioual et al. (1994) created a closed-loop system for controlling the position of transition over a flat plate using distributed suction. Although this system did not involve optimisation, it demonstrated successful application of hardware-in-the-loop in a wind tunnel. Hunt et al. (2005) successfully optimised the flapping rate and tail position of an ornithopter using a genetic algorithm (GA). This is a notable example of performance evaluation carried out in tethered flight, as opposed to a wind tunnel. Boria et al. (2009) devised a real-time control system to optimise the camber of a thin-airfoil wing for maximum lift coefficient and lift-to-drag ratio, also utilising a genetic algorithm. The authors emphasised the importance of experimental repeatability in evolutionary optimisation. A significantly more complex morphing wing was created by Cosin et al. (2010), with the ability to vary its camber and twist at 5 spanwise locations, resulting in 10 DoFs. However, no analysis of optimisation performance, convergence or repeatability was presented. Arguably the most extensive research on the topic of HIL aerodynamic optimisation was carried out at the École de technologie supérieure in Montréal (Popov et al. 2010a; Popov et al. 2010b; Coutu et al. 2011). The researchers built a rectangular wing with a flexible suction surface, and utilised shape memory alloy actuators and optimisation combining simulated annealing and hill climbing to extend the region of laminar flow over the wing. Their actuation system was reported to be unreliable, slow, power inefficient, and with a tendency to overheat, which largely negated the benefits of using hardware-in-the-loop. The authors also raised the issue

of the global optimum potentially laying outside of the physically available search space, which highlights the importance of adequate model design. Finally, Perseghetti et al. (2015) applied closed-loop, GA-based optimisation to improve the flapping pattern of a flapping wing air vehicle in two flight conditions: gliding and flapping. Their optimisation tests were successful, with observed trends confirming predictions, but the issue of repeatability was also emphasised.

Despite the broad range of applications, most of the systems mentioned above suffered from the same issues. Poor accuracy, repeatability and hysteresis of shape articulation, hardware wear, measurement hysteresis and insufficient efficiency of optimisation algorithms in the presence of noise were all frequently encountered difficulties. However, perhaps the most common and notable issue was the poor speed of the systems, which at least partly negated the benefits of using a hardware-in-the-loop system.

The main aim of this work is to demonstrate hardware-in-the-loop aerodynamic optimisation as a viable, reliable and time-efficient alternative to CFD-based optimisation, and to quantify its performance on a bluff body equipped with a diffuser with 3 degrees of freedom. Although other optimisation methods, such as surrogate modelling, or a hybrid method, combining an initial global search with local gradient-based fine-tuning, might be more suitable in real optimisation problems, the genetic algorithm was chosen for this study due to its versatility and popularity, which make it an ideal reference point. In the HIL problem investigated here, it is also required to have an algorithm that is robust to experimental noise, making the GA a good candidate. The choice of the best optimisation method for a particular problem is beyond the scope of this article.

The second aim is to explore ways to improve the efficiency of the HIL optimisation process. Firstly, two versions of the genetic algorithm were applied to a hardware-in-the-loop optimisation process in a wind tunnel, with focus on robustness of the GAs when provided with experimental data, and on reliability and speed of the optimisation process. Secondly, an attempt to minimise operational overheads was made, involving reducing the

settling and sampling times during data acquisition. The effect on convergence was studied using simulated optimisation runs, and finally the findings were validated experimentally.

METHODOLOGY

This section presents the experimental setup devised for this study, an overview of the optimisation algorithms and testing procedures, as well as the methodology used for minimising operational overheads.

Experimental Setup

The geometry chosen for this study is an automotive underbody diffuser, as it represents a complex, multidimensional aerodynamic optimisation problem that may be easily reproduced through a physical model. Due to the inherently three-dimensional and unsteady flow, this problem is relatively expensive to solve computationally, particularly in the context of optimisation, making it ideally suited for an experimental investigation. The system, comprised of a set of actuators, a supporting structure with a 6-axis force transducer, and a closed-loop control system, permitted fully automated aerodynamic optimisation of the given geometry in a wind tunnel.

The model was a cuboid with a cross-section of $180\,\text{mm} \times 120\,\text{mm}$ and a length of 590 mm. The bottom of the model was comprised of two surfaces. The underfloor, 350 mm long, was hinged to the nose, and its angle (the rake angle γ) was modified by an Actuonix L12-P-210 linear actuator, which was mounted to the roof of the model. The diffuser plate, 150 mm long, was hinged to the underfloor, and its angle θ was controlled by an identical actuator, also mounted to the underfloor. The model had an open end, i.e. there was no surface connecting the trailing edge of the diffuser plate and the trailing edge of the roof. Simple, mechanical actuators were used for their reliability and speed. A picture of the assembled setup is shown in Fig. 1, and a schematic of the system is presented in Fig. 2. It is worth noting that the model was inverted, i.e. the top surface of the wind tunnel acted as the fixed ground.

The model was suspended from a supporting structure by three Actuonix L12-P-210 linear actuators, which were used to control the ride height h, or the height of the model above the ground. The lowest ride height used in the tests was 2 mm, or $h/H = 0.017$ in non-dimensional form (where H is the height of the model), which was made possible by the lack of a moving ground. The maximum ride height was limited to 92 mm ($h/H = 0.767$), yielding a 90 mm ride height range. The rake angle was limited to a maximum of $\gamma = 6°$, whereas the maximum diffuser angle was limited by the roof of the model, and varied between $\theta = 46°$ at $\gamma = 0°$ and $\theta = 21°$ at $\gamma = 6°$. A summary of the degrees of freedom, together with their resolutions, is presented in Table 1. The total number of possible states of the model, accounting for the variable diffuser angle range, was approximately 7×10^5. The maximum time required to transform the geometry between any two configurations was equal to 15 s. For a true optimum for this automotive diffuser, the effect of additional factors, such as Reynolds number and yaw angle, would also need to be included. However, the purpose of this study was not the determine the true optimum diffuser geometry.

The actuation system was controlled by a MATLAB script running on a personal computer (PC), which sent target positions for each actuator to a Simulink control system built on a dSPACE DS1006 processing board. The target positions were compared to positions reported by the actuators' built-in potentiometers, and the required direction of travel was selected if the positional error was greater than the specified precision window. Appropriate signals were then sent to the actuators through a digital-to-analogue converter (DAC) in the dSPACE chassis, and through a custom control board powered by a direct current (DC) supply. Position feedback was transmitted from the actuators through the control board, through an analogue-to-digital converter (ADC), and back to MATLAB.

The supporting structure was mounted to an ATI Mini40 force transducer, which was mounted directly to the wind tunnel wall. The sensing range of the transducer in the direction of downforce was 120 N, with a 0.02 N resolution. The force transducer was connected through a DC power supply to a laptop, where force and moment readings were acquired

and a calibration applied in a LabVIEW program. The measurements were then sent to the main PC through the local network, using the UDP protocol. A schematic of the control and acquisition system is displayed in Fig. 3.

The optimisation runs were carried out in an open-circuit wind tunnel at the University of Southampton. The driving fan was located at the inlet, followed by a series of flow-straightening honeycomb meshes, a contraction with a cross-sectional ratio of 7:1, and a rectangular test section with fixed walls and a cross-section of $600 \times 450\,\mathrm{mm}$, which resulted in a blockage ratio of approximately 12%. The freestream velocity was calibrated against fan frequency to a precision of $\pm 1\,\mathrm{m/s}$ prior to the experiments, and was maintained at $16\,\mathrm{m/s}$ by a dedicated controller during the tests. This resulted in a length-based Reynolds number of 6×10^5. Further detail on the facility and the experimental setup may be found in Kekus (2021), Kekus and Angland (2018).

Optimisation Algorithms

Measurement noise, hysteresis and drift are inherent aspects of experimental studies, and will therefore affect any hardware-in-the-loop optimisation process. Therefore, an optimisation method with global characteristics, with low susceptibility to these factors, is desired. Genetic algorithms fulfil this requirement, while also being highly adaptable to different characteristics of a problem, such as search space topology. For these reasons they are the most commonly applied algorithms in HIL aerodynamic optimisation, and were therefore chosen for this study. As discussed in the introduction, the purpose of this work was not to find the best optimisation algorithm for this particular problem.

