Comput Mech (2018) 62:23–45
https://doi.org/10.1007/s00466-017-1482-4

ORIGINAL PAPER

Finding the optimal shape of the leading-and-trailing car of a high-speed train using design-by-morphing

Sahuck Oh[1] · **Chung-Hsiang Jiang**[1] · **Chiyu Jiang**[1] · **Philip S. Marcus**[1]

Received: 17 March 2017 / Accepted: 7 September 2017 / Published online: 3 October 2017

Abstract We present a new, general design method, called *design-by-morphing* for an object whose performance is determined by its shape due to hydrodynamic, aerodynamic, structural, or thermal requirements. To illustrate the method, we design a new leading-and-trailing car of a train by morphing existing, baseline leading-and-trailing cars to minimize the drag. In design-by-morphing, the morphing is done by representing the shapes with polygonal meshes and spectrally with a truncated series of spherical harmonics. The optimal design is found by computing the optimal weights of each of the baseline shapes so that the morphed shape has minimum drag. As a result of optimization, we found that with only two baseline trains that mimic current high-speed trains with low drag that the drag of the optimal train is reduced by 8.04% with respect to the baseline train with the smaller drag. When we repeat the optimization by adding a third baseline train that under-performs compared to the other baseline train, the drag of the new optimal train is reduced by 13.46%. This finding shows that bad examples of design are as useful as good examples in determining an optimal design. We show that *design-by-morphing* can be extended to many engineering problems in which the performance of an object depends on its shape.

✉ Philip S. Marcus
pmarcus@me.berkeley.edu

Sahuck Oh
sahucko@gmail.com

Chung-Hsiang Jiang
chjiang@gmail.com

Chiyu Jiang
chiyu.jiang@berkeley.edu

[1] Department of Mechanical Engineering, University of California, Berkeley, Berkeley, CA 94720, USA

Keywords Design-by-morphing · Train head · Optimization · Genetic algorithm · Artificial neural network · Drag reduction

1 Introduction and motivation

Finding the optimal shape of the leading and trailing cars of a high-speed train has recently become an important topic from both a theoretical and practical point of view [24,58]. As the speed of a train increases above $\sim$ 100 mph, there is often a drag crisis in which the aerodynamic drag increases precipitously as does the cost of energy in operating the train. It is estimated that sixty percent of the traction energy of a high-speed train is lost due to its aerodynamic drag and friction and that 8–15% of the traction energy can be saved by reducing the aerodynamic drag by 25% [5]. Another problem of high-speed trains occurs when the long, highly-rotational boundary layer surrounding the train sheds from the trailing car of the train. When the vorticity is shed alternately from the right and left sides of the trailing car, the aerodynamic forces associated with the shedding produce strong sideways motions on and rocking of the train. The rocking is not only an unpleasant experience for passengers, but also can lead to derailment, especially when modest cross-winds are present [4]. Thus, as high-speed trains are developed to have increasingly fast speeds, it is important to find their optimal aerodynamic shapes.

For an *optimally* designed train to be practical, the train not only must have small drag and be stable aerodynamically, but also must be inexpensive to manufacture and maintain and also must be structurally sound. Therefore, a practical design of a shape usually requires multi-objective optimization, rather than single-objective optimization. Furthermore, an optimally designed shape frequently must obey

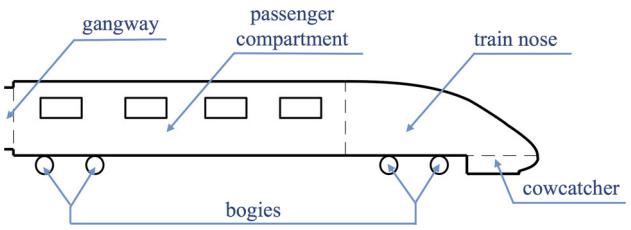


Fig. 1 Nomenclature for the major components of the leading-and-trailing car of a high-speed train. The *head* is the union of the nose and cow-catcher

a set of constraints. For example, the leading and trailing cars of a train must be constrained such that the cars fit on the tracks, pass through tunnels, couple to passenger cars, etc. However, in this paper, to best demonstrate the order-of-magnitude improvements that we can obtain over competing design methods, we limit ourselves to optimizing the leading and trailing car of a train and not the passenger cars between them. Consistent with this approach, we further limit our optimization to only the train head (see Fig. 1), and not the passenger compartment, of the leading and trailing car. We also limit our study to a single-objective optimization in terms of a cost function. Here, we minimize the train's nose's aerodynamic drag at one value of its operating speed, and we use simple constraints that require only that the train's nose smoothly join onto the cow-catcher and passenger compartment. Adding other objectives to the cost function is straight-forward if a cost function can be found that represents those objectives, and applying more geometric constraints can be handled by imposing boundary conditions—see Sect. 8.2 and Oh [43].

Here, we optimize an idealized train that has two properties. First, the leading and trailing cars of the train are identical. This is the norm for high-speed trains that travel "back-and-forth" between destinations with the leading and trailing cars exchanging roles at each terminus rather than having the train turn-around, which requires a large area. Second, for simplicity, the train consists of only two cars: a leading and a trailing car connected by a gangway. In Fig. 1, we define the nomenclature we use for the components of the leading-and-trailing car. The upper front shape of the car is the *nose*; the object underneath the nose is the *cow-catcher*; and the large part behind the nose is the *passenger component*. The train's *head* is the union of the nose and cow-catcher. Behind the passenger component of the leading car is a *gangway* which connects the leading-and-trailing car to the passenger cars (or in the case studied here connects the leading car to the trailing car). The wheel assemblies beneath the nose and passenger compartment are the *bogies*.

In this paper, we take the point of view that from a design perspective, it is important to find the best methodology for creating an optimal shape as well as to find the optimal shape itself. We are interested in exploring design spaces that contain not only designs that have small changes from existing shapes, but also designs that are radical departures. The latter often cannot be found with current design methods (see Sect. 2 for more detail). The main purpose of this paper is to introduce a new design method that we call *design-by-morphing*, which overcomes drawbacks of current methods in optimal design. As discussed below, traditional re-designs of a leading-and-trailing car of a train typically lead to decreases of the aerodynamic drag of 1–3%, (although a recent study using the adjoint method—see Sect. 3.2—found a decrease of $\sim 7\%$). Here using design-by-morphing, we create an optimal design of a train (using only two degrees of freedom) that decreases its aerodynamic drag by more than 13%.

The remainder of this paper is constructed as follows. In Sect. 2, we review how shapes of objects are represented and deformed in traditional methods of design and introduce our method of representing shapes and how we deform them. In Sect. 3, we review traditional methods of optimal design for trains and other objects where the aerodynamic or hydrodynamic shape is important, and in Sect. 4, we outline our new method and philosophy of *optimal-design-by-morphing*. A review of traditional morphing and an outline of the morphing methods we use here along with our way of imposing constraints on the morphed shapes are in Sect. 5. The principle steps in *design-by-morphing* for finding the optimal train design are laid out in Sect. 6, and our results of finding the optimal design are in Sect. 7. Our conclusions and discussion are in Sect. 8.

2 Shape representation and deformation

Methods for shape representation and deformation are essential to the overarching goal of aerodynamic shape optimization. Because a shape has, in some sense, an infinite number of degrees of freedom, in principle, the optimization of a shape is essentially a minimization problem over an *infinite*-dimensional space. Therefore, in a practical design process, the different schemes of shape representation and the different methods for their deformation require a severe reduction in the number of degrees of freedom. Hence, the choices of the *finite*-dimensional shape representation and deformation scheme are of paramount importance.

Regardless of the specifics of any particular problem, all shape optimization problems can be broken into three steps. First, define an engineering goal-driven cost function of the shape that is to be minimized:

$$\min_{\Omega} J(\Omega), \tag{1}$$

where J is the desired cost function and Ω is the shape of interest. Second, parameterize the target shape and represent the shape as a *finite*-length vector of numerical shape-parameters, i.e.:

$$\min_{\mathbf{s}} J(\Omega(\mathbf{s})), \tag{2}$$

where $\mathbf{s}$ is the vector corresponding to the shape in the parametric space of interest. Finally, the problem reduces to the minimization of the cost function in the parametric space. In most applications, the shape also has to satisfy a set of constraints. Therefore, a more general mathematical description of the problem is:

$$\min_{\mathbf{s}} \quad J\ (\Omega(\mathbf{s})) \tag{3}$$

$$s.t. \quad g_j\ (\mathbf{s}) \leq 0 \quad j = 1 \ldots m \tag{4}$$

$$h_k\ (\mathbf{s}) = 0 \quad k = 1 \ldots n, \tag{5}$$

where there are m constraints that can be written in the form of inequalities and n constraints in the form of equalities [10,64]. Depending on how Eqs. 3–5 are executed, existing shape optimization methods can be classified into different classes.

