A Generative Design and Drag Coefficient Prediction System for Sedan Car Side Silhouettes based on Computational Fluid Dynamics

Erkan Gunpinar a Umut Can Coskun b Mustafa Ozsipahi b Serkan Gunpinar c

aSmart CAD Lab, Istanbul Technical University, Turkey
bHeat and Fluid Research Group, Istanbul Technical University, Turkey
cDallas, Texas, USA

Abstract

A design support system is developed in this work that can be integrated into the car side silhouette design tools and can estimate the drag coefficient of a given silhouette. This task is typically performed via two manners: namely wind tunnel testing and computational fluid dynamics (CFD) simulations. Due to the high computational cost for these two approaches, it is impractical to employ them during the silhouette conceptual design stage in a real time. Therefore, a mathematical model is obtained in this study for the drag coefficient estimation of a given silhouette. First, the desired number of silhouettes are generated via a generative design (silhouette sampling) technique so that the silhouettes are evenly distributed in the silhouette design space. Each silhouette is then tested via computational fluid dynamics simulations, and their corresponding drag coefficients ($C_D$s) are obtained. A training dataset is formed with the silhouette geometries and $C_D$s of the silhouettes, and a mathematical model that can estimate the drag coefficient ($C_D$) of a silhouette is finally obtained via principal component analysis (PCA) followed by regression/neural network methods. These three steps are repeated until a desired level of reliable mathematical model is obtained. Finally, three generative design test cases are illustrated based on the mathematical model obtained to predict $C_D$ of a given silhouette.

Key words: Generative design, Computational fluid dynamics, Design support system, Design sampling, Machine learning

Fig. 1: Car side silhouettes generated using the proposed generative design technique (sampling algorithm) for the first (a), second (b), fourth (c), fifth (d) and sixth (e) algorithm runs.
1. Introduction

In the automotive industry, a car model’s appearance is a decisive factor in attracting customers’ attention. Several studies have been conducted to learn customer preferences concerning car side silhouettes [1–5]. Existing works [4–6] in the literature mainly couple their tools with an interactive genetic algorithm (IGA) to generate new car side silhouettes that satisfy customer desires. IGAs allow people with little or no CAD experience to interact with software to create and modify designs and shapes [5,7]. Gunpinar [1] has recently introduced a car side silhouette design technique; this gives a higher design flexibility to users and enables ordinary people to be directly involved in the silhouette design stage. This design technique is employed in the present work.

In addition to its stylistic form, the aerodynamic ability of a car is crucial. The aerodynamic performance of an object is expressed by the well-known dimensionless parameter of a car is crucial. The aerodynamic performance of an object is expressed by the well-known dimensionless parameter of $C_D$. The relation of $C_D$ is shown in Eq.1.

$$C_D = \frac{F_D}{0.5 \rho \infty V_\infty^2 A}$$

Here $F$, $V_\infty$, $\rho$, and $A$ stand for the drag force acting on the object, mean velocity of the free stream flow, density of air and frontal area of the car, respectively. For most cases, the area of an object is proportional to the drag force acting on it, which suggests that the frontal area of an object has minor effect on dimensionless $C_D$. However, the side silhouette (i.e., cross section) of the object has major role on $C_D$. This phenomenon can be further explained by comparing the $C_D$ of various objects. $C_{DPS}$ of concave hemisphere ($C_D=1.4$), circular disc ($C_D=1.17$) and convex hemisphere ($C_D=0.4$) with same frontal area by shape (lateral design) and size ($A$) placed perpendicular to flow with the same velocity have very different $C_{DPS}$ compared to each other [8]. Concerning these facts similar to its role in appearance, the car side silhouette is the most decisive aspect in its aerodynamic design [5], and therefore, the effects of lateral design of a car or its mirrors are ignored on $C_D$. By doing this, the computational costs for the $C_D$ calculation can be also reduced.

Some works in the literature have considered aerodynamic performance for cars. One important study on this topic was carried out by Tseng et al. [5], who adapted a feature-based aerodynamic drag coefficient meta-model called CDaero, originally developed by Calkins and Chan [9]. However, CDaero has some limitations, and it is not fully compatible with Gunpinar’s [1] silhouette design technique. These limitations are listed below:

(a) CDaero takes only a few parameters into account and assumes that some of these parameters have constant average values [5]. This may lead to incorrect predictions for the aerodynamic drag of a silhouette;

(b) The empirical equation for the automobile drag coefficient prediction, which was used as the basis for CDaero, was developed using a small number of cars. The empirical algorithm in [10] was applied to the automobile designs for models from 1988 through 1993; and

(c) Tseng et al. [5] only employed 18 parameters in the CDaero model. However, these parameters cannot fully describe a car side silhouette. Gunpinar’s [1] design framework employs nine Bezier curves with 26 control points (CPs) to represent a silhouette. There are 52 parameters in total. The empirical equation developed by Tseng et al. [5] may produce the same drag coefficient values for silhouettes with different geometries and have less accurate prediction results.

In this work, extensive computational fluid dynamics (CFD) tests with 1000 distinct silhouettes were performed, and the resulting data is employed to predict the drag of a given silhouette. The proposed system consists of three consecutive steps that are iteratively executed until a mathematical model is obtained that reliably (at a desired level) predicts the drag of a given car side silhouette. First, a specific number of silhouettes are generated via a generative design (silhouette sampling) algorithm, which can generate silhouettes that are evenly distributed in the silhouette design space (see some of the sampled silhouettes in Figure 1). In this way, better training data for the silhouette drag estimation step can be formed so that a reliable mathematical model can be obtained in a shorter time. The drag coefficient ($C_D$) of each silhouette is then calculated via CFD simulations. Finally, the silhouette geometries with their computed drag coefficients ($C_{DPS}$) are employed in a machine learning step to obtain a mathematical model for the silhouette drag prediction. If the mathematical model cannot estimate the $C_D$ of a new silhouette reliably (i.e., the error metric is lower than a user-defined threshold), these three steps are again repeated to obtain better training data for the machine learning step (i.e., drag coefficient prediction). Fig. 2 shows the flow of the proposed system. The mathematical model is appropriate for use in the silhouette design optimization, which is shown using generative design scenarios in the experiments.

The remainder of this paper is organized as follows: Section 2 reviews the relevant literature. Section 3 explains the proposed system, which obtains a mathematical model that can reliably predict the drag coefficient of a car side silhouette. The results and discussion are given in Sec. 4. Finally, concluding remarks and opportunities for future work are presented in Sec. 5.

2. Related works

The work reported here is related to user-centered car silhouette design, generative design, and automobile drag coefficient prediction. Some of the important previous works on these topics are discussed in this section.
Fig. 2. A desired number of silhouettes are generated via a generative design technique, which is called silhouette sampling. CFD tests are then performed to compute the drag coefficients ($C_D$) for the sampled silhouettes. The mathematical model (M. Model) to predict $C_D$ of a given silhouette is computed in a machine learning step using silhouette geometries and their $C_D$ obtained in the previous stages. These three steps are consecutively iterated until obtaining a mathematical model having a desired level of $C_D$ prediction accuracy.

2.1. User-centered car silhouette design

User-centered design in an automobile’s exterior shape is an important criterion for customers. Several efforts have been made to enable such design, some of which are mentioned here. Kelly et al. [4] employed IGAs for the generation of car silhouettes, while Cluzel et al. [6] proposed an IGA to sketch the desired car side silhouette. A Fourier decomposition of a 2D silhouette was adopted as the genotype, and a cross-over mechanism was employed to generate new silhouettes. Hyun et al. [11] synthesized car designs using a technique involving a mixture of Fourier decomposition, an eye tracker, and a shape grammar, while preserving brands design styles. In addition, Hyun and Lee [12] synthesized car designs based on strategic styling positioning and investigated ways of implementing design similarities for synthesizing styles. Shieh et al. [13] suggested an integrated model based on multi-objective optimization and multi-criteria decision-making.