In order to investigate the effectiveness of the GA on a problem with inherent measurement noise, hysteresis and drift, and to test the algorithm's sensitivity to its parameters, two implementations of a GA were tested in several distinctive configurations. The first implementation was the integer-encoded algorithm from MATLAB's Global Optimisation Toolbox (The MathWorks, Inc. 2017), and the second was a custom, binary-encoded algorithm, written specifically for this study. Apart from providing greater control over the

algorithm's functioning, the custom implementation enabled a comparison between integer and binary encoding. In the latter, all individuals are converted to binary strings for crossover and mutation, and converted back to integers before fitness evaluation. The discretisation of the search space of the diffuser model, dictated by the spatial resolution of the actuation hardware, required 21-bit strings in order to describe each configuration. Furthermore, crossover and mutation were applied to the entire population with their respective probabilities. Therefore, some individuals could undergo both operations, as opposed to the integer-encoded implementation, where each individual could undergo only either crossover or mutation. A large number of configurations of both implementations were tested on a sample problem, and three configurations with clearly distinctive behaviour were selected for each. The parameters of the chosen configurations are presented in Tables 2 and 3.

Testing Procedure and Problem Characteristics

Every algorithm configuration from Tables 2 and 3 was tested three times, in order to obtain average convergence characteristics and to qualitatively compare their effectiveness. All optimisation trials were terminated after an arbitrary limit of 60 min to assess the efficiency of the algorithms. This was not necessarily the time required for convergence (which was quicker in many cases), but it provides important information about the performance of the HIL system in terms of its time-efficiency.

The fitness function, or the cost function, is the function that the optimisation algorithm attempts to maximise or minimise, depending on the goal. In this study, the objective was to maximise the downforce (F_z) of the model, as its dependency on the three degrees of freedom of the system is well understood. In order to contextualise the discussion of these tests, the chosen objective function is plotted on two slices of the three-dimensional search space in Fig. 4. The plots were obtained by gradually increasing the diffuser angle at a range of ride heights and rake angles, and illustrate the trends in downforce experienced by the diffuser model. Large parts of the search space comprise monotonic variations, which constitute a simple optimisation problem. However, several discontinuities are present with

respect to diffuser angle, as well as minor local maxima, both of which add complexity to the optimisation problem. Furthermore, despite a sampling time of 2 s, experimental noise is also present in the measurements, most clearly visible in the low-gradient areas at high diffuser angles.

Another characteristic of this problem is aerodynamic hysteresis, where the forces generated by the body depend on the configuration or orientation of the body prior to its current state. In this case, gradually decreasing the diffuser angle results in different downforce values than when increasing it, even if the measurement is taken at the same diffuser angle. This phenomenon, which is a known feature of automotive diffuser aerodynamics, is illustrated in Fig. 5, where the difference in downforce depending on the direction of diffuser angle variations is plotted for the same two slices of the search space as in Fig. 4. The downforce coefficient difference is defined as the difference between the C_L when diffuser angle is being increased ($C_L\big|_{\theta\nearrow}$) and the C_L when it is being reduced ($C_L\big|_{\theta\searrow}$). The two distinct trends seen in both figures, referred to as regions A and B, are caused by shifts of the discrete changes in downforce from Fig. 4. The trends outlined above are discussed in more detail in Kekus and Angland (2018), Kekus (2021).

In the case of hardware-in-the-loop optimisation, aerodynamic hysteresis might result in different fitness values from repeated evaluations of a particular point in the search space, depending on the location of the preceding point. Furthermore, whereas Fig. 5 illustrates hysteresis with respect to diffuser angle, similar trends occur with respect to all three degrees of freedom. This significantly increases the complexity of the problem, posing an additional challenge for the optimisation algorithms.

Simulated Optimisation

The second aim of this work was to improve the overall operational efficiency of the HIL optimisation process. To this end, the impact of reducing the settling and sampling times was investigated. The settling time is the time allowed for the wind tunnel flow to settle after shape actuation is completed before measurements commence. Reducing the

settling time is likely to lead to increased hysteresis and noise in the measurements, due to potential capture of transient flow phenomena. The sampling time, on the other hand, is the duration of measurements used for averaging, and reducing it also increases measurement noise. However, the global nature of the genetic algorithm, which makes it resistant to both noise and hysteresis in the search space, means that it may be possible to reduce these times, and therefore the duration of each function evaluation, without negatively impacting convergence performance.

In order to investigate this problem and determine an optimal compromise between measurement noise and operational overheads, it was necessary to carry out a large number of tests with different values of settling and sampling times. As the wind tunnel running time required for this settling and sampling time study would be prohibitively large, a different methodology was devised to simulate these tests numerically. Configuration 2 of the binary-encoded GA was used for these simulations.

In order to obtain function evaluations quickly, interpolated downforce maps obtained in the wind tunnel, as illustrated in Fig. 4, were used in place of real-time wind tunnel measurements. However, a method to determine the noise of wind tunnel readings was necessary to simulate the impact of different settling and sampling times on convergence.

This was achieved by sampling the three-dimensional search space of the diffuser model with a low-resolution grid, and interpolating the results. The grid was $5 \times 5 \times 5$ points in size, equally spaced for every degree of freedom, i.e. the ride height, rake and diffuser angles. In order to obtain the standard deviation of downforce readings, i.e. noise of the function evaluations, 50 samples were collected in the wind tunnel at each point on the grid. Each sample was 10 s long, with a sampling frequency of 1000 Hz. Crucially, all samples were shuffled prior to data collection, so that the effect of aerodynamic hysteresis was taken into account, resembling real-time optimisation measurements as closely as possible. This was different to the experimental procedure used in the algorithm study, and was used only in the overhead reduction study. In order to validate the findings from this simulation study,

selected values of settling and sampling times were tested experimentally.

Maps of measurement noise were obtained for each combination of the following values of settling and sampling times, in addition to the baseline of $t_{\text{settling}} = 1.5\,\text{s}$ and $t_{\text{sampling}} = 2\,\text{s}$, which was used in the main HIL optimisation runs:

- Settling times: 0 s, 0.01 s, 0.05 s, 0.1 s, 0.2 s, 0.5 s, 1 s, 2 s, 5 s;
- Sampling times: 0.01 s, 0.05 s, 0.1 s, 0.2 s, 0.5 s, 1 s, 2 s, 5 s.

This was achieved by trimming the acquired 10 s samples according to the values of settling and sampling times. For example, given a settling time of 1.5 s and a sampling time of 2 s, only the parts between 1.5 s and 3.5 s were retained, and the remainder of the data was discarded. This corresponds to the experimental optimisation scenario, where sampling begins t_{settling} after actuation has ended, and ends after $t_{\text{settling}} + t_{\text{sampling}}$. After the desired sample fragments were isolated, they were averaged. This resulted in 50 measurements of downforce, obtained experimentally using the desired settling and sampling times. The standard deviation of these measurements was then computed across the search space and interpolated. During the simulated optimisation runs, the function evaluation at a given point in the search space was obtained by adding random noise of the corresponding standard deviation to the mean downforce value from this data point.

Furthermore, the corresponding wall-clock duration of each optimisation run was computed by adding the applied settling and sampling times to estimated actuation time between each pair of data points. This was calculated using actuation velocities for each degree of freedom measured experimentally in the wind tunnel. Every optimisation run was terminated once the calculated wall-clock duration exceeded 60 min, in order to allow comparisons to the baseline HIL optimisation runs. Furthermore, a minimum 1000 function evaluations were carried out in each run to provide sufficient comparison of convergence speed between configurations, as the configuration with the highest t_{settling} and t_{sampling} carried out fewer than 200 evaluations before 60 min elapsed.

A detailed explanation of the simulation procedure, including additional corrections, is presented in Kekus (2021).