One way of dividing traditional optimization methods into different classes is to note that some methods make *local* changes to the shape by representing the shape locally and using gradient-like methods to make small, incremental changes to the parameter values, so that the "optimal" solution is restricted to being only a local solution in the design space. As an example of a local method, the ANSYS Fluent software package can be required to use free-form deformation to parameterize and deform the shape locally, and then solve for the optimal deformation with gradient or adjoint methods. Other methods attempt to make *global* changes to the shape by representing the shape globally and making large changes to the values of the design parameters using genetic algorithms, simulated annealing, or similar maximization/minimization methods. Another way of dividing traditional optimization methods into different classes is to note that some can use a relatively large dimension (greater than, say, 100) design space while others are limited to a small dimension design space [60]. Generally, local and global changes of the shape are used with high- and low-dimension design spaces, respectively, because spaces with large dimensions cannot easily be optimized with global methods. Therefore, for global methods, choosing the relatively few parameters of the shape representation and deformation is of great importance. In the following sections, we will discuss current popular approaches of shape representation and shape deformation of an object.

2.1 Representation of shapes and their deformations using CAD and polygonal meshes

2.1.1 Representation of shapes

One common approach to shape representation is the use of CAD design parameters. Optimization is then performed by sweeping through the parameter values such that the optimal values are obtained that minimize a cost function. For example, Shojaeefard et al. [53] represented the shape of an Agnew draft tube for a turbine and optimized it with two parameters (degrees of freedom): its height and its attachment angle. In another example, Long et al. [36] represented the shape of a pulsatile ventricular assist device and optimized it with four parameters: the main chamber's diameter, its height, and the attachment angles of its inlet and outlet arms. As two other examples, Demeulenaere et al. [15] optimized the design of a turbine blade, and Demeulenaere et al. [16] optimized a turbocharger compressor wheel by representing the shapes with CAD design parameters. Generally, when optimizing shapes with CAD parameterizations there are a only a small number of design variables (so that there is a small dimensional design space to be searched for optimal values).

Another common method of surface shape representation uses polygonal meshes. A surface is represented as a set of nodes and interconnected polygonal faces. Meshes with triangular and quadrilateral faces are used most frequently, although higher-degree polygons as well as hybrid meshes with mixed degrees of polygons are also used. Theoretically, an infinitely fine polygonal mesh can approximate any continuous surface. However, a fine mesh leads to a large dimensional, and therefore possibly unwieldy, design space. Furthermore, changes to the location of any particular nodal point in a fine mesh induce very local shape changes.

2.1.2 Deformation of shapes

The degrees of freedom associated with shape deformation is generally either the same as or less than the number of degrees of freedom associated with shape representation. For shapes represented with CAD parameters, the number of parameters is small, and generally (unless some constraint on the shape requires that one or more parameters remain fixed) each CAD parameter is allowed to vary, so the number of degrees of freedom of the deformation is equal to the number of degrees of freedom of the shape parameterization. However, when a shape is represented with a mesh, the number of degrees of freedom (generally $3\times$ the number of mesh points) of the shape is so large that the number of degrees of freedom of the deformation is usually much smaller in order to make exploration of the design space computationally manageable. Thus, when deforming a mesh, the method of reducing the number of degrees of freedom is of great importance.

Generally, shape deformation of a mesh is carried out by defining a relatively small (compared to the number of degrees of the shape representation) set of control points. Control points have a specified region of influence, and their use changes the shape locally, rather than globally over the entire shape. Free Form Deformation (FFD) methods [22,49] and NURBS [26] are two common methods of changing shapes with control points. With FFD, a user chooses the locations of control points, and then the volume enclosed by these control points is parameterized and represented as an analytical function. The surface of the object that is to be redesigned is then embedded in this volume. The control points of the embedding volume are then moved, and the geometry inside the volume, including the surface of the object, deforms accordingly. This method has been widely used in aerodynamic shape optimization. As an example, this approach has been used to minimize the drag force on a two-dimensional airfoil [46]. As another example of this approach, FFD was used to minimize the drag on the Common Research Model wing [37] by distributing 720 FFD control points over the wing and varying their positions. Train shapes were optimized using control points by Li et al. [35] with FFD and by Sun et al. [57] using a commercial code with an Arbitrary Shape Deformation technique. However, shape explorations in all of these studies were constrained to specific types of changes to the original shapes, and as a result, only subtle changes to the original shape could be used to produce an "optimal" shape. With FFD and NURBS, generating global and radical changes to an object is difficult. In addition, since the positioning of the control points is arbitrary, the outcomes of the optimization process are dependent on the individual users performing the optimization. Increasing the number of control points may help in reducing issues associated with the arbitrariness of selecting control point locations, but the increased number also increases the number of degrees of freedom of the design space, which can make the optimization computationally prohibitive. We note that there are more complex methods of deforming meshes, such as nonlinear methods that enforce an "as-rigid-as-possible mesh deformation" [55].

2.2 Spectral methods

2.2.1 Representation of shapes

The spectral representation of a shape is a way of analytically expressing a shape. It is a particularly useful way of representing a shape that is initially given by a set of grid points when the shape needs to be morphed with other shapes or when the shape needs to be modified to conform to imposed constraints, such as having a wing fit onto a fuselage—see Sect. 8.2. To explain how spectral methods can be used to represent the shape of an object, we begin by illustrating their use in representing the 1D boundary of a 2D planar object. Consider the 2D object whose boundary lies in the plane shown in Fig. 2a. Using a polar coordinate system (r, ϕ) with the origin in the interior of the object, the object's shape can be described by the radius $r(\phi)$ of its boundary. We approximate the radius $r(\phi)$ as a truncated Fourier series of the azimuthal angle ϕ:

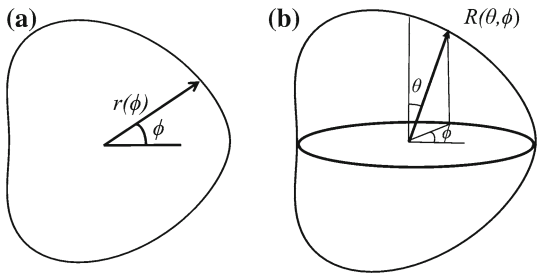


Fig. 2 The boundary of a star-shaped **a** 2D object or **b** 3D object can be represented by approximating its radius with respect to its center as a truncated sum of Fourier modes or spherical harmonic modes

$$r(\phi) \simeq \sum_{m=-M}^{M} a_m \, e^{i\,m\,\phi}, \tag{6}$$

where a_m is the mth mode Fourier coefficient. This representation has an advantage that if the shape of the object is sufficiently smooth, the series converges exponentially, which is referred to as spectral convergence. Of course, this representation is only possible if the boundary is *star-shaped*, meaning that an origin can be found within the 2D object such that $r(\phi)$ is single-valued as a function of ϕ, or equivalently, such that an origin can be found such that a straight line segment can be drawn between it and each point on the boundary such that segment does not intersect another point on the boundary.

Spectral representations of shapes can be easily extended to 3D. Consider a star-shaped object as shown in Fig. 2b. The 2D shape of this 3D object can be described by its radius $R(\theta, \phi)$ in spherical coordinates, where θ is the latitudinal angle and ϕ is longitudinal angle. The radius $R(\theta, \phi)$ is approximated as a truncated series of spherical harmonics $Y_l^m(\theta, \phi)$:

$$R(\theta, \phi) \simeq \sum_{l=0}^{L} \sum_{m=-l}^{l} a_l^m \; Y_l^m(\theta, \phi), \tag{7}$$

where l and m, are the degree and order of the spherical harmonics, L is the degree of truncation, and a_l^m is the (l, m) spectral coefficient. For a non-star-shaped object, mappings of θ and/or ϕ can be used to make the object star-shaped [8].

2.2.2 *Transforming between a spectral and a mesh representation*

Spherical harmonics have been used to approximate the shapes of brain structures [11,51,52,56]; however in these studies, a least squares method was used to compute the spectral coefficients in the spherical harmonic sum starting with brain shapes that were represented with a polygonal mesh. It is well-known that the use of the least squares method to compute spherical harmonic (or Fourier) coefficients is ill-conditioned [23], and its use leads to poor convergence (not exponential) of the truncated spectral sum, so that the errors associated with the truncated sum are large. Fortunately, there are well-established methods [7,9,23,38,50] for efficiently obtaining the coefficients of Fourier and spherical harmonic series from polygonal meshes that produce exponentially-convergent truncations. These numerically fast methods are equivalent to unitary linear transforms and inverse transforms. The inverse transforms are used to compute the spectral representation of a shape when the user is given the shape's representation as a polygonal mesh, and the transforms are used to compute the shape's representation with a polygonal mesh when the user is given its spectral representation. In particular, note that the spherical harmonic $Y_l^m(\theta, \phi)$ is the normalized product of the Fourier mode $e^{im\phi}$ and the associated Legendre function $P_l^m(cos\theta)$. For example given the shape of a star-shaped object as a set of mesh points, to find an analytic representation of the shape as a truncated series of spherical harmonics of degree L, as in Eq. 7, the following steps should be carried out: (1) choose an origin of a spherical coordinate system inside the object (as in Fig. 2b) such that the shape is star-shaped; (2) by interpolation on the mesh, create new points that represent the shape (i.e., the radius of the shape's surface with respect to the chosen origin) at the collocation points (θ_j, ϕ_k), for $j = 0 \ldots L$, where the θ_j are the inverse cosines of the Gaussian quadrature points over the closed interval $[-1, 1]$ [48], and $\phi_k \equiv 2k\pi/(2L + 1)$, for $k = 0 \ldots 2L$. (3) Then, following Canuto et al. [9], create the spectral coefficients a_l^m in Eq. 7 with an inverse transform by first doing a fast inverse Fourier transform in ϕ, followed by the inverse transform in θ using a Gaussian quadrature carried out as a matrix multiply. Once a shape is represented spectrally, it can be computed at the collocation points on the mesh by doing the sum over l in Eq. 7 (which is a matrix multiply) and then effectively doing the sum over m by doing a fast (forward) Fourier transform in ϕ. The order of doing these two sums (or transforms or matrix multiplies) is not important either in the forward (from spectral sum to evaluation on the mesh) or inverse (from mesh to spectral coefficients) transforms.