2.2. Generative design

Several advancements have been made in the field of generative design in the last decade. The development of generative systems has passed through various stages, and the implementation of generative design has already been widely recognized [14]. Krish [15] developed an exhaustive searched-based generative technique for the creation of design alternatives. Moreover, Kazi et al. [16] proposed a generative design system in which a user generated an initial design by sketching, and alternatives were then generated in the sketched context. A user needed to have digital sketching abilities to utilize the system. Sousa and Xavier [17] developed a symmetry-based generative system for digital fabrication of a variety of geometric shapes. Shea et al. [18] and Turrin [19] at al. introduced performance-driven generative design systems to create lightweight architectural structures. Utilizing generative design techniques, some researchers generated site layouts [20], as well as energy efficient [21] and eco-friendly building designs [22,23]. Burnap et al. [24] represented the design space as designs sampled from a statistical distribution of form. A generative model of the distribution is estimated using a set of images and design attributes of previous designs. Recently, Gunpinar and Gunpinar [25] proposed a design sampling technique for CAD models for the generation of space-filling designs using a particle tracing method. Khan and Gunpinar [26] suggested another sampling technique based on the teaching-learning-based optimization from Rao et al. [27]. The designs generated using this technique had semi-Latin Hypercube property, as well as space-fillingness [28]. Khan et al./Khan and Gunpinar employed spatial simulated annealing [29]/Latin hypercube sampling [28] to generate space-filling designs for customer-centered products [30]/[31]. Umetani [32] converted triangular meshes into quad meshes with consistent topologies. A low-dimensional representation for the shapes were constructed via the autoencoder networks, which was then employed to synthesize new shapes.

2.3. Automobile drag coefficient prediction

Currently, two approaches can be chosen to determine the aerodynamic drag properties of a vehicle’s geometry: namely, wind tunnel testing and CFD analysis. They both have advantages and disadvantages relative to one another. Wind tunnel testing provides more reliable results, but it is more time consuming and expensive [33]. CFD has evolved significantly in the past two decades, becoming a vital tool in industrial research, development, and investigation nevertheless, CFD’s accuracy depends on the modeling approach of the application. [34–36]. Therefore, there is a need to predict automobile drag coefficients, which can be integrated into the design tools as a design support system. Aerodynamic optimization has many applications such as aircrafts, cars, trains, bridges, wind turbines. Lyu and Martins [37] performed aerodynamic shape optimization for a 800-passenger blended-wing-body aircraft. A gradient-based optimizer and a CFD solver were used to minimize the drag coefficient at a cruise condition. Skinner and Zare-Belhtash [38] have reviewed the aerodynamic shape optimization methods with their applications. The aerodynamic drag predictive tool CDuero, developed by Calkins and Chan [9], is based on the empirical work of Carr and Stapleford [39]. Twenty 1984 vintage automobiles were evaluated using the empirical method and compared with the wind tunnel data results. The method’s accuracy was reported as ±5%. Carr and Stapleford’s vehicle model had 13 discrete parametric equations that modeled the primary contributions of aerodynamic drag. This model was improved by Guan [10], who defined the vehicle shape using 51 parameters. It was reported that the system could predict the vehicle drag coefficients within +8.2% to −15.2% of the real wind tunnel results. This model was further im-
proved by Chan [40], so that 6% accuracy on the same test cases was achieved.

2.4. Contributions

The main contributions of the proposed car side silhouette drag coefficient prediction system are as follows:

- A novel silhouette sampling technique is introduced to generate a desired number of uniformly distributed silhouettes in a silhouette design space;
- The silhouette sampling, CFD analysis, and machine learning steps work jointly until a mathematical form is obtained that reliably predicts the drag coefficient of a given silhouette to some extent;
- The silhouette design space in our work is far wider than those of the previous works [9,10,40]. As a result, the mathematical model for the drag prediction of the silhouettes is obtained using a greater number of more distinct silhouettes than in those works; and
- The mathematical model can be easily integrated into the car side silhouette design optimization step.

3. Silhouette generation

The car side silhouette (silhouette) in this work is represented using nine characteristic lines [1], as follows: the bumper line, grill line, hood line, windshield line, roof line, rear windshield line, two trunk lines, and rear bumper line (see Fig. 3). A silhouette, $S$, is composed of 21 design parameters, $[t_1, w_1, t_2, w_2, t_3, l_3, w_3, \ldots, t_9, w_9]$: $t_i$, $l_i$, and $w_i$ represent, respectively, the line template (representing the shape), length, and height for the $i$th characteristic line, where $i = 1, \ldots, 9$. The characteristic lines are indexed based on the order as mentioned in the first sentence of this paragraph. Line templates for the characteristic lines are represented by either quadratic or cubic Bezier curves, and they are obtained in a preprocessing step by changing the control point positions. Quadratic Bezier curves are utilized for the front and rear windshield lines, and cubic Bezier curves are used for the remaining lines. Furthermore, no length or height parameter is assigned to some of the characteristic lines, as it is not sufficiently meaningful to define these design parameters for these lines. The bumper, grill, trunk-2, and rear bumper lines have negligible lengths, while the roof and trunk-1 lines have negligible heights. Table 1 shows the design parameter bounds used in this study.

<table>
<thead>
<tr>
<th>Characteristic Line</th>
<th>Line Template</th>
<th>Length ($l$)</th>
<th>Height ($h$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bumper Line</td>
<td>$t_1$</td>
<td>$-$</td>
<td>$220 \leq w_1 \leq 450$</td>
</tr>
<tr>
<td>Grill Line</td>
<td>$t_2$</td>
<td>$-$</td>
<td>$110 \leq w_2 \leq 300$</td>
</tr>
<tr>
<td>Hood Line</td>
<td>$t_3$</td>
<td>$830 \leq l_3 \leq 1430$</td>
<td>$190 \leq w_3 \leq 340$</td>
</tr>
<tr>
<td>Windshield Line</td>
<td>$t_4$</td>
<td>$580 \leq l_4 \leq 930$</td>
<td>$320 \leq w_4 \leq 400$</td>
</tr>
<tr>
<td>Roof Line</td>
<td>$t_5$</td>
<td>$1320 \leq l_5 \leq 1690$</td>
<td>$-$</td>
</tr>
<tr>
<td>Rear Windshield Line</td>
<td>$t_6$</td>
<td>$540 \leq l_6 \leq 860$</td>
<td>$210 \leq w_6 \leq 290$</td>
</tr>
<tr>
<td>Trunk-1 Line</td>
<td>$t_7$</td>
<td>$220 \leq l_7 \leq 520$</td>
<td>$-$</td>
</tr>
<tr>
<td>Trunk-2 Line</td>
<td>$t_8$</td>
<td>$-$</td>
<td>$290 \leq w_8 \leq 440$</td>
</tr>
<tr>
<td>Rear Bumper Line</td>
<td>$t_9$</td>
<td>$-$</td>
<td>$320 \leq w_9 \leq 490$</td>
</tr>
</tbody>
</table>
end CP of its preceding characteristic line to make these CPs coincident. When such transformations are applied, the end point of the rear bumper line can be much higher or lower than the start point of the front bumper line, which is non-realistic. A non-realistic silhouette has a geometry that is far from the producible/real silhouette geometries. Therefore, silhouette repairing is performed, which scales the characteristic lines to obtain a realistic silhouette. The difference between the Y coordinates for the start point of the front bumper line and the end point of the rear bumper line should approach to zero for a realistic silhouette. Therefore, this difference is adjusted to zero via scaling some of the characteristic lines. The scaling amounts are proportional to the widths of characteristic lines.