RESULTS

Overview of Algorithm Convergence

Figure 6 illustrates the convergence of the HIL optimisation runs carried out using both implementations of the genetic algorithm. Tests using all configurations from Tables 2 and 3 are shown, and each configuration's translucent plot is defined by the maximum and minimum downforce coefficient (C_L) reached by a given function evaluation among its 3 trials. Apart from showing the pace of progression of the algorithms towards higher fitness, these plots also indicate the consistency of each algorithm implementation and configuration, with wide plots suggesting the algorithm's performance is susceptible to chance and vice versa. The values of maximum downforce coefficient $C_{L_{\max}}$ attained by each algorithm, as well as the corresponding model configurations (with the ride height h non-dimensionalised by model height H), are summarised in Table 4. The optimal ride height value of $h/H = 0.08$ was similar to the values determined in previous studies. The combination of high rake and diffuser angles results in strong pressure recovery along the model and therefore high downforce. However, neither the underfloor nor the diffuser plate were at their maximum deflections, suggesting that the global maximum laid within the physically available search space. Furthermore, nearly all optimisation iterations converged onto the same location, indicating that the global maximum was found consistently.

Figure 6a shows the progression of the integer-encoded genetic algorithm. The first two configurations exhibit similar performance, but configuration 2 reached a slightly higher downforce value in at least one run, and it was also slightly less consistent, reaching lower values after certain numbers of evaluations. Configuration 3 is the least consistent, and failed to locate the global maximum in at least one run, despite the greater number of function evaluations carried out within the 60 min time limit. The large number of evaluations and inconsistent convergence are indications of premature convergence, or too rapid a shift of

the population towards a small region of the search space. This was likely caused by the tournament selection, which tends to favour a small proportion of fittest individuals in every generation.

Next, Fig. 6b displays the convergence of the binary-encoded GA. Whereas its performance in the later stages of optimisation was similar for all configurations, the shift towards high fitness in the first few generations strongly depended on the settings. Configuration 2 consistently found the high fitness region fastest, but it retained sufficiently exploratory characteristics to avoid premature convergence, and located the overall highest values of downforce coefficient in all of its runs. The remaining configurations converged slower, but also consistently located the global maximum. The fast convergence of configuration 2 was likely caused by the exponential fitness scaling, which favoured high-performance individuals, while mutation biased towards less-fit individuals ensured sufficient exploration.

In order to better understand the characteristics of the two implementations, two representative runs are illustrated in greater detail in Fig. 7. The runs were obtained using configuration 2 of the integer-encoded and configuration 1 of the binary-encoded implementations. The left-hand side plots present spatial distributions of the diffuser configurations tested by the algorithms, displayed in the three-dimensional search space and coloured by fitness. The right-hand side plots display progression plots, where fitness of consecutive diffuser configurations is plotted against function evaluation, additionally sorted by fitness within individual generations.

The distribution plot in Fig. 7a illustrates the convergence of the integer-encoded genetic algorithm. It converges towards the region of maximum downforce effectively, placing relatively few points in the low-performance regions, but carrying out enough exploration to avoid falling into a local maximum. However, its behaviour is somewhat systematic, with many points appearing in lines, even within the high-performance region, which does not appear to be thoroughly explored. The progression plot in Fig. 7b confirms that the algorithm shifted most of the population to the high-downforce region quickly, while still

maintaining genetic diversity, as evidenced by the vertical spread of each generation. The distribution plot of the binary-encoded GA in Fig. 7c shows a more concentrated distribution of points, clustered around the high-performance region. Figure 7d shows that, with a smaller population size than the integer-encoded GA, the binary-encoded algorithm demonstrated gradual and seemingly more effective convergence onto the high-performance region, while also maintaining significant diversity.

The different encoding of the two implementations fundamentally changes the way the algorithms operate. The integer-encoded GA only exchanges values of particular variables, or genes, during crossover, and occasionally mutates some of the genes, which results in numerous duplicates or similar points (e.g. differing by only one variable and hence forming a line). As a result, on average only 61.4% of the diffuser configurations tested by configuration 1 of the algorithm were unique. The binary-encoded GA, on the other hand, constantly splits the genes in the form of binary strings for crossover, resulting in a more random yet concentrated distribution of data points, as well as fewer duplicates, with an average of 81.9% unique points across the 3 runs using configuration 2. However, these discrepancies are emphasised due to the low number of DoFs; usually at least one of the three genes is split in each crossover operation in the binary-encoded algorithm, and this ratio would be much smaller for problems with more DoFs. Notably, neither implementation was significantly affected by noise or hysteresis due to their population-based nature, locating the global optimum in all runs except for configuration 3 of the integer-encoded GA.

The data shown above is the result of fully-automated hardware-in-the-loop optimisation runs, carried out using simple, low-cost hardware. The efficiency and consistency of the GA-driven optimisation enabled reliable collection of up to 400 data points per hour, which vastly exceeds the results from previous studies of this kind, as well as what might be possible using any CFD solver capable of capturing the unsteady three-dimensional flow field generated by this kind of geometry. The performance of using a hardware-in-the-loop system for this aerodynamic optimisation problem was quantified. However, the quantitative findings

cannot be applied to any problem. In systems with large numbers of degrees of freedom and complex curvatures, requiring complicated actuating systems, the same performance may not be achieved. The experimental results also showed that GAs are sufficiently robust to provide consistent convergence even when provided with experimental data containing noise or problems that involve hysteresis. However, in order to seek further improvements to the efficiency of the process, an investigation into the reduction of overheads was carried out, and the results are presented in the following section.

Reduction of Overheads

The settling and sampling times that were used in the experimental optimisation runs discussed in the preceding section were conservatively set at 1.5 s and 2 s respectively, and on average constituted approximately 35% of the total optimisation time. Presented below are the results of simulated optimisation runs using a range of values of settling and sampling times.

The progression of mean maximum fitness obtained from 100 tests for each combination of settling and sampling times is compared in two ways. Firstly, with respect to function evaluation n, which reveals whether settling and sampling times had any effect on the convergence speed of the algorithms, and secondly, with respect to optimisation time t_{op}, which also takes into account the changing function evaluation length. Only the best-performing configuration of both implementations was used to illustrate this in Fig. 8 (configuration 2 in both cases — see Fig.7), and three distinctive settling and sampling time combinations:

- Baseline: $t_{\text{settling}} = 1.5\,\text{s}$, $t_{\text{sampling}} = 2\,\text{s}$;
- Reduced: $t_{\text{settling}} = 0.1\,\text{s}$, $t_{\text{sampling}} = 0.1\,\text{s}$;
- And minimal: $t_{\text{settling}} = 0\,\text{s}$, $t_{\text{sampling}} = 0.01\,\text{s}$.

Crucially, both implementations appear to be unaffected by reduced settling and sampling times, as their mean downforce grows at the same rate with respect to function evaluation for all three combinations (see Fig. 8a and 8c). This result confirms the resistance to noise

of the genetic algorithm, and affirms its suitability for hardware-in-the-loop optimisation systems.

The right-hand side plots (Fig. 8b and 8d) illustrate how reducing t_{settling} and t_{sampling} improves the overall efficiency of the optimisation process, allowing the algorithms to reach higher downforce values in the same amount of time. Average downforce gains of the reduced combination relative to the baseline after 60 min were 1.5% and 0.7% for the integer- and binary-encoded GAs respectively. However, after 20 minutes, these gains were 3.5% and 2.7% respectively, due to less advanced convergence. From a different perspective, the reduced settling and sampling times allowed the algorithm to attain downforce of 8 N ($C_L = 2.42$) 33.9% and 25.1% quicker for the two GA implementations. The differences between the reduced and minimal combinations were significantly smaller, due to the negligible change in function evaluation length.

In order to better visualise the potential benefit of reducing settling and sampling times on all tested combinations, mean maximum downforce relative to the baseline ($t_{\text{settling}} = 1.5\,\text{s}$ and $t_{\text{sampling}} = 2\,\text{s}$) after 20 min was averaged from all 100 trials for each case, and plotted as a function of $t_{\text{settling}} + t_{\text{sampling}}$ in Fig. 9. The 20 min point was chosen, rather than the time limit of 60 min, as most optimisation runs are not converged at this stage, highlighting gains in the early phase of optimisation. Every cross on the plots corresponds to a unique combination of settling and sampling times, and the coloured crosses correspond to the baseline, reduced and minimal combinations discussed above.

As seen in Fig. 9a and 9b, both implementations exhibit similar trends in response to changing settling and sampling times, with relative downforce growing from $0.95 < C_L < 0.97$ at $t_{\text{settling}} + t_{\text{sampling}} > 6\,\text{s}$, to $1.02 < C_L < 1.03$ at $t_{\text{settling}} + t_{\text{sampling}} \leqslant 1\,\text{s}$. At lower values the trend is approximately flat, with no significant improvement between 1 s and 0.01 s.