3 Methods of optimal design

3.1 Non-systematic or trial and error methods

When the design of shape of an object under-performs, engineers often rely on heuristic ways of improvement. For example, they usually rely on their intuition based on previous experience to decide whether a surface of a leading-and-trailing car of a train should be made more concave or convex, whether another piece should be lengthened or shortened, or whether a part should be more circular or more elliptical, etc. More systematic ways of creating a new design, such as FFD and "push-pull" tools used with NURBS, are often time-consuming. Moreover, because these methods represent the shapes locally (e.g., with splines), they cannot create radical new designs. Even modest design changes created with these methods have shapes that are often problematic when coupled to other computer packages such as mesh generators. In particular, surfaces generated with push-pull packages are often not "watertight". That is, the mesh points along the newly designed surface are interpreted by some computer packages to be multiple, disconnected surfaces rather than a single continuously connected surface.

3.2 Gradient-based and gradient-free methods

Gradient-based optimization methods, which use derivative information to search for minimum values of the cost function, are efficient for high-dimensional problems. These methods compute the cost function and "march downhill" until a local minimum is achieved. One gradient-based method that is especially sophisticated is the adjoint method, pioneered by Pironneau [45] and Jameson [28]. The adjoint method is used in tandem with control point-based shape deformation methods. In aerodynamics, the adjoint method formulates the optimization problem by computing the dual of the cost function and then uses Lagrangian multipliers to minimize it by computing the sensitivities of the aerodynamic variables with respect to the adjoint variables [29]. Recently, adjoint methods have been used to optimize the nose shapes of high speed trains. Munoz-Paniagua et al. [41] performed single-objective optimization to minimize the drag of the train, reducing it by 7.2%. Jakubek and Wagner [27] achieved a 20% reduction in the pressure wave generated by a high speed train, but their attempt to reduce its drag had only a 0.005% improvement.

Gradient-free methods that minimize the cost function use heuristic global optimizers such as genetic algorithms [40] and particle swarm algorithms [63]. However, these global methods can be prohibitively computationally expensive when the number of design parameters is large. Therefore,

these gradient-free methods are usually used when the representation of a design can be done with only a few design parameters, such as CAD shapes [15,16].

4 Design by morphing approach

4.1 Philosophy and dimensionality reduction

Our *design-by-morphing* methodology was developed for cases in which a global, or even radical, design change is desired and when the cost function of the design is computationally expensive to compute. The latter is the case for most designs in which the aerodynamic or hydrodynamic behavior of the shape needs to be calculated as part of the procedure of evaluating the cost function. In addition, it is also often the case when the cost function depends on thermodynamic or structural properties of the design. The philosophy governing *design-by-morphing* is that the number of shape parameters and the number of parameters used for their deformation should be severely reduced (fewer than 20–100 variables) so that the design space is small and optimal solutions can be found with global, rather than local, methods with relatively few CFD or other types of numerically expensive calculations. Therefore, design-by-morphing is, in some sense, the antithesis of finding a design with an adjoint method, which explores a high-dimensional design space, but settles for a local-attractor-prone, gradient-based optimizer.

Because design-by-morphing only uses a few design variables, it is necessary that the variables be chosen appropriately. Rather than using a small set of control points or a few CAD parameters to reduce the number of design parameters, design-by-morphing uses the shapes of existing or proposed objects (or sub-objects from which an object is made). We call these the *baseline shapes*. New designs are then created by morphing the baseline shapes with different weights for each baseline shape. Generally, the morphed object must be constrained (so that its length, width and/or volume are fixed, so that the angles of attachments of sub-objects are fixed, or so that the location, slopes (i.e., the tangent planes), and/or curvatures of its back or front are fixed, etc.). The method by which constraints are imposed can make the number of degrees of freedom of the design smaller or larger than the number of baseline objects.

The reason that we use existing shapes of an object rather than CAD variables or shapes is that we believe that there is a large amount of useful information contained in an existing shape because the existing object has already been empirically tested and likely has some useful features. By combining shapes of the baseline objects, we assume that the new design will contain some of the good features of the baseline objects or that even better features can be created from them. In our method, combining the shapes of the baseline objects is called *morphing*, and morphing is implemented by determining the weights of baseline shapes used in the morph. The shape of a new object can be optimized by finding the optimal morphing weights. In the optimization process either an associated cost function is minimized, or a performance function is maximized, with respect to the weights. In design-by-morphing, since each degree of freedom is associated with the overall shape of a baseline design, shape changes induced by a variation in morphing weights are global. The values of weights can be negative as well as positive. If all weights are positive, it is called *interpolation morphing*. Otherwise, the morphing is called *extrapolation morphing*.

Perhaps the method most similar to our design-by-morphing method was carried out by Kang [31] who attempted to create a new shape of ships from different ships. Kang et al. developed new ship designs by combining baseline ships based on triangular mesh morphing. Although Kang et al. demonstrated a capability to morph multiple objects, they did not impose any geometric constraints on their morphed designs nor report on any improved performance of the morphed shapes.

4.2 Reconstruction of response surfaces with an artificial neural network (ANN)

The direct calculation of the cost function using CFD or other large computer codes is generally the computationally most expensive part of design-by-morphing. As discussed above, we, like other designers, try to minimize the number of direct calculations of the cost function by using response surface methodology, which is a collection of mathematical and statistical techniques for modeling the functional relationship between a dependent variable, such as the cost function, and the independent variables on which it depends [32,33]. With design-by-morphing, the weights of the baseline shapes used in the morph are the independent variables. Response surface methodology is widely used in the optimization of engineering systems [6]. It is particularly useful when the functional relationship between the independent and dependent variables is not analytically known or is largely unknown empirically and must be evaluated many times and at great computational or laboratory cost.

To construct our response surfaces, design-by-morphing uses artificial neural networks (ANN), which recently rose to prominence in a number of fields for its enormous capacity in deep learning and artificial intelligence. ANNs have been used extensively in a number of fields as universal function approximators [25]. A feed-forward, multi-layered neural network has been shown to be capable of approximating a continuous function arbitrarily well [13]. The use of an ANN for response surface methodology has been discussed and demonstrated by Anjum et al. [3].

Here, we illustrate the effectiveness of an ANN for the optimization of a train using 3 or 2 train baseline shapes that are constrained such that the design space has 2 or 1 degrees of freedom. We use two different ANNs. One is the proprietary ANN from NUMECA [62] and the other is our own ANN that has 2 hidden layers. The backbone code for our ANN is powered by the open-source library TensorFlow [1]. The latter ANN is used to validate and extend the findings of NUMECA's ANN. The *training cost function J* of the ANN,[1] which is used to train the ANN to simulate the response of a full CFD calculation of the train's cost function (which here is the drag of the train) is defined by a mean-squared-error with a regularization penalty λ:

$$J(\mathbf{w}) = \sum_{i=1}^{n} (C_{d,i} - \overline{C}_{d,i})^2 + \lambda ||\mathbf{w}||_2^2, \tag{8}$$

where $\{C_{d,i}\}$ are the values of the drag coefficient of the ith test design (defined by its own unique set of weights of the baseline shapes) computed with CFD, $\{\overline{C}_{d,i}\}$ are the values of the drag coefficient predicted by the ANN, $\mathbf{w}$ is a vector of neural network connectivity weights, λ is the penalty function for the connectivity weights, and n is the number of design points in the training set. Our reconstructed response surface from the ANN used for the 3-baseline trains with 2 degrees of freedom is presented in Sect. 6.7.

5 Morphing

Obviously an important part of design-by-morphing is the morphing algorithm that creates the test object from the baseline shapes. Here, we briefly review current morphing methods, and in more detail we describe the spectral morphing algorithm that we used here.