The silhouettes obtained by assembling the line templates may not always have geometric continuities at the connection points between the windshield and roof lines. Such continuity constraint is not enforced in this work as there are no silhouettes. Different line templates can be generated by changing the CP positions. Each row of the feature vector is represented using X and Y coordinates for these points. The cost function $E$ in Eq. 2 should be minimized and favors the generation of silhouettes with different geometries based on the potential energy of Audze and Eglais [41,28]. There are two terms in $E$, as follows: $E$ and $E$ (Eq. 3 and 4, respectively). $E$ favors the generation of N silhouettes that are distinct from each other, while $E$ allows the generation of the $N$ silhouettes that are distinct from the $M$ silhouettes obtained from the previous runs of the sampling algorithm. Note that an algorithm run consists of a single run of the silhouette sampling. As stated above, the algorithm runs these consecutive steps until a stopping criterion is met. In the first algorithm run, the term $E$ can be omitted, as there are no silhouettes.

$$E = E + E$$  \hspace{1cm} (2)

$$E = \sum_{p=1}^{N-1} \sum_{q=p+1}^{N} \sum_{t=1}^{9} \frac{1}{D_{pq}^2}$$  \hspace{1cm} (3)

$$E = \sum_{p=1}^{N} \sum_{q=1}^{M} \sum_{t=1}^{9} \frac{1}{D_{pq}^2}$$  \hspace{1cm} (4)

$$D_{pq}^t = \sqrt{\sum_{j=1}^{3n} d^2(\chi_{p,j}^t,\chi_{q,j}^t)}$$  \hspace{1cm} (5)

Let $S_p$ and $S_q$ be two different silhouettes among the $N$ silhouettes, $(S_1,S_2,\ldots,S_N)$, to be generated. And $S_r$ represents a silhouette obtained from the previous runs of the silhouette sampling algorithm. $\chi_{p,j}^t/\chi_{q,j}^t/\chi_{r,j}^t$ denotes the $j^{th}$ row of the feature vector for the $t^{th}$ characteristic line of $S_p/S_q/S_r$, respectively. $D_{pq}^t$ in Eq. 5 is based on the function $d(\chi_{p,j}^t,\chi_{q,j}^t)$. Each row of the feature vector is represented using X and Y coordinates, and this function computes the Euclidean distance between the $j^{th}$ rows of $\chi_{p,j}^t$ and $\chi_{q,j}^t$. Here, $t$ varies from 1 to 9 (i.e., the bumper, grill, hood, windshield, roof, rear windshield, trunk-1, trunk-2, and rear bumper lines). $D_{pq}^t$ in Eq. 4 is a distance metric for $S_p$ and $S_q$, similar to $D_{pq}^t$.

A spatial simulated annealing (SSA) technique is employed for sampling car side silhouettes. This approach was originally introduced by van Groenigen and Stein [42] and optimizes spatial environmental sampling schemes [29]. First, $N$ silhouettes are generated via randomly setting 21

4. Proposed system

The proposed system comprises three successive steps, which are iteratively executed and explained in the following sub-sections.

4.1. Silhouette sampling

The silhouettes generated should be evenly distributed in a predefined silhouette design space. In this way, the training data for the drag coefficient prediction (i.e., machine learning) step can be formed well so that a reliable mathematical model can be obtained in a short time. The objective is to generate a desired number of silhouettes that are regularly (uniformly) distributed in the silhouette design space. A characteristic line is described by a feature vector, $\chi$, in the form of a $3n \times 2$ matrix. $n$ equally spaced points on the characteristic line are first computed. The first $n$ rows in $\chi$ involve the X and Y coordinates for these points. The second and third $n$ rows contain, respectively, the first and second order derivatives at the sampled points.

The cost function $E$ in Eq. 2 should be minimized and favors the generation of silhouettes with different geometries based on the potential energy of Audze and Eglais [41,28]. There are two terms in $E$, as follows: $E$ and $E$ (Eq. 3 and 4, respectively). $E$ favors the generation of N silhouettes that are distinct from each other, while $E$ allows the generation of the $N$ silhouettes that are distinct from the $M$ silhouettes obtained from the previous runs of the sampling algorithm. Note that an algorithm run consists of a single run of the silhouette sampling. As stated above, the algorithm runs these consecutive steps until a stopping criterion is met. In the first algorithm run, the term $E$ can be omitted, as there are no silhouettes.

$$E = E + E$$  \hspace{1cm} (2)

$$E = \sum_{p=1}^{N-1} \sum_{q=p+1}^{N} \sum_{t=1}^{9} \frac{1}{D_{pq}^2}$$  \hspace{1cm} (3)

$$E = \sum_{p=1}^{N} \sum_{q=1}^{M} \sum_{t=1}^{9} \frac{1}{D_{pq}^2}$$  \hspace{1cm} (4)

$$D_{pq}^t = \sqrt{\sum_{j=1}^{3n} d^2(\chi_{p,j}^t,\chi_{q,j}^t)}$$  \hspace{1cm} (5)

Let $S_p$ and $S_q$ be two different silhouettes among the $N$ silhouettes, $(S_1,S_2,\ldots,S_N)$, to be generated. And $S_r$ represents a silhouette obtained from the previous runs of the silhouette sampling algorithm. $\chi_{p,j}^t/\chi_{q,j}^t/\chi_{r,j}^t$ denotes the $j^{th}$ row of the feature vector for the $t^{th}$ characteristic line of $S_p/S_q/S_r$, respectively. $D_{pq}^t$ in Eq. 5 is based on the function $d(\chi_{p,j}^t,\chi_{q,j}^t)$. Each row of the feature vector is represented using X and Y coordinates, and this function computes the Euclidean distance between the $j^{th}$ rows of $\chi_{p,j}^t$ and $\chi_{q,j}^t$. Here, $t$ varies from 1 to 9 (i.e., the bumper, grill, hood, windshield, roof, rear windshield, trunk-1, trunk-2, and rear bumper lines). $D_{pq}^t$ in Eq. 4 is a distance metric for $S_p$ and $S_q$, similar to $D_{pq}^t$.

A spatial simulated annealing (SSA) technique is employed for sampling car side silhouettes. This approach was originally introduced by van Groenigen and Stein [42] and optimizes spatial environmental sampling schemes [29]. First, $N$ silhouettes are generated via randomly setting 21
design parameters for the silhouettes. Next, we apply an SSA approach for obtaining the silhouettes that minimize the cost function $E$. The initial temperature $T$ is set to 80, and the algorithm stops when it becomes less than 1.0. $R$ denotes the shortening rate for $T$, which is set to 0.99. The Markov chain length $L$ (i.e., iterative times at the same temperature level) is set to two times greater than $N$.

In each iteration, a silhouette, $S_p (1 \leq p \leq N)$, is randomly selected, which is then regenerated by randomly setting its $n_w$ number of design parameters. After drop in temperature, the parameter $n_w$ decreases. $n_w$ has an integer value by rounding the value obtained by multiplying the total number of design parameters (21) by the shortening rate $R$ (in the SSA algorithm) to the smallest integer value. As the SSA temperature drops, a smaller number of design parameters is chosen to modify the silhouette. After the Markov chain (i.e., $2N$ times silhouette generations), the silhouette set minimizing the cost function $E$, is selected. The temperature, $T$, is then decreased, and the new silhouette set is again searched until the SSA temperature drops to 0.1.

4.2. CFD model generation

The challenges in CFD analyses are the long durations of analyses and the difficulty of predicting the accuracy of the results obtained. In the current study, turbulent flow around 1000 different car side silhouettes were modeled with an acceptable accuracy to generate a reliable $C_D$ prediction model. To achieve this goal in a feasible time, the model details, such as control volume dimensions, duration of simulation, boundary and initial conditions, turbulence model, wall function, order of discretization, time and mesh resolutions were selected carefully.

4.2.1. Selection of solution method, boundary conditions, control volume dimensions, and turbulence parameters

Many experimental [33,43,44] and numerical [33,34,45,46] studies have been conducted in the last decade about flow around a simplified car geometry, called Ahmed Body, to serve as a benchmark problem [33]. The dimensions of the computational domain, boundary conditions, turbulence model, and wall function are selected according to the reported results of flow around the Ahmed Body in the literature and turbulence test cases conducted in this study on a 2D Ahmed Body.