Experimental Validation

In order to validate the results presented in the preceding section, experimental tests were performed using the wind tunnel model, similar to the tests presented in Section 3. Due to

the small variations in convergence performance with changing settling and sampling times, 10 repeated trials of every combination of settling and sampling times were performed to obtain meaningful average performance. The same combinations were used as in Fig. 8, i.e. baseline, reduced and minimal. Configuration 2 of the integer-encoded GA was chosen for the validation tests due to its high performance, broad applicability to hardware-in-the-loop optimisation, and public availability. All optimisation runs were terminated after 60 min, following the previously established convention.

Similar to the simulation results (see Fig. 8), mean maximum downforce progression was averaged for each combination of t_{settling} and t_{sampling}, and plotted against function evaluation and optimisation time in Fig. 10. An additional correction was applied to account for the higher noise in measurements with short sampling time; details are omitted here for brevity, but may be found in Kekus (2021). It can immediately be seen from Fig. 10a that all three configurations of settling and sampling times exhibit very similar convergence speeds. The baseline configuration progressed slightly faster between the 100^{th} and 200^{th} function evaluations, but no significant differences are present after the 220^{th} evaluation. This confirms that the convergence speed of the genetic algorithm is not hindered by the increased noise due to reduced settling and sampling times, as previously suggested by the simulation results.

The most realistic comparison of convergence efficiency is shown in Fig. 10b, which displays a comparison of progression of downforce with respect to optimisation time. The plot suggests that decreasing the settling and sampling times from baseline to reduced, and from reduced to minimal, results in a gradual improvement in optimisation efficiency, with the minimal configuration achieving the highest average value of downforce throughout the entire run, higher than the baseline by up to 2% in the second half of the runs. However, in order to better understand the improvement in optimisation efficiency, we may swap the axes, and instead of comparing what downforce levels can be achieved in the same amount of time, compare how much time it takes to achieve the same levels of downforce.

Figure 11a shows the plot from Fig. 10b with swapped axes, representing the time needed to attain particular mean maximum downforce values. It can be seen that the shorter the settling and sampling times, the less time is needed to reach the majority of downforce values. In order to quantify the efficiency gain, Fig. 11b shows the time needed to reach particular downforce values relative to the baseline settling and sampling times, for both the reduced and minimal combinations. The two curves confirm that shortening function evaluations enables the GA to attain the same performance quicker. Both combinations can reduce the required optimisation time by up to 65%, or by around 30% to 40% on average, but crucially, reducing the times down to $t_{\text{settling}} = 0\,\text{s}$ and $t_{\text{sampling}} = 0.01\,\text{s}$ provided the greatest benefits. The minimal configuration was also able to attain the highest mean maximum downforce, confirming both the positive impact of reducing the function evaluation length, and the GA's resistance to noise.

However, the plots presented so far only illustrate average performance, but give no indication of the consistency of convergence when settling and sampling times are decreased. This crucial aspect of optimisation performance was inspected through plots of the standard deviation (SD) of downforce at a given function evaluation and optimisation time, as illustrated in Fig. 12. Figure 12a shows that the standard deviation of maximum downforce decreases rapidly as the algorithm converges, but in the first 200 evaluations it is lowest for the baseline combination, suggesting that the increased noise indeed has some negative effect on consistency. However, the reduced ($t_{\text{settling}} = 0.1\,\text{s}$, $t_{\text{sampling}} = 0.1\,\text{s}$) and minimal ($t_{\text{settling}} = 0\,\text{s}$, $t_{\text{sampling}} = 0.01\,\text{s}$) combinations allow the algorithm to converge further, and SD of downforce decreases down to 0.3% and 0.8% of the mean respectively, compared to 1.9% for the baseline.

Figure 12b displays a plot of standard deviation of downforce against optimisation time, and it can be seen that although similar values are achieved by the three combinations in the first 30 min, the reduced and minimal settling and sampling times enable the GA to attain much more consistent downforce levels in the second half of the runs. Therefore, despite

the increased experimental measurement noise, shorter settling and sampling times enable the optimisation algorithm to converge faster, to higher absolute values of fitness, and more consistently, in the same amount of time.

It should be noted that the optimal order of magnitude of the settling and sampling times will depend on the geometry used and the testing conditions. In systems where the flow takes longer to stabilise following geometry modifications, or where significant low-frequency vibrations or other phenomena are present, such low values of settling and sampling times may introduce a bias and negatively impact convergence performance. Nevertheless, the results shown above indicate that, with a stochastic optimisation algorithm with high resistance to noise and hysteresis, reducing the settling and sampling times to very low values might improve the overall efficiency of the optimisation process. These values are much lower than what intuitively might be expected for a HIL optimisation problem with noisy experimental data.

CONCLUSIONS

The concept of hardware-in-the-loop aerodynamic optimisation had been successfully implemented in several previous studies. However, the most notable potential advantage of the system — its efficiency — had not been hitherto exploited to its full potential. In this work, we showed that HIL optimisation can be applied to a relatively complex aerodynamic problem in a reliable and time-efficient manner. Even though the problem only contained 3 degrees of freedom, the flow field around the model was three-dimensional, with unsteady separated flows and contained aerodynamic hysteresis. Simulating it would require, at a minimum, a three-dimensional Reynolds-averaged Navier-Stokes computation.

The concept was applied to a simplified geometry of an automotive underbody diffuser, and the HIL system comprised a mechanical actuation system and a force transducer placed in a wind tunnel, with 3 degrees of freedom and 7×10^5 possible configurations. The genetic algorithm was used to perform the optimisation due to its global characteristics, which make it more resistant to noise in the data relative to local search algorithms. Optimisation

tests were carried out using several distinctive configurations of two implementations of the GA, and most of them provided fast and consistent convergence on the global maximum, suggesting limited dependence on algorithm settings for this problem. The binary-encoded implementation generally outperformed the integer-encoded counterpart, but it is expected that the difference would diminish with more degrees of freedom of the problem.

Crucially, the tests confirmed the genetic algorithm's robustness in the presence of the noise, hysteresis and drift that are inherent to experimental measurements. Therefore, it was hypothesised that it might be possible to reduce the wind tunnel settling time and the measurement sampling time without deteriorating convergence effectiveness. A series of simulated optimisation runs showed that the GA's performance was not noticeably affected even when the flow settling period was removed, and the sampling time reduced to as little as 0.01 s. Selected configurations were tested on the hardware-in-the-loop system in a wind tunnel, and the results confirmed that the GA could converge equally effectively, even with minimal settling and sampling times. The reduced overheads enabled it to acquire data twice as fast, at up to 700 configurations of the model per hour. This led to more thorough exploration of the search space in the same amount of time, or faster attainment of certain performance values, compared to more conservative settling and sampling times.

After the settling and sampling times were reduced, actuation of the model became the most significant overhead in the optimisation process. Notably, as genetic algorithms and other evolutionary optimisation methods operate by manipulating populations of individuals, it is possible to modify the order in which the individuals of a given population are tested, without affecting the operation of the algorithm. The model configurations from a single population may be represented as points in an N-dimensional hyperspace, where N is the number of degrees of freedom of the model. The total actuation time for the entire population will be correspond to the N-dimensional path linking all the points. Therefore, the objective is to find a path that connects all the points in the shortest possible time. This problem is analogous to the Chebyshev Travelling Salesman Problem, and its application to hardware-

in-the-loop aerodynamic optimisation is explored in Kekus and Angland (sub).

In summary, it was demonstrated that high time-efficiency can be achieved in a hardware-in-the-loop aerodynamic optimisation system through the use of fast and reliable actuation, a robust optimisation method, and overhead reductions in the form of reduced wind tunnel settling and sampling times. The study was carried out on a 3-DoF diffuser geometry with rigid parts, which although relatively complex aerodynamically, is simple to reproduce with a physical model. In systems with large numbers of degrees of freedom and complex curvatures, such as aircraft wings, the complexity of creating a mechanical system capable of providing sufficient geometry control might make the hardware-in-the-loop setup prohibitively costly and time-consuming. The trade-off between CFD-based and HIL optimisation will depend on a large number of factors, and an optimal method must be identified for each application.