5.1 Current methods for surface mesh based morphing

The key to shape morphing is the issue of surface point correspondence. In most applications in computational geometry, point correspondence is achieved by using shape parameterization. Shape parameterization finds point-to-point correspondence for multiple input shapes as long as the input shapes are parameterized to the same domain. Since shape morphing is essentially the morphing of multiple surface mesh patches, mesh parameterization techniques embed the surface of three-dimensional objects in a common two-dimensional domain [17]. Thereafter, point correspondence can be easily achieved using a point-in-element search, and nodal values can be acquired using barycentric interpolation [39]. Depending on the geometric properties of the base shape for morphing, different parameterization techniques are used. For meshes with open boundaries, as well as complex geometries, the mesh usually undergoes segmentation and each piece undergoes planar parameterization [47]. For a genus-zero closed mesh (i.e., no open boundaries), the natural parameterization is to a sphere due to its topological equivalence, and a number of morphing techniques using spherical parameterization have been proposed [21,39]. However, the coding process for such a routine can be tedious. For simple star-shaped genus-zero objects, spectral representations with spherical harmonics are both easier and more accurate than mesh parameterizations.

5.2 Spectral and grid based morphing

In the design-by-morphing method that we use in this paper, morphing is implemented in two ways: (1) spectral morphing and (2) grid-based morphing. Baseline objects are linearly morphed by combining them with their corresponding weights in both cases. The difference between the two methods is the former starts by computing the set of spectral coefficients that are used in each of the truncated spectral sums that are used to represent the baseline shapes, while the latter directly uses the locations of each baseline shape's mesh points. Using the spectral morphing methods outlined in Sect. 2.2.1, a new morphed shape with radius $R(\theta, \phi)$ is created from N baseline shapes as:

$$R(\theta, \phi) \equiv \sum_{l=0}^{L} \sum_{m=-l}^{l} A_l^m Y_l^m(\theta, \phi), \tag{9}$$

where the spectral coefficient A_l^m with degree l and order m is

$$A_l^m \equiv \sum_{k=0}^{N-1} \omega_k \, a_l^m(k), \tag{10}$$

where $\{a_l^m(k)\}$ are the spectral coefficients of the kth baseline shape, and ω_k is its weight. The origins used in the spectral representation of each of the baseline shapes must all be the same point.

With the spectral representation in Eqs. 9 and 10, there is a 1–1 mapping between the surface of the object and the surface of a sphere. If we limit the domain to $0 \le \theta \le \pi/2$ and $-\pi/2 \le \phi \le \pi/2$, there is a 1–1 mapping between the shape of a surface and a quarter-sphere. In this paper, not only are all of the train noses star-shaped, but also the back and bottom of the nose are planes that form a right angle. Furthermore, the train is right–left symmetric with respect

[1] The *training cost function* of the ANN and the cost function of the optimized train, which in this case is the drag of the train, are two distinct functions and should not be confused with each other.

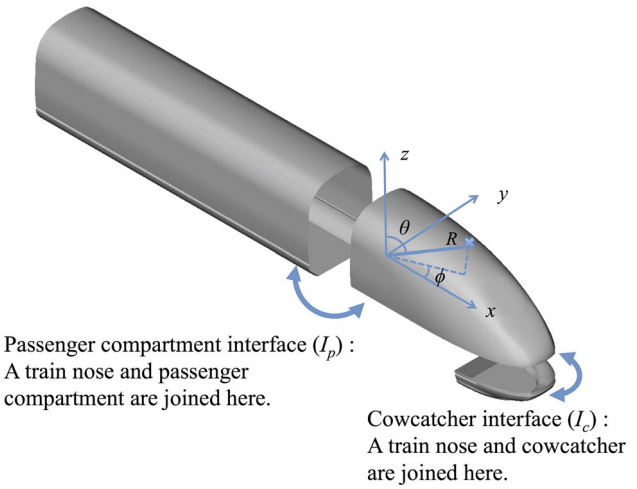


Fig. 3 Geometric constraints should be imposed on the passenger compartment and cow-catcher interfaces with the nose. The spherical coordinate system we use is shown with its origin at the back and bottom of the nose. The train velocity is along the positive x-axis so the leading point of the nose is at $\phi = 0$ and $\theta \simeq \pi/2$

to the centerline of the railroad track, so that we represent the left half of the nose over the domain $0 \le \theta \le \pi/2$ and $0 \le \phi \le \pi/2$ (with the right half of the nose having $0 \le \theta \le \pi/2$ and $-\pi/2 \le \phi \le 0$). The shape is an even function of ϕ (see Fig. 3). The back of the nose (where it joins the passenger compartment) has $\phi = \pm\pi/2$, and the bottom of the nose (where it joins the cow-catcher and bogie) has $\theta = \pi/2$. The symmetry plane is at $\phi = 0$. The front of the nose is at $\phi = 0$ and $\theta \simeq \pi/2$.

Grid-based morphing among the grids of N baseline shapes, each with P mesh points, is implemented by combining grids of baseline shapes after one finds (possibly by interpolation) a one-to-one correspondence for each of the P mesh points for all of the N baseline shapes. (That is, if we were morphing N human bodies, and the ith mesh point of the first baseline shape was the tip of a nose, then the ith mesh point of all of the other baseline shapes must be a tip of a nose and not, say, the big toe). The locations of the P grid points on the surface of the kth baseline shape are defined as $\mathbf{X^i}_k \equiv (x_k^i, y_k^i, z_k^i)$, where x_k^i, y_k^i and z_k^i are respectively the x, y and z positions of the ith mesh point of the kth baseline shape. A morphed object with shape $\mathbf{X}^i_{morphed}$ is obtained by

$$\mathbf{X}^i_{morphed} = \sum_{k=0}^{N-1} \omega_k \, \mathbf{X}^i_k, \tag{11}$$

for all $i = 1 \ldots P$. If all of the baseline shapes are star-shaped, and if we use spherical coordinates such that (x_k^i, y_k^i, z_k^i) and $(r_k^i, \theta_k^i, \phi_k^i)$ correspond to the same physical location, and if (θ_k^i, ϕ_k^i) are the same as the collocation point angles (as defined in Sect. 2.2.2), the spectral and grid-based morphing are the same up to round-off error.[2]

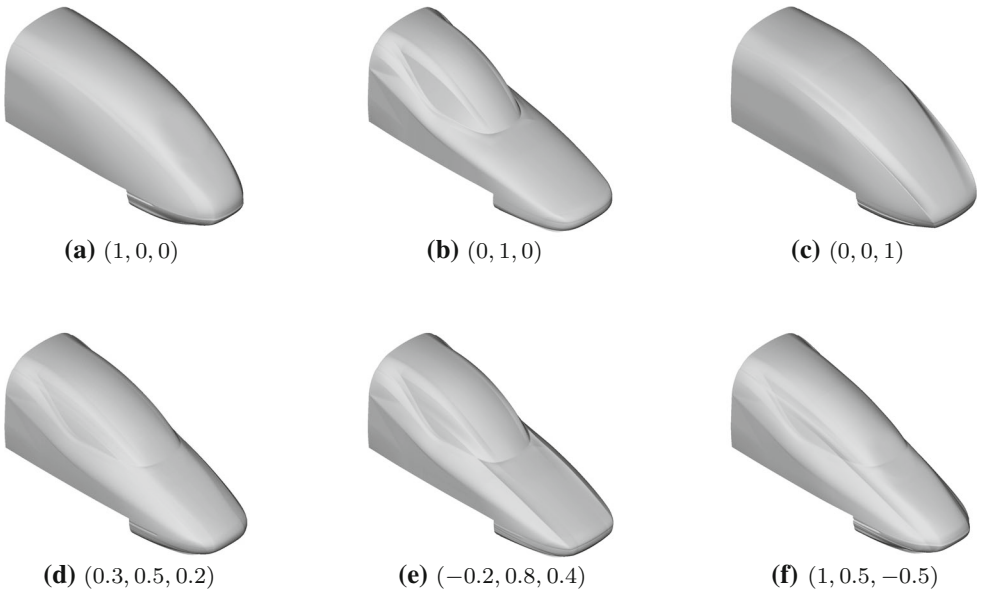
As discussed in the next section, the nose and cow-catcher of the morphed leading-and-trailing car were constrained so that there were no discontinuities in their surface shape locations where they join. The cow-catchers were either represented using spherical harmonics, mapped to one quarter of a sphere like the nose and morphed spectrally, or they were represented with a polygonal mesh and changed with grid-morphing. Both representations and their morphings worked well, but spherical harmonics have two advantages. First, due to the exponential convergence of the spectrally truncated sums in Eq. 9, it takes far fewer spectral coefficients than polygonal mesh points to represent a (sufficiently-smooth) shape. Second, with spectral representation, geometric features of a shape, such as its tangent planes and curvatures have analytic representations that are very accurate and are infinitely differentiable. These features can be very useful when computing cost functions and other properties of the morphed design.

5.3 Spectral and grid representation constrained by weights

In practical designs, geometric constraints are often required. Here, we require that the back of a train's nose seamlessly connected to the passenger compartment so that the locations and slopes of the surfaces have no discontinuities. Also, when a train's nose and cow-catcher are joined, the shape of the train's nose at the joint should be the same as that of the cow-catcher. In this paper, we use the simplest method to impose these constraints. We defer discussions of more sophisticated ways of imposing constraints to our Discussion section. Consider the interface where a train's nose is connected to the passenger compartment, as shown in Fig. 3. To enforce the constraint, we restrict ourselves to baseline train noses that all join onto the *same* shape passenger compartment and such that the locations and slopes at the interface of all of the baseline noses with their passenger compartments are continuous. With this set of baseline train noses, it is straight-forward to show (see the Appendix) that the morphed leading-and-trailing car will have a shape such that the location of the surface and slopes of the surface are continuous at the interface as long as the sum of all of the morphing weights of the baseline noses is equal to unity.