The Ansys Fluent finite volume-based commercial software is used as the flow solver. The nature of the flow is transient due to vortex shedding in the wake of the silhouettes. Transient solution is preferred in order to achieve better convergence for all cases and to increase the accuracy. The length and height of the bounding box of the biggest car silhouette is selected as unit dimensions for 5$m$ and 2$m$, respectively. The control volume sizes are then selected as $16L \times 5H$, as suggested by Ercoftac [45]. A uniform inlet velocity of $40m/s$ and $0Pa$ constant pressure outlet value is selected as the boundary conditions at the inlet and outlet boundaries, respectively. A symmetry boundary condition is applied at the top boundary. The first 5$m$ of the bottom surface is treated as a slip wall, while the no-slip boundary condition is selected for the remaining ground surface and surface of the car silhouette. It should be noted that the resulting geometries after extruding the two-dimensional silhouettes should represent the actual three-dimensional car models appropriately. Concerning the large empty space between the wheels of an actual car, the car side silhouettes cannot be extruded without excluding the tires [33]. In the present study, the wheels are omitted from the CFD tests and the bottom surface of the car silhouettes are closed using flat surfaces with no-slip wall boundary condition. Fig. 5 shows the computational domain with the reference mesh and boundary conditions used in this study.

Liu and Moser [45] applied various RANS models for the 3D Ahmed Body simulations and stated that the standard $k-\epsilon$ model gives a solution at the 5% error band. To find the suitable wall function and order of spatial discretization, the turbulent flow over a 2D Ahmed Body problem is modeled using several different turbulence models with the suggested $y+$ values. In this test and all simulations, the SIMPLE algorithm (a widely used numerical procedure in CFD) is selected as pressure-velocity coupling, and the convergence criterion is selected as $1e-4$ for all flow variables. The list and results of the turbulence test cases are shown in Table 2 and Figure 6.

It can be seen from Figure 6 that the standard $k-\epsilon$ model yields a closer result to the experimental value of 0.299 [43,44] than the realizable $k-\epsilon$ model. Since the enhanced wall treatment requires more cells than the other wall functions in the vicinity of the walls, the predicted $C_D$ values change 2% compared to each other and the first-order spatial discretization yields inaccurate results, second-order
Table 2
Turbulence test cases.

<table>
<thead>
<tr>
<th>Model</th>
<th>Wall Function</th>
<th>$y^+$</th>
<th>$C_D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard $k$-$\epsilon$</td>
<td>Standard</td>
<td>30-120</td>
<td>0.316</td>
</tr>
<tr>
<td>Standard $k$-$\epsilon$</td>
<td>Non-Equilibrium</td>
<td>30-120</td>
<td>0.311</td>
</tr>
<tr>
<td>Standard $k$-$\epsilon$</td>
<td>Enhanced Wall Treatment</td>
<td>1-5</td>
<td>0.309</td>
</tr>
<tr>
<td>Realizable $k$-$\epsilon$</td>
<td>Standard</td>
<td>30-120</td>
<td>0.243</td>
</tr>
<tr>
<td>Realizable $k$-$\epsilon$</td>
<td>Non-Equilibrium</td>
<td>30-120</td>
<td>0.235</td>
</tr>
</tbody>
</table>

4.2.2. Mesh and time step size dependency tests

A two-dimensional, conformal, block-structured, multi-body mesh is generated consisting entirely of quad cells around a randomly selected car silhouette. The silhouette represents whole CFD cases in the mesh dependency context, since the gradients of flow variables especially in the vicinity of the walls are similar for all the silhouettes due to the same free stream inlet velocity is applied in all cases. The mesh dependency tests are not only conducted by changing the size of all the cells by a constant ratio, but also the mesh is parametrized as the tangential mesh size on the silhouette surface, first cell height and growrate in the normal direction of the wall boundaries. All these parameters are investigated separately. The tested values of these parameters are listed and various combinations are investigated to find the most suitable mesh and time step size. The list of values for the tested mesh parameters and time step sizes are shown in Table 3.

The selected parameters are shown in bold in Table 3 and form the reference mesh and time step size. Figure 7 shows the results of the mesh and time step size dependency tests.

Table 3
List of values for the tested mesh parameters and time step sizes.

<table>
<thead>
<tr>
<th>Tangential Mesh Size [m]</th>
<th>Grow Rate</th>
<th>First Cell Heights [m]</th>
<th>Time Step Size [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>1.04</td>
<td>0.001</td>
<td>0.0002</td>
</tr>
<tr>
<td>0.02</td>
<td>1.06</td>
<td>0.002</td>
<td>0.0004</td>
</tr>
<tr>
<td>0.04</td>
<td>1.08</td>
<td>0.004</td>
<td>0.0008</td>
</tr>
<tr>
<td>0.08</td>
<td>1.1</td>
<td>0.008</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 6. Effect of order of spatial discretization on the standard $k$-$\epsilon$ turbulence model with the standard wall function.

spatial discretization and the standard $k$-$\epsilon$ turbulence model with a standard wall function are selected.

Fig. 7. (a) Mesh and (b) time-step size dependency test results.

Each line illustrates the effects of these parameters on the number of cells and $C_D$ values. The remaining two mesh parameters are kept at the reference values, while the related parameter is changed. All three lines converge to a certain value as the number of cells increases, and a parameter is set if the relative change is below a certain limit concerning the computational cost of all 1000 cases. All the mesh dependency tests are conducted using the smallest time-step size, and the time step size test is considered using the reference mesh.

From these tests, 0.0002s is selected as the time step-
size, while mesh independency is achieved at 130000 cells, where the tangential mesh size, first cell height, and growth rate are selected as 0.02$m$, 0.001$m$ and 1.06, respectively. The resulting mesh is shown in Figures 5 and 8.

![Reference mesh around a car silhouette.](image)

Fig. 8. Reference mesh around a car silhouette.

Finally, the duration of the simulation is selected by examining the time history of the $C_D$ values. The $C_D$ values converge around 1$s$; however, the duration of the simulations is selected as 2$s$ concerning extreme car silhouettes, and the reported $C_D$ values are the averages of the last 0.25 $s$ for each simulation.

4.2.3. Autonomous generation of meshes and CFD models

The CFD part involves the steps of geometry and mesh generation, preparation of the case setup, running the simulation, and post-processing. These steps are carried out autonomously, as it is not easy to handle 1000 cases (silhouettes) manually, and avoid making mistakes. Once $N$ car silhouettes are generated, the coordinates of five evenly distributed points on each line are stored to create Ansys Workbench and Fluent journals. These journals are then used to generate the geometry, mesh, and Fluent cases with the desired setup. When all $N$ cases are generated, the simulations run on multiple computers simultaneously.

Autonomously generated mesh consists of several rules. In the geometry step, the unit normal vectors are calculated at the joints of each line of the car silhouette and scaled by a constant parameter to create the first ring of 10 points around the silhouette. The second ring of points is produced at the outer boundaries of the control volume, matching the first ring points to create a suitable mesh. Using these silhouette-specific points, multi-body structured mesh parameters are calculated, and cases are generated.

4.3. Drag coefficient prediction

The control point coordinates, $(x_1, x_2, \ldots, x_{52})$ (see Fig. 3), are employed in this step to represent the car side silhouette. The $X$ and $Y$ coordinates of the first control point for the bumper line are denoted by $(x_1, x_2)$. $(x_3, x_4, \ldots, x_{52})$ are defined in a similar manner; they are also the $X$ and $Y$ coordinates of the other CPs. Note that the end point of a characteristic line and start point of its following line are coincident, and therefore, a single control point is used for both. For example, $(x_{13}, x_{14})$ represents the end point of the grill line and the start point of the hood line.