DATA AVAILABILITY STATEMENT

Some or all data, models, or code that support the findings of this study are available from the corresponding author upon reasonable request. Selected data is available from the University of Southampton repository at https://doi.org/10.5258/SOTON/D3089.

REFERENCES

Alkhatib, F., Kasim, N., Goh, W. I., Shafiq, N., Amran, M., Kotov, E. V., and Albaom, M. A. (2022). "Computational aerodynamic optimization of wind-sensitive irregular tall buildings." *Buildings*, 12(7), 939.

Asghari Mooneghi, M. and Kargarmoakhar, R. (2016). "Aerodynamic mitigation and shape optimization of buildings: Review." *Journal of Building Engineering*, 6, 225–235.

Boria, F., Stanford, B., Bowman, S., and Ifju, P. (2009). "Evolutionary optimization of a morphing wing with wind-tunnel hardware in the loop." *AIAA Journal*, 47:2, 399–409.

Cosin, R., Angelo, M. V., Catalano, F. M., and De Salvi, F. T. B. (2010). "Mission adaptive wing optimization with wind tunnel hardware in the loop." *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, TX, USA.

Coutu, D., Brailovski, V., Terriault, P., Mamou, M., and Mébarki, Y. (2011). "Aerostructural model for morphing laminar wing optimization in a wind tunnel." *Journal of Aircraft*, 48(1), 66–76.

Droandi, G. and Gibertini, G. (2015). "Aerodynamic shape optimisation of a proprotor and its validation by means of CFD and experiments." *The Aeronautical Journal*, 119(1220), 12231251.

Drek, P. S., Kubacki, S., and ótak, J. (2022). "Multi-objective surrogate model-based optimization of a small aircraft engine air-intake duct." *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 236(14), 2909–2921.

Drek, P. S., Kubacki, S., and ótak, J. (2023). "Kriging-based framework applied to a multi-point, multi-objective engine air-intake duct aerodynamic optimization problem." *Aerospace*, 10(3), 266.

He, X., Fang, D., Li, H., and Shi, K. (2019). "Parameter optimization for improved aerodynamic performance of louver-type wind barrier for train-bridge system." *Journal of Central South University*, 26, 229–240.

Hunt, R., Hornby, G. S., and Lohn, J. D. (2005). "Toward evolved flight." *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, New York, NY, USA, ACM, 957–964.

Kekus, P. (2021). "Aerodynamics and experimental optimisation of automotive underbody diffusers in the presence of rake." Ph.D. thesis, University of Southampton, University of Southampton.

Kekus, P. and Angland, D. (2018). "Automatic wind tunnel-based optimisation of an automotive underbody diffuser." *2018 AIAA Science and Technology Forum*, Kissimmee, FL, USA.

Kekus, P. and Angland, D. (sub.). "Multi-dimensional chebyshev travelling salesman problem in aerodynamic optimisation.

Klimczyk, W. A. (2022). "Aerodynamic design and optimization of propellers for multirotor."

Aircraft Engineering and Aerospace Technology, 94(1), 21–30.

Klimczyk, W. A. and Goraj, Z. J. (2018). "Analysis and optimization of morphing wing aerodynamics." *Aircraft Engineering and Aerospace Technology*, 91(3), 538–546.

Levinsky, E. S. and Palko, R. L. (1978). "Semispan wind tunnel test of a computer-controlled self-optimising flexible technology wing." *10th Aerodynamic Testing Conference*, San Diego, CA, USA, 123–135. AIAA 78-786.

Levinsky, E. S. and Palko, R. L. (1979). "Supercritical tests of a self-optimising variable-camber wind tunnel model." *Advanced Technology Airfoil Research, Volume 1*, Hampton, VA, USA, NASA, 297–313. N79-20048.

Levinsky, E. S. and Palko, R. L. (1982). "Tests of an improved, computer-controlled, self-optimizing, variable-geometry wing." *12th Aerodynamic Testing Conference*, Williamsburg, VA, USA, 215–223.

Martins, J. R. R. A. (2022). "Aerodynamic design optimization: Challenges and perspectives." *Computers and Fluids*, 239, 105391.

Nabawy, M. R. A., ElNomrossy, M. M., Abdelrahman, M. M., and ElBayoumi, G. M. (2012). "Aerodynamic shape optimisation, wind tunnel measurements and CFD analysis of a mav wing." *The Aeronautical Journal*, 116(1181), 685708.

Perseghetti, B., Gallagher, J., Goppert, J. M., Yantek, S., Matson, E., and Hwang, I. (2015). "Optimizing energy efficiency of a flapping robotic bird through application of evolutionary algorithms." *2015 AIAA Science and Technology Forum*, Kissimmee, FL, USA.

Popov, A. V., Grigorie, L. T., Botez, R., Mamou, M., and Mébarki, Y. (2010a). "Closed-loop control validation of a morphing wing using wind tunnel tests." *Journal of Aircraft*, 47(4), 1309–1317.

Popov, A. V., Grigorie, L. T., Botez, R., Mamou, M., and Mébarki, Y. (2010b). "Real time morphing wing optimization validation using wind-tunnel tests." *Journal of Aircraft*, 47(4), 1346–1355.

Rioual, J. L., Nelson, P. A., and Fisher, M. J. (1994). "Experiments on the automatic control

of boundary-layer transition." *Journal of Aircraft*, 31(6), 1416–1418.

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

The MathWorks, Inc. (2017). "Global Optimization Toolbox - MATLAB. Available: `https://www.mathworks.com/products/global-optimization.html` [Accessed: 05 June 2017].

Zhang, W., Sun, J., Wang, L., Wu, J., and He, L. (2020). "Rotor airfoil aerodynamic design method and wind tunnel test verification." *Chinese Journal of Aeronautics*, 33(8), 2123–2132.

Zhao, X., Gouder, K., Graham, J. M. R., and Limebeer, D. J. N. (2016). "Buffet loading, dynamic response and aerodynamic control of a suspension bridge in a turbulent wind." *Journal of Fluids and Structures*, 62, 384–412.

NOTATION

The following symbols are used in this paper:

C_L = downforce coefficient (-);

$C_{L_{\max}}$ = maximum downforce coefficient (-);

H = model height (m);

F_z = downforce (N);

N = number of degrees of freedom;

h = ride height (m);

n = number of function evaluations;

t_{settling} = settling time (s);

t_{sampling} = sampling time (s);

t_{op} = optimisation time (s);

γ = rake angle (°);

θ = diffuser angle (°).

TABLE 1: Degrees of freedom of the HIL optimisation system.

Degree of freedom	Min. value	Max. value	Resolution	Number of states
Ride height h	2 mm	92 mm	0.4 mm	226
Rake angle γ	0°	6.0°	0.15°	41
Diffuser angle θ	0°	46.0° @ $\gamma = 0°$ 21.0° @ $\gamma = 6.0°$	0.43°	107

TABLE 2: Configurations of the integer-encoded genetic algorithm.

Setting	Configuration 1	Configuration 2	Configuration 3
Population size	50	70	30
Fitness scaling function	Rank	Rank	Top
Elite size	3	0	3
Selection function	SUS	SUS	Tournament
Crossover function	Uniform	Heuristic	Heuristic
Crossover fraction	0.8	0.9	0.9
Mutation function	Adaptive feasible	Adaptive feasible	Uniform

TABLE 3: Configurations of the binary-encoded genetic algorithm.

Setting	Configuration 1	Configuration 2	Configuration 3
Population size	50	30	70
Initialisation	Random	Random	Random
Fitness scaling function	Linear	Exponential	Linear
Elite size	3	3	3
Selection function	Roulette	Roulette	Tournament
Crossover function	Single-point	Two-point	Uniform
Crossover rate	0.8	0.8	0.8
Mutation function	Unbiased	Biased towards less-fit individuals	Unbiased
Mutation rate	0.05	0.08	0.1

TABLE 4: Maximum-downforce configurations attained by both GA implementations.