In this study, the locations of the surface of the nose and the surface of the cow-catcher are required to be continuous

[2] If shapes are not represented on grids in this manner, then the morphed shapes based on their grid representations will not only differ from the morphed shapes based on their spectral representations, but also will often produce unexpected shapes that look very different from the objects from which they were morphed.



(a) $(1, 0, 0)$ **(b)** $(0, 1, 0)$ **(c)** $(0, 0, 1)$

(d) $(0.3, 0.5, 0.2)$ **(e)** $(-0.2, 0.8, 0.4)$ **(f)** $(1, 0.5, -0.5)$

Fig. 4 Morphing of train heads. Top row: the three baseline train noses with cow-catchers. From left to right, are baselines **0**, **1** and **2**. Bottom row: three examples of morphed train heads. The three morphing weights are shown in parenthesis as $(\omega_0, \omega_1, \omega_2)$. The weights sum to unity which makes the locations and slopes of the tangent planes at I_p continuous. The morphing weights of the nose and cow-catcher are always the same so that there are no discontinuities in the locations of the surface at I_c. **a** (1, 0, 0). **b** (0, 1, 0). **c** (0, 0, 1). **d** (0.3, 0.5, 0.2). **e** (− 0.2, 0.8, 0.4). **f** (1,0.5, − 0.5)

at their interface (I_c in Fig. 3). However, the slopes of the tangent planes at interface I_c are not required to be continuous. (In fact, the baseline leading-and-trailing car that we labeled baseline **2** has discontinuous slopes at I_c.) It can be easily shown that if the locations of all of the baseline leading-and-trailing cars are continuous at I_c *and* if the noses are morphed with the same weights as their corresponding cow-catchers, the locations of the surface of a morphed train at I_c will also be continuous, even if the morphing weights do not sum to unity. In this study, we use the same same weights for the morphed noses and the morphed cow-catchers. The reason that we did not represent the nose and cow-catcher, or head, as a single object, was that the heads were neither star-shaped nor sufficiently smooth for spectral representations with spherical harmonics.

5.4 Examples of constrained morphing

Figure 4 shows three baseline and three morphed train heads produced with constraints. The train baseline heads are in the upper row. Those heads, like those in the morphed trains in the lower row, have no discontinuities in location at I_c because the morphing weights are the same for the noses and cow-catchers. Baseline **2** in panel (c) has discontinuities in the slopes of its tangent planes at I_c causing all of the morphed trains to have discontinuous slopes there. There are no discontinuities between the nose's location or its slope at the interface I_p with the passenger compartment (not shown) because the sum of the morphing weights equals unity. The noses were morphed with spectral morphing. The cow-catchers were morphed with both spectral morphing and grid morphing, but we found no discernible differences.

More examples of morphed train heads (and a view from a different perspective) are shown in Fig. 5. Here, the morph is between only baseline **0** and **1**, shown in the the top row. The morph in panel (c) has weights $(\omega_0, \omega_1, \omega_2) = (0.5, 0.5, 0)$ and represents an interpolation. The morphs in panels (d) and (e) are extrapolations with morphing weights $(1.5, -0.5, 0)$ and $(-0.5, 1.5, 0)$, respectively. Extrapolation morphing is important (and can be applied to more than two baseline objects) because negative weights allow the new design to de-emphasize characteristics of under-performing baseline objects, and weights greater than unity allow the enhancement of well-performing characteristics of baseline objects. As mentioned in our Discussion section, extrapolation can be particularly useful when different morphing weights are applied to different sub-objects (such as the nose, the cow-catcher, and the passenger compartment) of an object. Radical design changes can be made with extrapolation.

In this paper, we represent all of the train noses as truncated sums of spherical harmonic functions, although the shape of each nose was originally provided to us as a set of points on a polygonal mesh. To accurately represent a nose, we set the degree of truncation, L in Eq. 7, to 512. With this degree of truncation, all of the features of a train nose are well-resolved (See Fig. 6). The accuracy of our truncated expansion of spherical harmonic functions can be defined quantitatively as

$$E_r = \frac{||R_{mesh}(\theta, \phi) - R(\theta, \phi)||_2}{||R_{mesh}(\theta, \phi)||_2} \tag{12}$$

where $R_{mesh}(\theta, \phi)$ is the radius of the train nose on the mesh, and $R(\theta, \phi)$ is the radius computed with the truncated

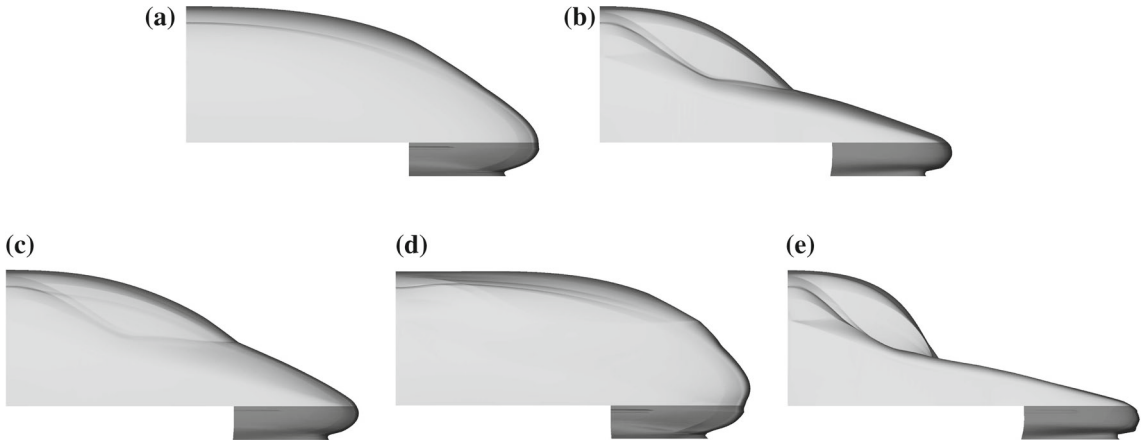


Fig. 5 Morphing of train heads. The train heads in **a** and **b** are baselines **0** and **1**, respectively. The morph in **c** is an interpolation with weights $(0.5, 0.5, 0)$, and the extrapolations in **d** and **e** have weights $(1.5, -0.5, 0)$ and $(-0.5, 1.5, 0)$, respectively. Here, train noses are colored as light gray, while train cow-catchers are dark gray

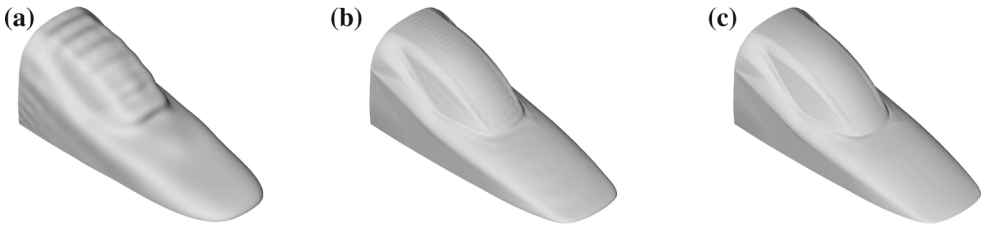


Fig. 6 Effect of the degree L of truncation of a truncated series of spherical harmonics on the shape of the baseline **1** train nose. The shapes of the train noses in panels **a**–**c** are computed with $L = 32$, 128 and 512, respectively. Wiggles at the windshield of the train nose occur when the train nose is approximated with the low L in panels (**a**) and (**b**). These wiggles are not present in panel (**c**), which is virtually indistinguishable from the original shape that was provided to us as a set of mesh points

spectral sum. The value of E_r with $L = 512$ for baseline train noses **0**, **1** and **2** are 2.815×10^{-7}, 3.903×10^{-7} and 2.024×10^{-7}, respectively.