In this study, a reliable mathematical model is developed to predict drag coefficients using 52 control point coordinates. The proposed two-step procedure requires the implementation of principal component analysis [47] and three machine learning algorithms including linear regression [48], lasso regression and neural network models. Principal components are obtained from non-linear transformations of the original silhouette coordinates (variables). As a non-linear relationship is expected between the $C_D$ and variables, non-linear terms are generated by all pairwise interactions of original variables up to 3rd order polynomial expansion ($e.g., x_1 * x_2, x_1^2, x_1^3$). Thus, the number of variables that enter the linear regression model is increased from 52 to approximately 1271, resulting in multicollinearity due to the existence of high correlations between the newly generated variables. Multi-collinearity occurs when the independent variables are too highly correlated with each other. To convert correlated variables into a set of values of linearly uncorrelated variables, called principal components (PCs), a statistical procedure called principal component analysis (PCA) is applied. This transformation also helps in reducing newly generated variables to a few PCs that account for as much of the variability in the data as possible. As the assumption of variable independency in linear regression is now satisfied, a multiple linear regression model is created with a predefined number of PCs, and the results are obtained. With the presence of non-significant variables, a new model including only those PCs with small p-values (typically $\leq 0.05$) is built and rerun to obtain the final coefficient values of the significant variables.

The mathematical model obtained with this procedure can be seen in Eq. 6.

$$0.318929 - 0.000724 \cdot PC_2 - 0.000259 \cdot PC_3 +$$
$$0.001831 \cdot PC_4 + 0.002367 \cdot PC_5 +$$
$$0.000415 \cdot PC_6 - 0.000395 \cdot PC_8 +$$
$$0.000538 \cdot PC_{14} - 0.000474 \cdot PC_{15} +$$
$$0.001209 \cdot PC_{16} - 0.004002 \cdot PC_{17} -$$
$$0.002449 \cdot PC_{18} - 0.002768 \cdot PC_{19} +$$
$$0.002798 \cdot PC_{20} + 0.005030 \cdot PC_{21} -$$
$$0.005352 \cdot PC_{22} - 0.001446 \cdot PC_{23} -$$
$$0.004349 \cdot PC_{26}$$

As indicated in the previous paragraph, lasso regression and neural network model are also developed and tested. Least absolute shrinkage and selection operator (lasso) regression method performs both variable selection and regularization simultaneously. It improves the prediction error by reducing potential overfitting. Nevertheless, an artificial neural network is an interconnected group of nodes and process complex data inputs. We tested different feedforward neural network structures and identified the best fit based on one input, one hidden and one output layers. Network includes four nodes in the hidden layer and uses
a non-linear sigmoid activation function.

The model results are tested for accuracy using two approaches. First, in-sampling error is measured on the complete dataset using the mean square error (MSE). This measures the average of the squares of the errors or deviations, that is the difference between the actual result and what is predicted. Second, a model validation technique called cross-validation is used to assess how the results of our model generalize to an independent data-set. One usually wants to avoid overfitting and estimate how accurately a predictive model will perform with unknown data. Thus, our complete dataset is partitioned into five subsets, performing the analysis on four subsets and validating the model on one subset. We repeat this process five times; thus the model created in each round is tested for accuracy once in each subset. MSE results are then combined (i.e., averaged) over the rounds to estimate the performance of the final predictive model. Prior to this step, principal component analysis is performed only for neural network and linear regression models. As indicated earlier, the lasso regression technique handles regularization by its nature so PCA step is not necessary. 1271 variables obtained after non-linear transformations of original variables are used in the model all at the same time. Additionally, PCA performed only on training set was used to create PC scores of test set in cross-validation. Therefore, test set was standardized using training mean and standard deviation and then projected onto the subspace defined by 27 principal components of training set.

Similar to cross-validation procedure, when predicting the drag coefficient of new Sedan car side silhouettes, it is necessary to standardize the original variables (i.e., $\text{std}(x_i)$) and calculate PC scores on the subspace already defined by existing data. PCA helped to reduce dimensionality from 1271 original variables to 27 PCs. Due to space limitation, we only included 2 out of 27 PCA equations in Eq. 7 and 3 terms within each equation out of 1271 total terms. Note that each PC has 1271 coefficients associated with the original 1271 variables. The whole PCs can be found in the supplementary material. Once the PCs’ values are determined, the model coefficients can be multiplied by the corresponding PCs and summed up to obtain the drag coefficient.

$$PCA_1 = -0.0000236 \cdot \text{std}(x_3) + 0.0037610 \cdot \text{std}(x_4) + \ldots + 0.0388814 \cdot \text{std}(x_{51})$$

$$PCA_{27} = -0.0017478 \cdot \text{std}(x_3) + 0.0005298 \cdot \text{std}(x_4) + \ldots + 0.0986195 \cdot \text{std}(x_{51})$$

In this section, the results for the car side silhouette sampling and mathematical model for the drag coefficient prediction are first discussed. Following this, the CFD model and its reliability are investigated. The computational time for the techniques used and the stopping criterion of the proposed system are then mentioned. Finally, three generative design test cases are introduced, in which optimum car side silhouettes are obtained based on different objectives using an optimization technique.

Fig. 9. (a) The temperature $T$ drops and the number $n_{sw}$ of changed design parameters decreases as the number of algorithm iterations increases. Here, 1-100 stands for the first algorithm run to generate the silhouettes between the first and 100th ones. (b) The energy $E$ decreases rapidly in the first 15 iterations and then decreases more slowly. The energy for each subsequent run is larger than that of the preceding run.
Fig. 10. Potential energies for the five silhouettes generated: (a) \( E = 1.8 \), (b) \( E = 5.6 \), (c) \( E = 16.5 \), and (d) \( E = 47.9 \). The silhouettes look different from each other when \( E \) is lower (a), whereas they look like each other if \( E \) is higher (d).

5.1. Car side silhouette sampling results

100 silhouettes are generated in each execution of the silhouette sampling algorithm. In other words, \( N \) is set to 100 for the algorithm runs. Fig. 1 shows the sampled silhouettes for the first (a), second (b), fourth (c), fifth (d), and sixth (e) runs. The sampling algorithm is executed ten times. Each run between the first and eighth runs has 437 iterations, and the temperature drops in each run. Fig. 9 (a) depicts the temperature and number of changed design parameters versus the iteration numbers. In addition, Fig. 9 (b) shows the energy (\( E \)) values after the iterations, which decreases quickly in the first 15 iterations and has a slight reduction thereafter. Therefore, only 30 iterations are performed for the ninth and tenth runs so that lesser processing times are required for them. Additionally, the energy in the next run is always greater than that of the previous run as the number of previously obtained silhouettes (the parameter \( M \) in Eq. 4) becomes larger.

To compare the sampled silhouettes with lower and higher potential energies, five silhouettes are generated randomly and using the SSA approach. Recall that the silhouettes having lower potential energy are expected to differ from each other more. Fig. 10 shows silhouettes that are ordered from above (a) to below (d) based on their potential energies (lower and higher, resp.). When the silhouettes in Figure 10 (d) is analyzed, the silhouettes look similar to each other (i.e., particularly the first, second, fourth and fifth one), therefore having higher potential energy \( (E = 47.9) \). On the other hand, the silhouettes in Figure 10 (a) looks different from each other as the potential energy \( (E = 1.8) \) is lower in this case. There are short, medium-sized and long silhouettes. Furthermore, the characteristic lines of the silhouettes look different (i.e., see hood and front bumper lines). The potential energies for the silhouettes in Figure 10 (b) and (c) are, respectively, 5.6 and 16.5.

5.2. Investigation of the CFD model

In this subsection, drag coefficients of 1000 different car silhouettes were calculated using CFD simulations. The \( C_D \) values of these cases varied in the range of 0.18 – 0.55, and among these cases, the results for the silhouettes with the lowest and highest \( C_D \) can be observed that the silhouette with the lowest \( C_D \) looks more streamlined and has softer edges. Furthermore, as expected, the wake behind the car is narrower, and the stagnation zone in the leading faces is smaller compared with the silhouette with the highest \( C_D \).

Fig. 11. Velocity contours of silhouettes with the lowest (a) and highest (b) \( C_D \)s.