GA implementation	$C_{L_{\max}}$	h/H	γ	θ
Integer-encoded	2.3764	0.080	4.96°	24.63°
Binary-encoded	2.3844	0.073	4.96°	24.63°

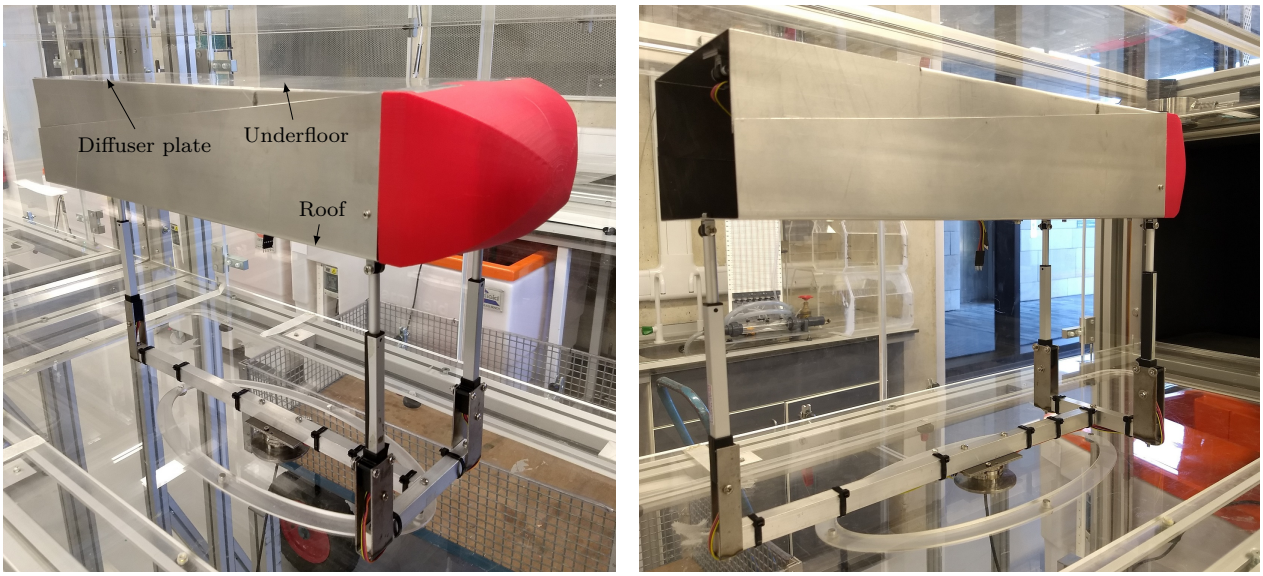


FIG. 1: The diffuser model used for HIL optimisation installed in the wind tunnel; $\gamma = 0°$, $\theta = 0°$. The model was installed upside down, with the wind tunnel roof acting as the ground.

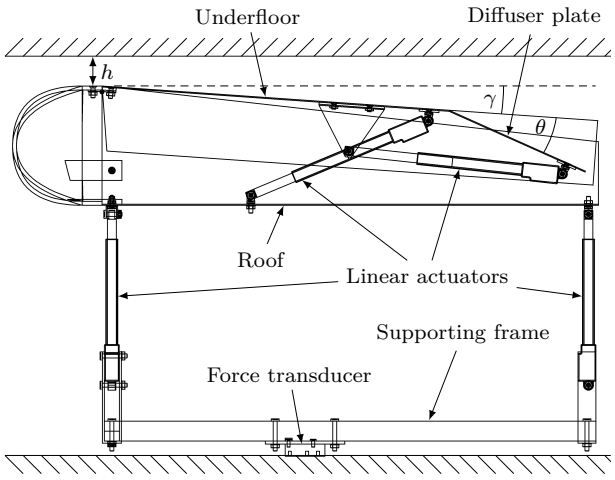


FIG. 2: Schematic of the diffuser model; $\gamma = 4°$, $\theta = 20°$.

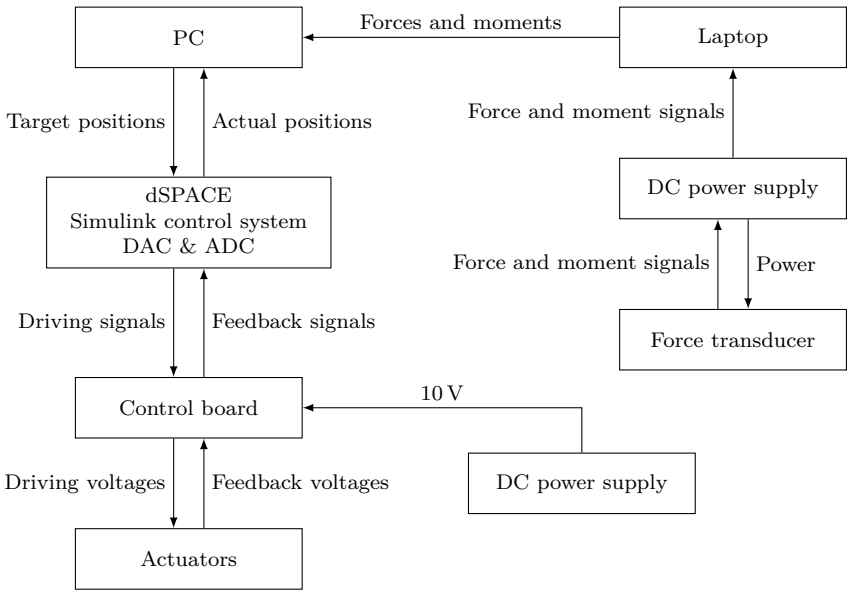
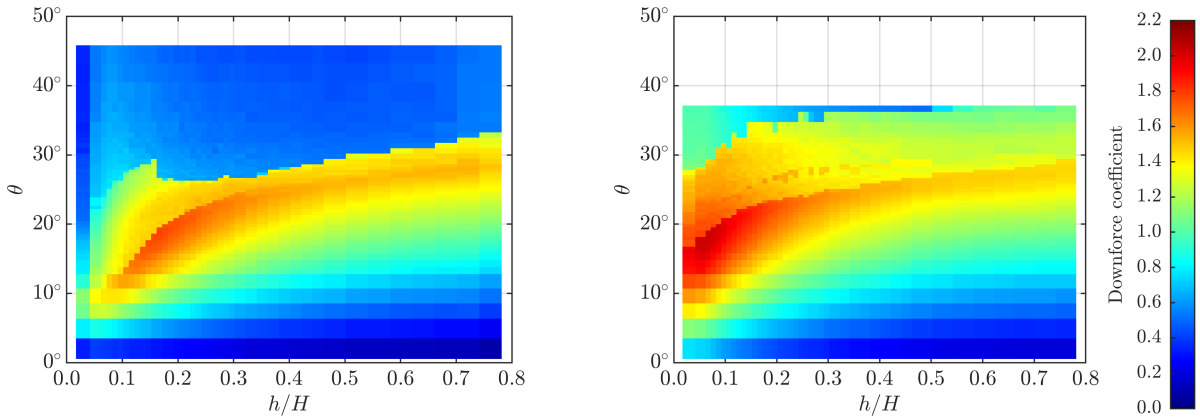


FIG. 3: Schematic of the small-scale model's control and acquisition system.



(a) $\gamma = 0.0°$.

(b) $\gamma = 2.0°$.

FIG. 4: Surface plots of downforce coefficient of the diffuser model, sampled with increasing diffuser angle for each ride height. The plots correspond to two slices of the three-dimensional search space.

(a) $\gamma = 0.0°$.

(b) $\gamma = 2.0°$.