6 Steps in *design-by-morphing*

6.1 Definition of the train and cost function to be optimized

In this paper, we consider a train in which the leading car is coupled directly to an identical trailing car as in Fig. 7. We consider this idealized train due the observation by Cooper [12], who showed that flow structure downstream of the leading the car of a high-speed train and upstream of the trailing edge of the trailing car does not significantly vary after approximately one car length. We carry out single-objective optimization by minimizing the total drag (our cost function) on the train at one speed. Minimization of the rocking due to vortex shedding, especially in a cross-wind, and the reduction of the micro-pressure wave when a high-speed train travels in a long tunnel could also have been objectives we could have easily considered. However, we decided in this study of

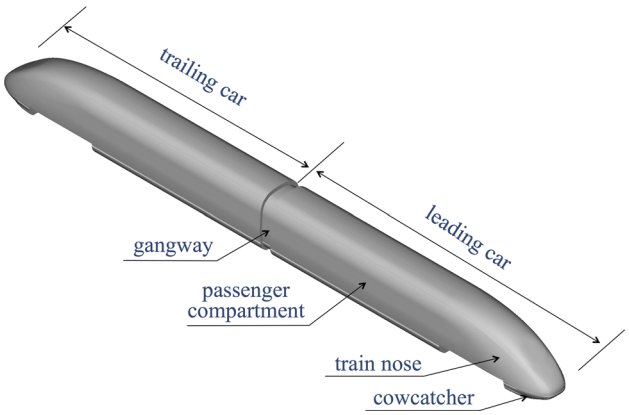


Fig. 7 A simulated high speed train. The train consists of a leading car and trailing car. The leading and trailing cars have the same shape

design-by-morphing to only consider the drag. It should be noted that aerodynamic drag is approximately proportional to the square of a train's speed [44], so energy dissipation caused by the aerodynamic drag, which scales as the cube of the train's speed becomes an overarching issue as train speeds increase. When optimizing a train here, we only consider changes to the shape of the head, in part due to the

work of Ku et al. [34] who found that the drag was much more dependent on the shape of the nose than it was on the passenger compartment. Our own studies showed that the effects of the bogies have little effect on the optimal shape, so here we exclude the bogies, not only from the optimization, but also from the computation of the drag with the CFD code.

6.2 Defining the baseline shapes

The baseline shapes shown in the first row of Fig. 4 were based on existing leading-and-trailing cars. Baseline **0** was based on publicly available train shapes that mimicked the CRH1 train. Baseline **1**, with its accentuated slenderness and concavity was based on the Talgo 350 and Talgo AVRIL leading-and-trailing cars. Baseline **2** was based on publicly available files of an existing train and was chosen because it has a large drag compared with the other baseline leading-and-trailing cars. We wanted to see if the optimal design found by our algorithm would give baseline **2** a morphing weight that was small or negative. The sum of the morphing weights is required to be unity.

6.3 Design steps

We first searched for an optimal solution using only baseline heads **0** and **1**. Because the two morphing weights were constrained to add to unity, there was only one degree of freedom in that design space. We also searched for optimal solutions in a design space with two degrees of freedom by using all three baseline heads constrained such that the sum of the three morphing weights added to unity. Our optimization can be roughly divided into the following steps.

1. *Design Space* Choose the ranges that the independent morphing weights can have in the search for the optimal design. However, if it becomes apparent (say, by finding that the optimal solution is at or near the edge of the design space), that the optimal solution is outside of the initial choice of search space, the ranges should be extended.
2. *Design of Experiment* Choose the points in the design space where the drag is to be initially calculated with CFD so that an ANN can be trained to compute a reconstructed response surface in which the independent variables are the independent morphing weight(s) of the baseline shapes and the dependent variable is the drag.
3. *Drag computed with CFD* Compute the drag by first defining a computational mesh around the train; then compute the flow (which in this case was forced to be a steady state); and then compute the drag on the train.
4. *ANN* Train and validate the ANN. Initially, do this with the design points found in step 2 along with their CFD-compute drags from step 3. However when iterating, use the expanded set of design points (defined in step 6 below) and their CFD-computed drags. Create the reconstructed response surface using the drag predicted by the ANN.
5. *Optimize* Use a genetic algorithm to find the optimal morphing weights using the drags computed with the ANN. Using the optimal weights, create the "optimal" design. Validate the "optimal" design selected by the genetic algorithm (that used the ANN-computed drag) by using CFD to compute the drag of the "optimal" design and determine the error E, which is the difference between the cost function computed with the ANN and with the CFD code (as defined below in Eq. 13).
6. *Iterate steps 4 and 5* Go back to step 4 and retrain NUMECA's ANN by adding the "optimal" design point most recently found in step 5 along with its CFD-computed drag to the original set of design points along with their drags and to all of the design points and drags found in previous iterates. Then, repeat the optimization in step 5. Repeat this iteration until either a maximum allowable number (set by the user) of iterations are carried out, or until there is no further decrease in the drag of the optimal design.

6.4 Design space

For the one-degree of freedom optimization based on baselines **0** and **1**, we chose a design space such that the morphing weight of baseline **0** satisfied $-1 \leq \omega_0 \leq 2$. This range allows extrapolation in both directions, i.e., so that the design space contains a region with $\omega_0 < 0$ and with $\omega_1 \equiv 1 - \omega_0 < 0$. For the two-degree of freedom optimization based on the 3-baseline trains, we chose a design space such that the morphing weights of baseline **0** and **1** satisfied $-1 \leq \omega_0 \leq 3$ and $-1 \leq \omega_1 \leq 2$. The search space of ω_0 is bigger than ω_1 because we found that the drag is very small when $\omega_0 \simeq 2$. Therefore, we increased the upper limit of ω_0 from 2 to 3. These ranges allow extrapolation in all three directions, i.e., so that the design space contains a region with $\omega_0 < 0$, with $\omega_1 < 0$, and $\omega_2 \equiv 1 - \omega_0 - \omega_1 < 0$.

6.5 Design of experiment

There are many well-studied methods for selecting the initial points in the design space to begin the reconstruction of the response surface. For example, Vanaja and Shobha Rani [59] discuss the Plackett-Burmann method, and the factorial and cubic face centered, are reviewed by Shyy et al. [54]. Here we used a discrete, level-based point distribution for our initial choice of points where initial design points are equally distributed in the design space because this method is intuitive and easy to understand. In particular for the 1-degree of freedom design space, 10 equally spaced points were chosen. For the 2-degree of freedom problem, six points

were equally distributed in each of the two independent directions ω_0 and ω_1 of the design space, so a total of thirty-six design points were initially selected for the first design of the experiment. Although the level-based point distribution works well when there are only a few design variables, the computational work of this method increases rapidly as the number of design variables increases, making the level-based point distribution impractical. When there are large numbers of design variables, more efficient methods of the design of experiment should be used such as the D-optimal method [19] or the optimal Latin hypercube method [61].

6.6 Drag computed with CFD

To compute the drag numerically, we solved the Reynolds Averaged Navier–Stokes (RANS) equation to simulate the flow around a high-speed train. A k-ω model with the wall function was used to model the turbulence. The turbulence intensity and length scale were defined as 3% and 0.3 m, respectively, and the velocity of the high-speed train was set at 360 km/hr. The train was located inside a simulation box with size $4L \times 2L \times L/4$ in length, width and height, respectively, where $L = 40$ m is the length of the simulated high-speed train. The reason we choose a relatively large width and small height of the simulation box is because the trailing vortices generated from the back of the train spread over a wide horizontal plane but do not travel very far upward. The distance between the upstream side of the simulation box and the front of the train was $1L$, leaving a distance of $2L$ between the back of the train and the downstream side of the simulation box. We found that increasing the stream-wise size of the simulation box had no significant effect on the drag nor on the qualitative features of the flow around the train. The velocity at the upstream boundary of the simulation box was uniform and at the downstream boundary of the simulation box we applied a uniform pressure boundary condition. No-slip boundary conditions were used at the train's surface and at the bottom of the simulation box. External boundary conditions, in which the flow is required to obey asymptotic conditions representing the far field, were used on the top and side surfaces of the computational box. The bottom boundary was modeled as a smooth surface without a model of a track or ties. Using NUMECA's HEXPRESS™/Hybrid, mesh generator, we created a new mesh for each of the baseline trains and for each morphed train that resolved the boundaries and the key vortical features of the wake (see Fig. 8). We used NUMECA's FINE™ /Open to simulate the flow and the drag on the train. Calculations were validated by increasing the spatial resolution until convergence was reached. In addition, some of the drag calculations were independently validated using the OpenFOAM code.

Several shapes of train heads at the ten initially selected points are presented in Fig. 9 with their corresponding weight

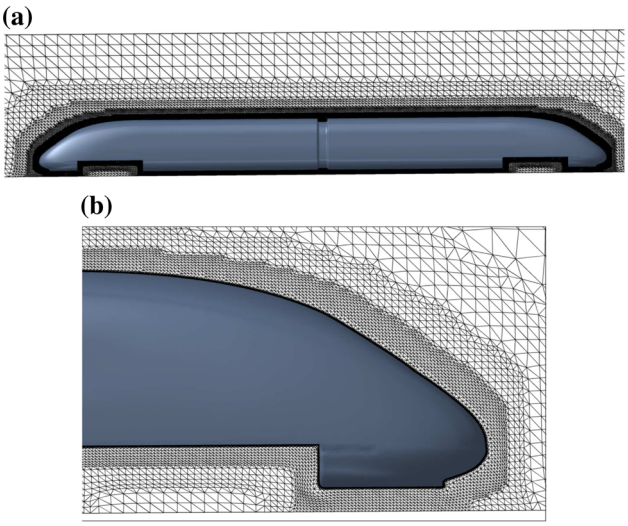


Fig. 8 Mesh around a **a** high-speed train with **b** a blow-up of the mesh at the leading edge of the train

ω_0 and computed drag coefficients C_d. Figure 10 show several shapes of morphed train heads at the 36 initially selected design points along with their corresponding weights and computed drag coefficients.