5.3. Reliability of the mathematical model for the drag estimation

In this subsection, the two-step procedure mentioned in Sec. 4.3 is applied to the silhouette training data generated by the sampling algorithm and through the CFD simulations. Starting with the initial 100 silhouettes, the data are repetitively extended with the next hundred silhouettes to create ten data sets with differing sizes (sample size changing from 100 to 1000). The tests are then executed separately on each of the data-sets, and the mean square error (MSE) values on both complete and cross-validated test data are compared. The results show that the average squared errors reduces with the increasing sample size (Fig. 12 (a)). In other words, more observations help to obtain a more precise estimate of a drag coefficient. In addition, as expected, in-sampling errors obtained on complete data were lower than the errors on the test data. The tests are then executed separately on each of the data-sets, and the MSE values on both complete and cross-validated test data are compared for the machine learning algorithms.

To understand true statistics behind data, a simulation exercise was performed on 1000 bootstrap samples (split into train/test sets), resulting in 1000 different models,
model coefficients and test errors. Using the pool of obtained results, 95% confidence intervals were constructed to identify the range of values parameters lie in. It can be concluded that values of regression coefficients and test errors range within small intervals (see Table 4).

As a result of the non-linear transformation of original variables, multi-collinearity is introduced in the regression model. To obtain uncorrelated PCs and use them in the drag coefficient prediction model, PCA is initially performed, and the variation that each PC captures is shown in Table 5. As the marginal benefits obtained beyond 20 PCs are very minimal, we proceed with 27 PCs that account for about 97% of the total variability. Additionally, correlations between PCs and control point coordinates obtained from PCA are used to explain the role of each PC in shape deformation. Size of our correlation matrix for the PC and control point variables is 27 by 1271. Therefore, we only listed top 10 highly correlated variables with 27 PCs for the sake of simplicity. Table 5 summarizes the correlations between PCs and control point coordinates. PC1, which captures 32.9% variance, is mostly correlated with $x_{39}$ (end/start X control point coordinate of trunk-1/trunk-2 line) followed by $x_{28}$ (third Y control point coordinate of roof line), $x_{30}$ (end/start Y control point coordinate of roof/rear windshield line) and $x_{33}$ (end/start X control point coordinate of rear windshield/trunk line). PC2 captures 10.9% variance, and is mostly correlated with $x_{13}$, which is end/start Y control point coordinate of grill/roof line. Finally, a regression model is developed with these PCs, and model results are obtained.

Table 4: 95% confidence intervals, $\hat{\beta}_{\text{Intercept}}, \hat{\beta}_{\text{PC1}}, \ldots, \hat{\beta}_{\text{PC27}}$ and test error, (lower 2.5% and upper 2.5%), respectively, for intercept, regression coefficients and test error.

<table>
<thead>
<tr>
<th>PC</th>
<th>Lower 2.5%</th>
<th>Upper 2.5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>0.314663631</td>
<td>0.323392948</td>
</tr>
<tr>
<td>$\beta_{\text{PC1}}$</td>
<td>-0.000161452</td>
<td>0.000265166</td>
</tr>
<tr>
<td>$\beta_{\text{PC2}}$</td>
<td>-0.001118327</td>
<td>-0.000347733</td>
</tr>
<tr>
<td>$\beta_{\text{PC3}}$</td>
<td>-0.000920032</td>
<td>0.000460364</td>
</tr>
<tr>
<td>$\beta_{\text{PC4}}$</td>
<td>-0.000978557</td>
<td>0.000397555</td>
</tr>
<tr>
<td>$\beta_{\text{PC5}}$</td>
<td>-0.000920084</td>
<td>0.0003062676</td>
</tr>
<tr>
<td>$\beta_{\text{PC6}}$</td>
<td>-0.001371101</td>
<td>0.000241016</td>
</tr>
<tr>
<td>$\beta_{\text{PC7}}$</td>
<td>-0.001381465</td>
<td>0.0001610846</td>
</tr>
<tr>
<td>$\beta_{\text{PC8}}$</td>
<td>-0.001251251</td>
<td>0.000755903</td>
</tr>
<tr>
<td>$\beta_{\text{PC9}}$</td>
<td>-0.000894357</td>
<td>0.001108548</td>
</tr>
<tr>
<td>$\beta_{\text{PC10}}$</td>
<td>-0.001044688</td>
<td>0.001081749</td>
</tr>
<tr>
<td>$\beta_{\text{PC11}}$</td>
<td>-0.001152374</td>
<td>0.00107264</td>
</tr>
<tr>
<td>$\beta_{\text{PC12}}$</td>
<td>-0.001083512</td>
<td>0.000165715</td>
</tr>
<tr>
<td>$\beta_{\text{PC13}}$</td>
<td>-0.001144169</td>
<td>0.000181219</td>
</tr>
<tr>
<td>$\beta_{\text{PC14}}$</td>
<td>-0.001251413</td>
<td>0.0001391584</td>
</tr>
<tr>
<td>$\beta_{\text{PC15}}$</td>
<td>-0.001580638</td>
<td>0.000451524</td>
</tr>
<tr>
<td>$\beta_{\text{PC16}}$</td>
<td>-0.00201632</td>
<td>0.000399952</td>
</tr>
<tr>
<td>$\beta_{\text{PC17}}$</td>
<td>-0.005509261</td>
<td>0.000326234</td>
</tr>
<tr>
<td>$\beta_{\text{PC18}}$</td>
<td>-0.00512252</td>
<td>0.000349527</td>
</tr>
<tr>
<td>$\beta_{\text{PC19}}$</td>
<td>-0.005658036</td>
<td>0.0004577668</td>
</tr>
<tr>
<td>$\beta_{\text{PC20}}$</td>
<td>-0.006188652</td>
<td>0.0005914507</td>
</tr>
<tr>
<td>$\beta_{\text{PC21}}$</td>
<td>-0.006755264</td>
<td>0.0006749884</td>
</tr>
<tr>
<td>$\beta_{\text{PC22}}$</td>
<td>-0.006926406</td>
<td>0.000538633</td>
</tr>
<tr>
<td>$\beta_{\text{PC23}}$</td>
<td>-0.004754339</td>
<td>0.00016321</td>
</tr>
<tr>
<td>$\beta_{\text{PC24}}$</td>
<td>-0.00252471</td>
<td>0.0002549768</td>
</tr>
<tr>
<td>$\beta_{\text{PC25}}$</td>
<td>-0.004823371</td>
<td>0.0004302823</td>
</tr>
<tr>
<td>$\beta_{\text{PC26}}$</td>
<td>-0.005907878</td>
<td>0.0004914241</td>
</tr>
<tr>
<td>$\beta_{\text{PC27}}$</td>
<td>-0.004848325</td>
<td>0.0004283171</td>
</tr>
<tr>
<td>Test Error</td>
<td>0.001815209</td>
<td>0.00238847</td>
</tr>
</tbody>
</table>

Fig. 13. Frequency of error range on $C_D$ prediction.
5.4. Alternative machine learning methods

Lasso regression [49] and neural networks are also employed to obtain a mathematical model for the CDS prediction. Fig. 12 shows in-sampling and cross validation mean square errors (MSEs). The results show that the MSE values on both in-sampling and cross-validated data are very similar for the linear and Lasso regressions when the silhouette size is greater than 400. While the MSE values on in-sampling data for the neural networks approach are lower than those of linear and Lasso regression methods, these values on cross-validated data are greater for the neural networks approach.
ral networks. As neural networks tend to overfit, its cross-validation error is higher than other two methods. Thus, we propose to use linear regression model with variables up to third degree polynomial order when predicting $C_D$ values of silhouettes. We also compared the mean, median and maximum of the absolute percent errors for the linear regression, Lasso regression and neural networks approaches. For the silhouettes having $C_{DS}$ between 0.25 and 0.4, the means and medians for the absolute percent errors in $C_{DS}$ are between %5 and %10 as shown in Figure 14 (a) and (b). All machine learning approaches have quite similar means and medians for the silhouettes having $C_{DS}$ less than 0.4. For the silhouettes with $C_{DS}$ greater than 0.4, the neural networks have less mean and median $C_D$ errors compared to those of regression methods. The mean and median errors for the overall silhouettes are similar for the whole methods. When the maximum absolute percent errors in $C_{DS}$ are compared, the neural network approach has about %50 error, which is less than other approaches (%60-%65 for the regression models).