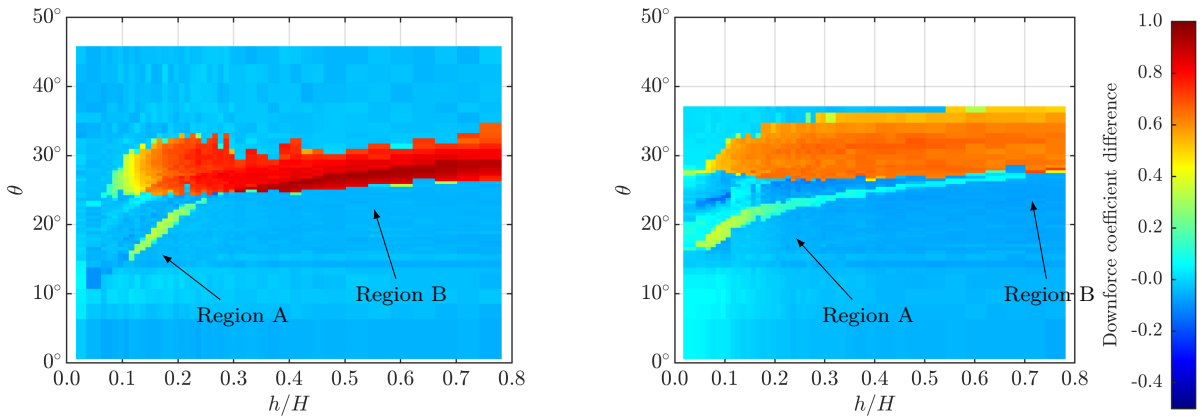
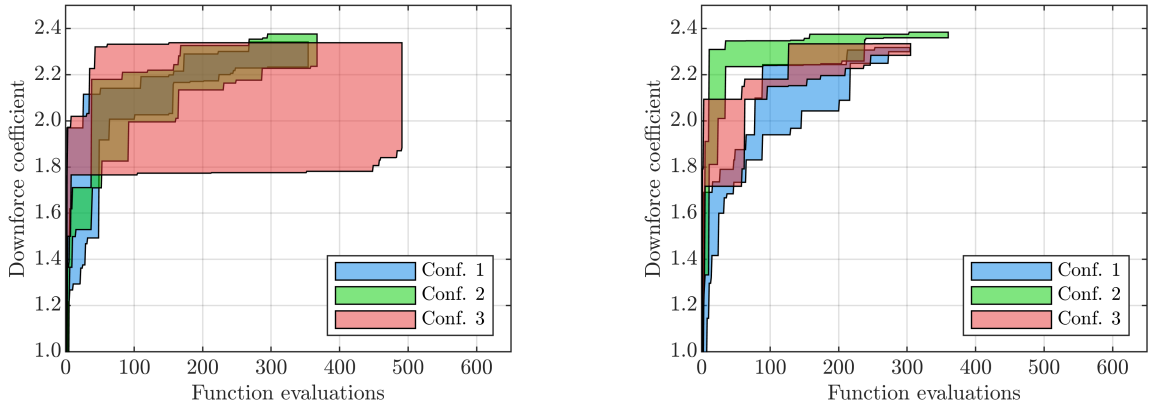


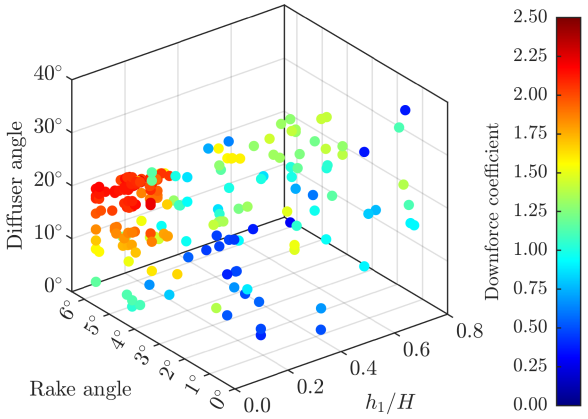
FIG. 5: Surface plots of hysteresis of downforce coefficient $C_L\big|_{\theta\nearrow} - C_L\big|_{\theta\searrow}$ of the diffuser model, showing the difference in downforce depending on the direction of diffuser angle variations.



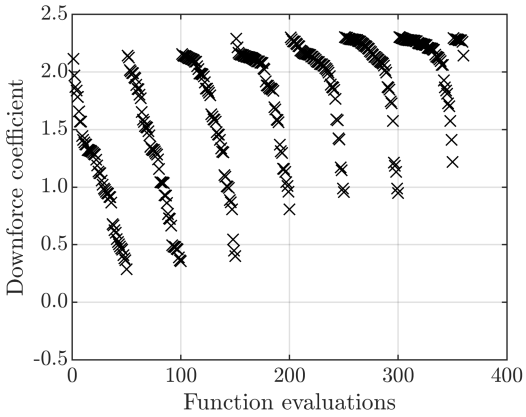
(a) Integer-encoded GA.

(b) Binary-encoded GA.

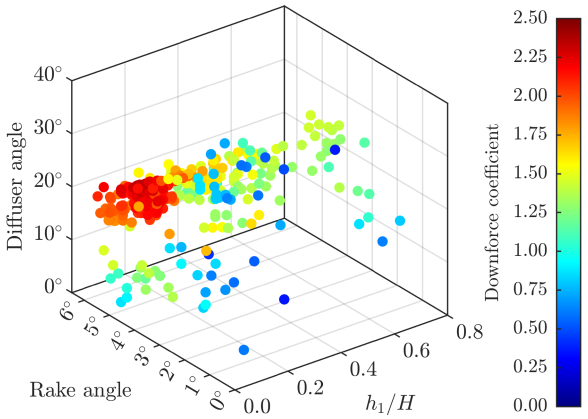
FIG. 6: Convergence of the two GAs during the 60 min optimisation runs. 3 trials per algorithm configuration.



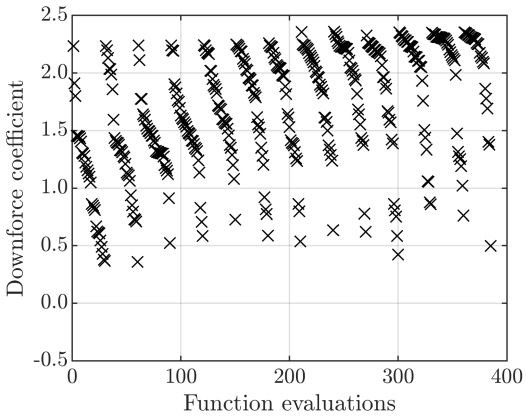
(a) Integer-encoded GA - spatial distribution of points.



(b) Integer-encoded GA - progression of fitness.

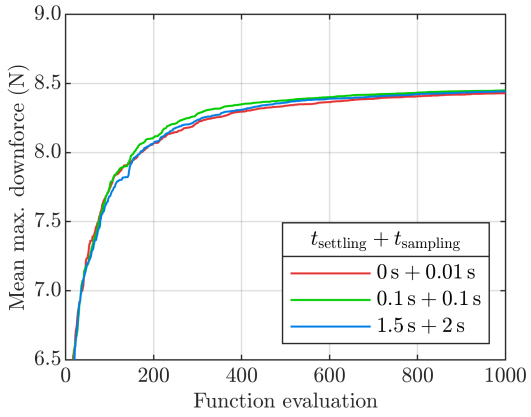


(c) Binary-encoded GA - spatial distribution of points.

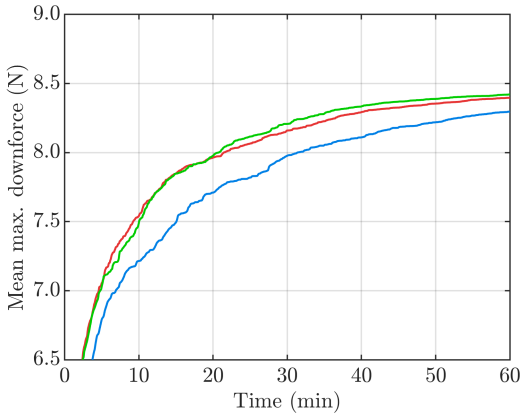


(d) Binary-encoded GA - progression of fitness.

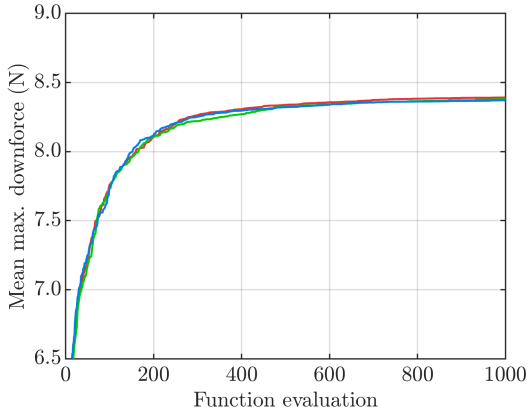
FIG. 7: Example spatial distributions (left-hand side) and progression plots sorted within individual generations (right-hand side) of both implementations of the GA, obtained from the HIL optimisation runs.