6.7 ANN

In both the 2-baseline train optimization with 1 degree of freedom and the 3-baseline train optimization with 2 degrees of freedom, we used NUMECA's ANN, i.e., the software FINE™/Design3D, to find the optimal shapes of a train. For the 1-degree of freedom optimization, NUMECA's ANN was initially trained (i.e., step 4 in Sect. 6.3) using the CFD-computed drag at 10 equally distributed points in the design space (Fig. 11). Then, we used a generic genetic algorithm along with the ANN-computed drag coefficients (i.e., step 5 in Sect. 6.3) to find the "optimal" weights of the "optimally" designed train. We then iterated (i.e., step 6 in Sect. 6.3) the training and optimization steps 10 times. The train with the least drag has $\omega_0 = 0.536$ and is an interpolation, rather than an extrapolation of the two baseline trains. For the 2-degree of freedom optimization using 3-baseline trains, NUMECA's ANN was initially trained using the CFD-computed drag at the 36 equally-distributed design points shown as black solid circles in Fig. 12. The ANN training and optimization were iterated (i.e., step 6 in Sect. 6.3) 20 times.

To train and validate the ANN, we defined the error, E:

$$E = \frac{\sum_{i=1}^{M} \left(C_{d,i} - \overline{C}_{d,i}\right)^2}{M}, \tag{13}$$

where $C_{d,i}$ and $\overline{C}_{d,i}$ are respectively ith drag coefficient from the CFD simulation and from the ANN, and M is the number

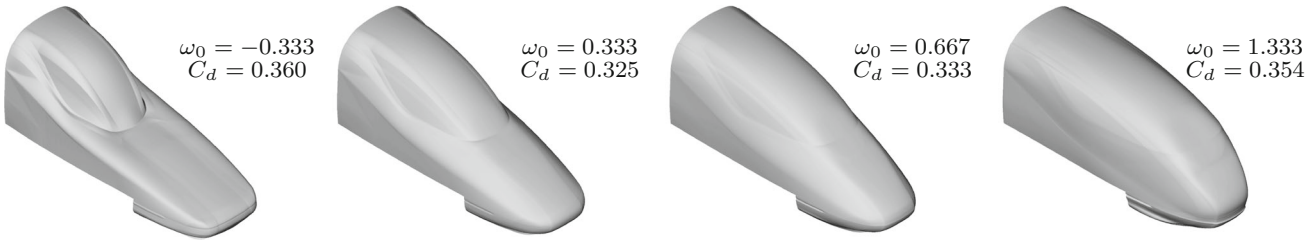


Fig. 9 Several examples of morphed train heads among the initially chosen ten design points of the 2-baseline train with 1 degree of freedom. The weight ω_0 and CFD-computed drag coefficient of each train are shown

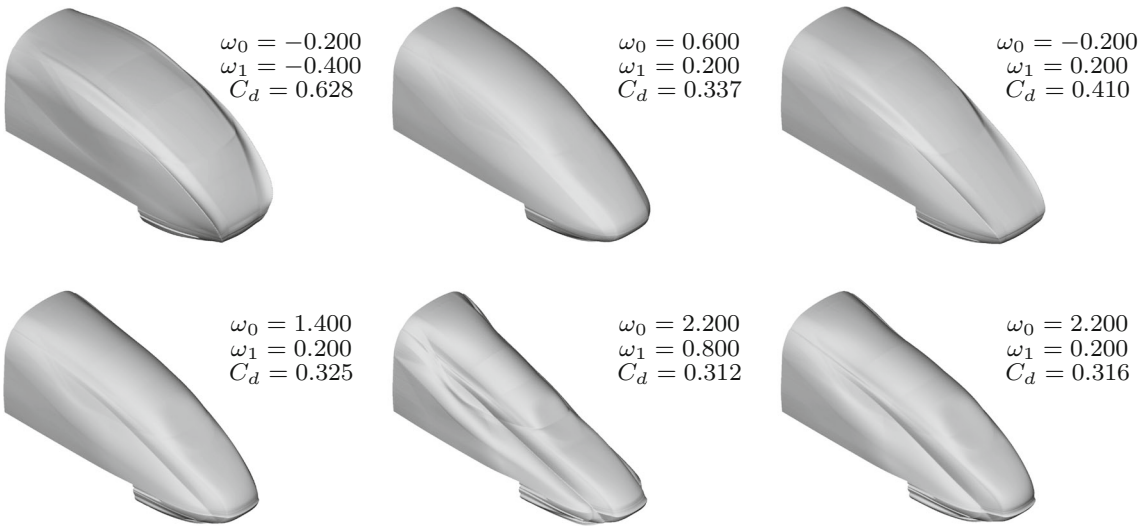


Fig. 10 Several examples of morphed train heads among the initially chosen thirty-six design points of the 3-baseline train with 2 degrees of freedom. The weights ω_0 and ω_1, and CFD-computed drag coefficient of each train are shown

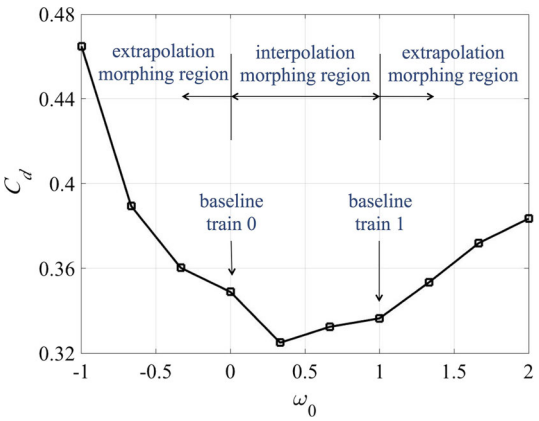


Fig. 11 CFD-computed drag coefficients of the 10 initially selected design points of the 2-baseline train with 1 degree of freedom as a function of their weights ω_0

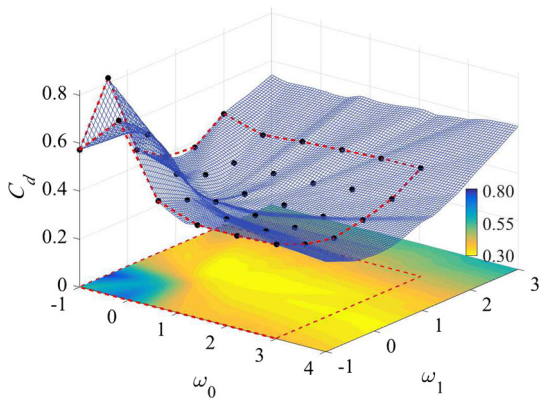


Fig. 12 The reconstructed response surface from the ANN built with TensorFlow for the 3-baseline train with 2 degrees of freedom. The uniformly distributed solid circles are the CFD-computed drag for the 36 design points from the training set. The gridded web is the reconstructed response surface found by the ANN trained with those 36 design points. The bottom part of the figure is a color-map of the ANN-computed C_d as a function of ω_0 and ω_1. The red dashed line shows the range of the design of the 36 training points

of design points. The error defined in Eq. 13 can be used to compute two types of errors: (1) training error when the error is computed with the design points used to train the ANN and (2) test error when the error is computed with design points that were *not* used to train the ANN. Unfortunately, only a training error is accessible with NUMECA's ANN because the software does not permit the user to supply independent design points and then query the ANN to compute a value for its prediction of the drag. However, the training error of

Table 1 Input parameters for our artificial neural network (ANN) built with TensorFlow that was used to compute the reconstructed response surface for 3-baseline train with 2 independent degrees of freedom

Hidden layer 1 size n_1	256
Hidden layer 2 size n_2	128
Activation function	Rectified linear unit (ReLu)
Initialization STD σ	2×10^{-2}
Regularization constant λ	10^{-4}
Base learning rate α	10^{-2}
Exponential decay Rate d	0.8
Decay steps	1000
Training steps	20,000

NUMECA's ANN with the initial 36 design points of the 3-baseline train was 3.53×10^{-5}.

To obtain an estimate for the test error of NUMECA's ANN and also to explore the possibility that the optimal train design lay outside the test range of the weight parameters we explored, we carried out several numerical experiments with our TensorFlow-based ANN for the 3-baseline train. We trained the TensorFlow-based ANN with the same 36 design points shown in Fig. 12 used to train NUMECA's ANN. The training error of the TensorFlow-based ANN was 2.22×10^{-7}. Figure 12 shows the reconstructed response surface (fine mesh) of the ANN-computed drag created with the TensorFlow-based ANN that was built with the parameters in Table 1. In this figure, the 2 degrees of freedom, ω_0 and ω_1, extend beyond the range (shown with red dashed lines) that was explored in the optimization with NUMECA's ANN. At the bottom of Fig. 12, the values of ANN-computed drag coefficients are shown with a color-map where the color indicates C_d. The range of the TensorFlow-based ANN was extended toward the direction where the influence of the baselines **0** and **1** is increased and baseline **2** decreased because it allowed us to consider the possibility that the optimal solution exists outside of our originally defined design space. However, the drag in this extended range is not small and does not decrease with distance from the red dashed lines, suggesting that the optimal train might well be within the range explored by NUMECA's ANN. We also studied the possibility that the optimal design has large negative values of ω_0 or ω_1 (not shown in the figure) because a reconstructed response surface found by our TensorFlow-based ANN suggested there was another local, or perhaps global, minimum of the drag for negative values of ω_0 and ω_1. However, when we further extended the design space to include more negative values of ω_0 and ω_1, we found that the drag increased rapidly.