$$\text{Fig. 15. When the number of sampling algorithm runs increase, the processing time increases linearly. For the models between 801\textsuperscript{th} and 1000\textsuperscript{th}, only few iterations are performed.}$$

5.5. **Computational time**

A personal computer with an Intel Core i7 4820 3.7 GHz processor and 8 GB memory is used for the experiments of the sampling algorithm, and the implementation is single-threaded. The computational time depends on several parameters. The SSA algorithm stops when the temperature $T$ becomes less than the temperature threshold, which is set to 1.0. If it is set to a smaller/larger value, the processing time is longer/shorter. The shortening rate $R$ is set to 0.99. If it has a smaller/larger value, the computational time will be shorter/longer. The feature vector for a characteristic line is formed by $n$ equally spaced points on the characteristic line, which is set to 10 in this work. Setting it to a larger value will increase the computational cost. The sampling algorithm also depends on the number $N$ of silhouettes sampled in the current sampling stage and that ($M$) in the previous stages. Fig. 15 shows that the processing time increases almost linearly as the number of runs increases. However, one can also stop the SSA iterations earlier—after 15 steps—, as the energy decreases quickly following the previous analysis based on Fig. 9 (b). Therefore, only few iterations were carried out to obtain the models between 801\textsuperscript{th} and 1000\textsuperscript{th}.

The drag coefficient prediction is not a computationally expensive process, given that the maximum numbers of variables in PCA and linear regression are 1271 and 27, respectively. In the case of all 1000 observations, the R software runs for a total time of 6.94 seconds, including 3.68 s for PCA, 0.02 s for linear regression on the complete data-set, and 3.24 s for cross-validation of the regression model. In addition, the $C_D$ prediction using the obtained mathematical model for a given silhouette takes less than a second.

For the CFD simulations, computations were carried out simultaneously at multiple computers. Since there were 1000 cases to run and the duration of one simulation was relatively short, the CFD simulations were not run parallel; instead, several were run simultaneously. For a personal computer with an Intel Core i7 4820 3.7 GHz processor, a CFD simulation of a single case took approximately 46 hours.

5.6. **$C_D$ errors versus number of algorithm runs**

The proposed system have three successive steps that are iteratively executed until a mathematical model with a desired level of $C_D$ prediction accuracy is obtained. Actual $C_D$ for a silhouette in the training data and its predicted $C_D$ has already known after the machine learning step. The percent errors in $C_{DS}$ can be computed using actual and predicted $C_{DS}$. Fig. 16 (a) shows means and medians of the absolute percent errors in $C_{DS}$ after each algorithm run, in which the silhouette size changes between 100 and 1000. We have seen decreases in their values except when the sample size increases from 100 and 200. They both became less than 10% when the sample size was 900 and 1000. Based on the needs of a person/company, the silhouette size can be increased more so that the $C_D$ errors can decrease further. Fig. 16 (b) shows the mean of the percent errors in $C_{DS}$ with the error bars (showing their standard deviations) after the algorithm runs. The means of percent errors in $C_{DS}$ got close to zero and their standard deviations decreased as the sample size increased.

5.7. **Generative design test cases**

In engineering design processes, the lower and upper bounds for the design parameters (such as characteristic line widths and lengths) are first determined. The parameter values for the designs generated by the designers/computers lie mostly within these bounds. In this work, the silhouettes are uniformly sampled as much as
possible in the design space (formed by the design parameter bounds) by means of Audze and Eglais potential energy [41,28]. As the silhouettes obtained in the silhouette sampling stage spread regularly as much as possible in the design space, the $C_D$ of a newly arrived silhouette (that is also in the established design space) is expected to be predicted well. However, if we look from a generative design perspective, particularly when a designer tunes the design space by changing lower and upper bounds, the mathematical model may not predict $C_D$ well for the silhouette if it is located outside the design space in which silhouette sampling and machine learning steps are carried out. It should be noted that another usage of generative design is to generate quickly high-performing design alternatives from an idea. Here, such generative design cases are provided in which silhouette alternatives are automatically generated in the design space where the silhouette sampling and linear regression are performed.

5.7.1. Test case-1

A silhouette is represented using 21 design parameters, and millions of silhouettes can be generated by setting different values to these design parameters. The objective is to find the silhouette with the lowest drag based on the design specifications. As described previously, performing a CFD analysis for a silhouette takes about two days; thus, it is impractical to carry out such analysis in an autonomous generative design step. Therefore, the mathematical model obtained in Sec. 4.3 is employed for the silhouette optimization. The car length $L$ is determined by the user, which can be considered as design specification: $4m \leq L \leq 5m$. Furthermore, the car height $H$ is restricted and should be smaller than $1.2m$.

A similar to the SSA approach explained in Sec. 4.1 is also employed here to generate the silhouette with the lowest drag based on the specifications mentioned above. Design parameters of the silhouette are modified randomly until the temperature drops to the specified temperature threshold (i.e., 1.0). The parameters are set to the same values with the SSA approach. However, here, the Markov chain involves 10 silhouette generations. The objective function to be minimized is the $C_D$ value for the silhouette. In total, 437 iterations are performed, in which 43700 random models are investigated. The total processing time was 17.4 seconds. Figure 17 (a) shows the silhouette found with the drag coefficient of 0.145. Figure 18 (a) provides a graph of the drag coefficient versus number of iterations. When the temperature drops to 20, few changes occur in the drag coefficient. The silhouette with the highest drag (0.45) is also tested using the SSA approach, as can be seen in Fig. 17 (b). Here, the objective function used is $1$ over the $C_D$ value for the silhouette. The computational time for this process was 18.2 seconds.

![Figure 17](image)

Fig. 17. Silhouette with the lowest (a) and highest (b) drag coefficients generated using the SSA approach.

5.7.2. Test case-2

As a second test case scenario, a desired number of (20) space-filling (i.e., a lower Audze and Eglais potential energy [41,28]) silhouettes are generated, where the drag coefficients are less than 0.3. The SSA approach is employed, where the Markov chain involves two times silhouette (i.e., $10N$) generations. The objective function is the sum of two terms as shown in Eq. 8. The first one is the Audze and Eglais potential energy. The second one favors the generation of the silhouettes the drag coefficients less than 0.3. Here, $C_D^p$ denotes the drag coefficient for the silhouette $p$. The function $T(a)$, in which $a$ represents the $C_D$ value for the silhouette, penalizes the silhouette with the undesired $C_D$ value.

$$E = \sum_{p=1}^{N-1} \sum_{q=p+1}^{N} \sum_{t=1}^{9} \frac{1}{D_{pq}^{t}} + \sum_{p=1}^{N} T(C_D^p)$$  (8)
where
\[
T(a) = \begin{cases} 
0 & \text{if } a < 0.3; \\
1000 \times a & \text{else.}
\end{cases}
\]

The generated 20 silhouettes with their predicted drags can be seen in Figure 19. The computational time was 28077.3 seconds. Fig. 18 (b) depicts a graph of drag coefficient versus SSA temperature. When the temperature dropped to 2, few changes occurred in the drag coefficient.