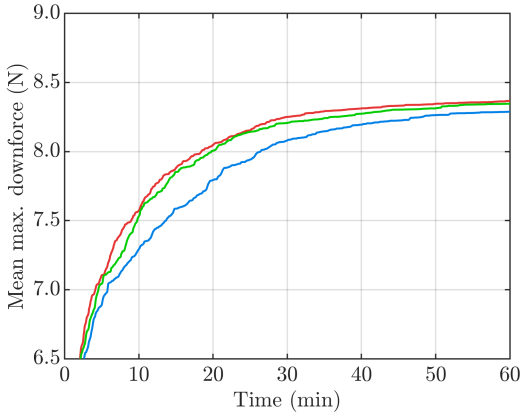
(a) Progression with n; integer-encoded GA.



(b) Progression with t_{op}; integer-encoded GA.

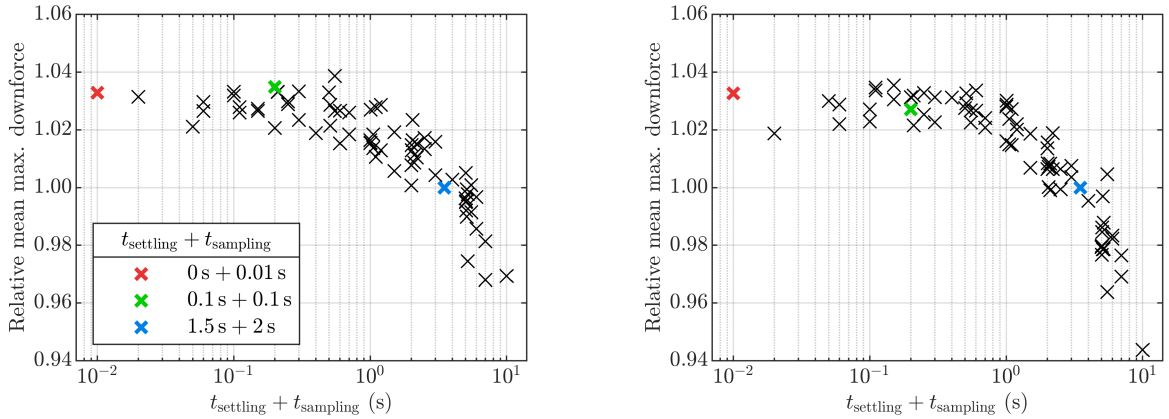


(c) Progression with n; binary-encoded GA.



(d) Progression with t_{op}; binary-encoded GA.

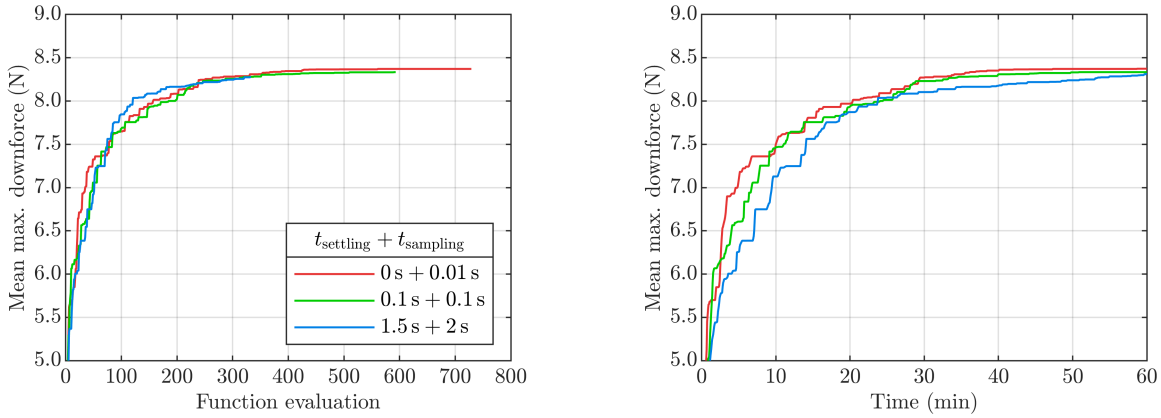
FIG. 8: Mean maximum fitness progression of selected configurations of both GA implementations, plotted against function evaluation and optimisation time.



(a) Integer-encoded GA.

(b) Binary-encoded GA.

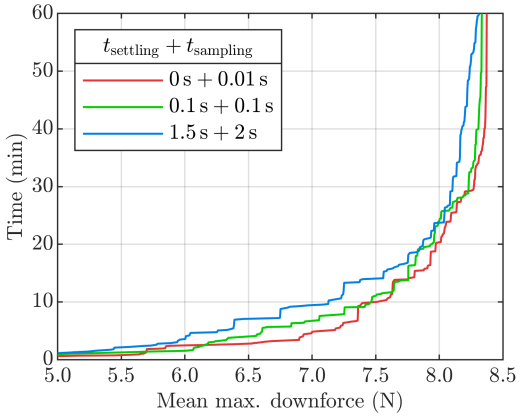
FIG. 9: Scatter plots of mean maximum fitness after 20 min relative to the baseline case, for both GA implementations, plotted as a function of $t_{\text{settling}} + t_{\text{sampling}}$. Each cross corresponds to a different combination of t_{settling} and t_{sampling}.



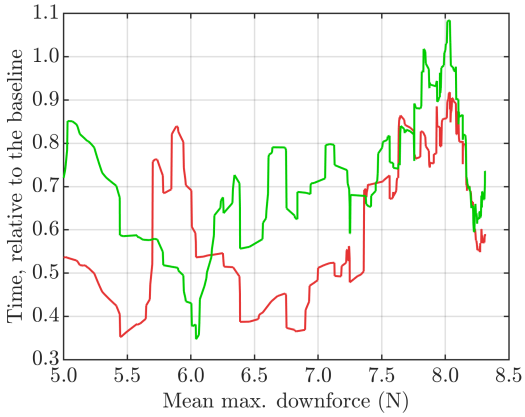
(a) Progression with n.

(b) Progression with t_{op}.

FIG. 10: Progression of mean maximum downforce against function evaluation and optimisation time, averaged from 10 experimental trials using configuration 2 of the integer-encoded genetic algorithm.

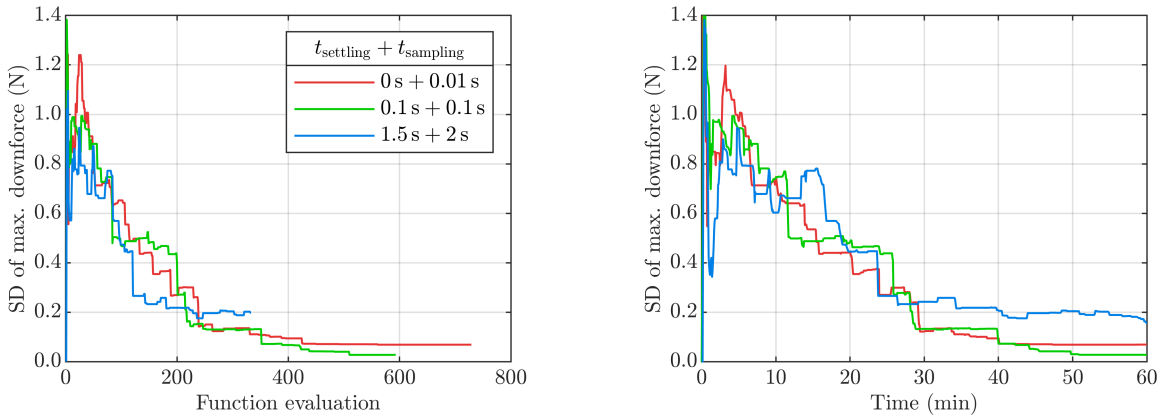


(a) All three configurations.



(b) Reduced and minimal combinations relative to the baseline.

FIG. 11: Time needed to reach particular mean maximum downforce values, for all settling and sampling time configurations, and relative to the baseline configuration.



(a) SD of max. downforce against function evaluation (n).

(b) SD of max. downforce against total optimisation time (t_{op}).

FIG. 12: Standard deviation of maximum downforce with respect to function evaluation and optimisation time.