5.7.3. Test case-3

Three silhouettes were separately sampled using the SSA approach for drag coefficients smaller than 0.25, between 0.33 and 0.35, and greater than 0.4. The Markov chain here involved $10N$ silhouette generations. For all these cases, similar objective functions are employed as in Eq. 8. However, the function $T(a)$ is revised as $T'(a)$, $T''(a)$ and $T'''(a)$, respectively, for the first, second and third test scenarios, which are as follows:

\[
T'(a) = \begin{cases} 
0 & \text{if } a < 0.25; \\
1000 \times a & \text{else.}
\end{cases}
\]

\[
T''(a) = \begin{cases} 
0 & \text{if } a > 0.33 \text{ or } a < 0.35; \\
1000/a & \text{else if } a \leq 0.33; \\
1000 \times a & \text{else if } a \geq 0.35.
\end{cases}
\]

\[
T'''(a) = \begin{cases} 
0 & \text{if } a > 0.4; \\
1000/a & \text{else.}
\end{cases}
\]

Fig. 20 shows the silhouettes generated for the test scenarios. It can be observed that the silhouettes with the drag coefficient greater than 0.4 had geometries far from the streamlined geometry, while the others were much more closer to this geometry. The computational time for the first, second and third test scenarios were 468.4, 492.4 and 494.8 seconds, respectively.

Finally, Figure 18 shows plots for the energy versus number of iterations for the all three generative design test cases. The energies drop in the initial iterations rapidly, while they have little changes in the final iterations. Note that all silhouettes generated in the test cases can be found in the supplementary material.
5.8. Out of sample validation

5.8.1. Validation case-1

A few silhouettes were randomly generated having predicted $C_D$ values changing between 0.21 and 0.51. Their $C_D$ values were then computed using CFD simulations to validate the performance of the $C_D$ prediction model. Table 6 shows the actual and predicted $C_D$ values with their absolute percent error. The average, median and standard deviation for the absolute percent error are 9.59%, 8.84% and 5.83%, resp.

![Figure 19. Twenty distinct models whose drag coefficients are less than 0.3.](image)

Table 6

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>PerErr</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.218</td>
<td>0.226</td>
<td>3.82</td>
</tr>
<tr>
<td>0.220</td>
<td>0.186</td>
<td>15.65</td>
</tr>
<tr>
<td>0.233</td>
<td>0.227</td>
<td>2.59</td>
</tr>
<tr>
<td>0.252</td>
<td>0.305</td>
<td>20.83</td>
</tr>
<tr>
<td>0.265</td>
<td>0.304</td>
<td>14.55</td>
</tr>
<tr>
<td>0.277</td>
<td>0.307</td>
<td>10.77</td>
</tr>
<tr>
<td>0.281</td>
<td>0.305</td>
<td>8.67</td>
</tr>
<tr>
<td>0.294</td>
<td>0.315</td>
<td>7.10</td>
</tr>
<tr>
<td>0.308</td>
<td>0.330</td>
<td>7.15</td>
</tr>
<tr>
<td>0.314</td>
<td>0.316</td>
<td>0.69</td>
</tr>
<tr>
<td>0.361</td>
<td>0.426</td>
<td>17.87</td>
</tr>
<tr>
<td>0.421</td>
<td>0.436</td>
<td>3.48</td>
</tr>
<tr>
<td>0.468</td>
<td>0.411</td>
<td>12.12</td>
</tr>
<tr>
<td>0.502</td>
<td>0.456</td>
<td>9.01</td>
</tr>
</tbody>
</table>

Fig. 20. Three silhouettes with drag coefficients lower than 0.25 (a), between 0.33 and 0.35 (b), and greater than 0.4 (c).

5.8.2. Validation case-2

The optimized silhouettes in the second generative design test case were also validated through CFD simulations. The optimized 20 space-filling silhouettes (i.e., silhouettes that have distinct characteristic lines as much as possible) with drag coefficients less than 0.3 were generated using the SSA approach. Table 7 shows, respectively, the actual and predicted $C_D$ values for these silhouettes. 17 of them have $C_D$ values less than 0.3, two of them have $C_D$ values that are very close to, but greater than, 0.3 (they are 0.306), and one of them has a $C_D$ value of 0.34. Finally, the mean, median and standard deviation for the absolute percent errors in $C_D$ values are 6.22%, 6.00% and 4.45%, resp.

![Figure 20. Three silhouettes with drag coefficients lower than 0.25 (a), between 0.33 and 0.35 (b), and greater than 0.4 (c).](image)

Table 7

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>PError</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.306</td>
<td>0.3</td>
<td>1.96</td>
</tr>
<tr>
<td>0.279</td>
<td>0.299</td>
<td>7.17</td>
</tr>
<tr>
<td>0.340</td>
<td>0.296</td>
<td>12.94</td>
</tr>
<tr>
<td>0.243</td>
<td>0.298</td>
<td>22.63</td>
</tr>
<tr>
<td>0.286</td>
<td>0.285</td>
<td>0.35</td>
</tr>
<tr>
<td>0.284</td>
<td>0.299</td>
<td>5.28</td>
</tr>
<tr>
<td>0.283</td>
<td>0.294</td>
<td>3.89</td>
</tr>
<tr>
<td>0.279</td>
<td>0.293</td>
<td>5.02</td>
</tr>
<tr>
<td>0.306</td>
<td>0.287</td>
<td>6.21</td>
</tr>
<tr>
<td>0.272</td>
<td>0.281</td>
<td>3.31</td>
</tr>
<tr>
<td>0.249</td>
<td>0.299</td>
<td>20.08</td>
</tr>
<tr>
<td>0.266</td>
<td>0.263</td>
<td>1.13</td>
</tr>
<tr>
<td>0.293</td>
<td>0.296</td>
<td>1.02</td>
</tr>
<tr>
<td>0.299</td>
<td>0.288</td>
<td>3.68</td>
</tr>
<tr>
<td>0.291</td>
<td>0.298</td>
<td>2.41</td>
</tr>
<tr>
<td>0.267</td>
<td>0.297</td>
<td>11.24</td>
</tr>
<tr>
<td>0.279</td>
<td>0.296</td>
<td>6.09</td>
</tr>
<tr>
<td>0.277</td>
<td>0.296</td>
<td>6.86</td>
</tr>
<tr>
<td>0.287</td>
<td>0.294</td>
<td>2.44</td>
</tr>
<tr>
<td>0.284</td>
<td>0.282</td>
<td>0.70</td>
</tr>
</tbody>
</table>

6. Conclusions and future works

This paper proposed a design support system to estimate the drag coefficient of a given silhouette. Gunpinar’s [1] car...
side silhouette design technique was utilized to represent the silhouettes. A silhouette sampling method generated silhouettes that were evenly distributed in the silhouette design space by means of adapting Audze and Eglais’s technique [41,28] for the silhouettes. CFD simulations were performed to calculate the silhouette’s $C_D$s. Using the geometries and computed $C_D$s of the silhouettes, a mathematical model was obtained for the $C_D$ prediction of silhouettes by applying PCA and regression/neural network methodologies. These three steps were executed until a reliable mathematical model to some extent was established that could estimate the $C_D$ of a given silhouette correctly within an acceptable error range. Finally, generative design test case scenarios were illustrated to design car side silhouettes with desirable $C_D$s based on the predefined design specifications.

In future research, the work will be extended to 3D car models. In addition, a 3D car body optimization user-centered system, in which CAD and CFD tasks are integrated, will be studied. The user will only notify the system about his/her desires for the car body (such as lengths), and thus, the user-centered 3D car bodies will be produced automatically, with all the engineering tasks carried out by the system. The mathematical model (predicting $C_D$ of a silhouette) obtained after the runs of the silhouette sampling algorithm can be integrated into the sampling process so that silhouettes with desired (i.e., lower) $C_D$s can be oversampled. We would like to also sample a less number silhouettes in each run of the silhouette sampling algorithm, and find out whether fewer total CFD simulations are required to obtain the same desired level of $C_D$ prediction model. Additionally, the use of the local geometric properties for the characteristic lines in line with the control point coordinates in the machine learning step will be investigated. Furthermore, we aim to perform similar CFD work for the yacht hull models generated using by Khan et al. [50].

Finally, some selective silhouettes will be extruded in the depth direction and physically produced. Their experiments will then be carried out in a wind tunnel to increase the reliability of the CFD model obtained in this work.

Acknowledgements

The authors would like to thank Sertac Cadirci for his valuable comments and The Scientific and Technological Research Council of Turkey for supporting this research (Project Number: 315M077).

References