Using our TensorFlow-based ANN, we had the ability to give it an arbitrary design point and query it for its prediction of the drag.[3] We therefore computed a test error for the TensorFlow-based ANN using the 20 "optimal" design points found with NUMECA'S ANN (in the iteration of step 6 in Sect. 6.3) and for which we already had the values of the CFD-computed drag coefficients. Using these 20 points, the test error of the TensorFlow-based ANN is 5.39×10^{-4}, which demonstrates the high accuracy of the TensorFlow-based ANN in the design space near the optimal designs. We presume that NUMECA's ANN has a similar value for its test error. Note that the 20 test points were *not* used to re-train the TensorFlow-based ANN, which was never re-trained after its training with the 36 design points in Fig. 12.

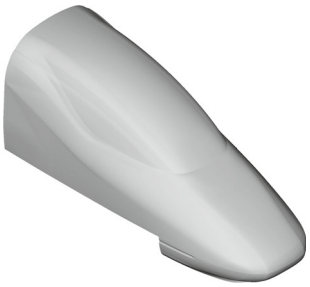


Fig. 13 The shape of the optimal train head obtained from the 2-baseline train with 1 degree of freedom optimization

7 Results

7.1 Summary of optimizations

We found the optimal shapes of a high-speed train for the 2-baseline train (with 1 degree of freedom) optimization and for the 3-baseline train (with 2 degrees of freedom).

7.1.1 2-baseline train

The optimal train head of the 2-baseline train with 1 degree of freedom optimization has $\omega_0 = 0.536$ and $\omega_1 \equiv 1 - \omega_0 = 0.464$. The optimal design is an interpolation, rather than an extrapolation and is shown in Fig. 13. It has a drag coefficient $C_d = 0.3095$, while the C_d of the baseline **0** and **1** trains is

[3] The ANN provided by NUMECA is not open-source code and is written in a way that does not give the user the ability to find the ANN-predicted drag at an arbitrary design point. (The NUMECA ANN only predicts drag values of the design points in a training set.) Determining the ANN's predicted drag at points outside the training set is important for reconstructing a response surface with a fine mesh (and not just interpolating the drag from the design points of the training set) and for predicting drag at points outside the design space as shown in Fig. 12. The latter is useful to determine whether the design space is too small; if the location of the ANN-predicted minimum drag is near the boundary of the design space or outside the design space, then the design space is too small.

Table 2 Weights and CFD-computed drag coefficients of several 3-baseline trains. The first three rows correspond to the three baseline trains. The two rows labeled "Best from DOE" correspond to the morphed train with the smallest drag that was found from the 10 design points of the 2-baseline train (i.e., the design points shown with circles in Fig. 11), or to the morphed design with the smallest drag found from the 36 design points of the 3-baseline train (i.e., the design points shown with solid circles in Fig. 12). The two rows labeled "Optimal train" correspond to the optimization of the 2- and 3-baseline trains with the lowest drags. The column labeled "Reduction of C_d" is the percentage of the CFD-computed drag reduction that the morphed train has compared to the best-performing baseline train, which is always baseline **0**

	Weights	C_d	Reduction of C_d (%)
Baseline train 0	(1.000, 0.000, 0.000)	0.3365	–
Baseline train 1	(0.000, 1.000, 0.000)	0.3489	–
Baseline train 2	(0.000, 0.000, 1.000)	0.4523	–
Best from DOE (2 baseline trains)	(0.333, 0.667, 0.000)	0.3250	3.418
Best from DOE (3 baseline trains)	(2.200, 0.800, − 2.000)	0.3124	7.162
Optimal train (2 baseline trains)	(0.536, 0.464, 0.000)	0.3095	8.024
Optimal train (3 baseline trains)	(2.023, 0.619, − 1.642)	0.2912	13.462

0.3365 and 0.3489, respectively. Thus, the optimal train has a 8.024% reduction in C_d of the best performing baseline train.

7.1.2 3-baseline train

Table 2 summarizes the results of the optimization of the 3-baseline train with 2 degrees of freedom. The table shows that the train optimized with design-by-morphing has a drag reduction of 13.462% with respect to the best performing baseline train. As expected, the morphing weight of the poorly-performing baseline **2** train, is negative, meaning that the optimal train design is an extrapolation. However, the 3-baseline train provided us with an unanticipated result. By including the poorly-performing shape of baseline **2**, the drag reduction of the optimal train with respect to the best performing baseline train is almost twice as large as the drag reduction found when baseline **2** is *not* included in the morph. We initially believed that providing our design-by-morphing algorithm baseline shapes with *good* features would allow our algorithm to choose among the best features to create an optimal shape, and our conjecture was proved by the result of the 2-baseline train with 1 degree of freedom optimization. The optimization with three baseline trains shows that choosing baseline shapes with *bad* features can also lead to large improvements by providing the algorithm with examples of features to avoid.

The optimal shape of the head is shown in Fig. 14. Compared to the shapes of the three baseline trains, the optimal design is radically changed. Inward curling edges are present at both sides of the optimal train nose. No baseline train noses have this geometric characteristics, but these features appear on the optimal design. One issue regarding this optimal shape is that its inward curling edges might be difficult to manufacture, maintain, or clean. This point is discussed below in Sect. 8.2.

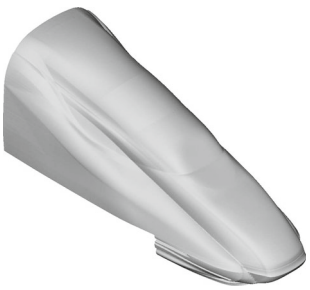


Fig. 14 The shape of the optimal train head obtained from the 3-baseline train with 2 degrees of freedom optimization

7.1.3 Cost function as a function of iteration number

Figure 15a shows the optimal weights of the 2-baseline train with 1 degree of freedom optimization as a function of the optimization iteration step (i.e., step 6 in Sect. 6.3). Figure 15b shows the CFD-computed and ANN-computed drag coefficients at each iteration step. Note that the best train shape was found in the sixth iteration, rather than in the last iteration. The ANN-computed values of the drag are smaller in iterations 8, 9, and 10 than it is in iteration 6, but the CFD-computed drags are slightly larger in iterations 8, 9, and 10 than it is in the 6th iteration. Figure 16 is similar to Fig. 15 but computed for the optimization with three baseline trains with 2 degrees of freedom. Surprisingly, the best train shape was found at the third iteration step. Both the CFD-computed and ANN-computed drags oscillated after the third iteration rather than increasing or decreasing monotonically. Defining $C_{d,i}$ and $\overline{C}_{d,i}$ respectively as the CFD-computed and ANN-computed drag coefficients at the ith iteration step, the

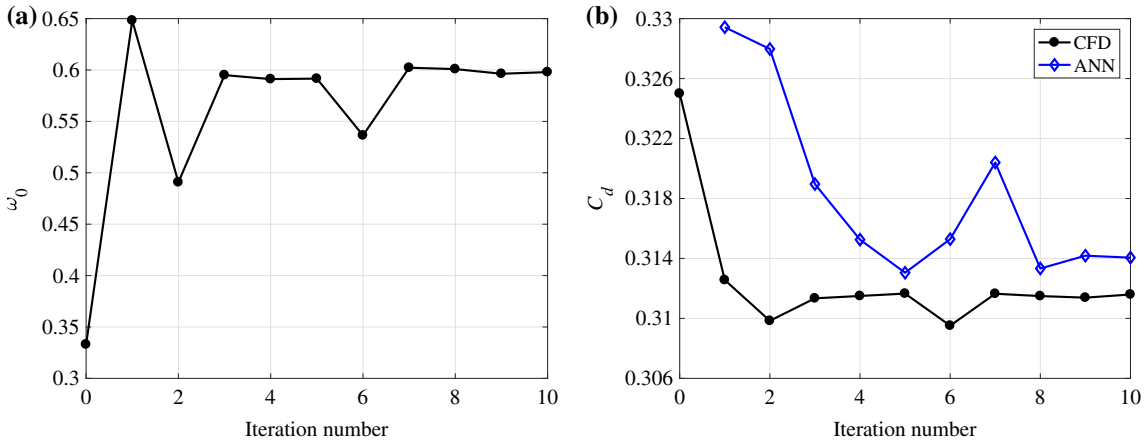


Fig. 15 The weight ω_0 in **a** and CFD-computed ANN-computed drag coefficients in **b** of the 2 baseline train optimization (with 1 degree of freedom) as functions of the optimization iteration step. The drag coefficients computed from CFD and the ANN are respectively plotted in black and blue in **b**. The minimum drag was found at the sixth iteration step at $\omega_0 = 0.536$. The values of C_d and ω_0 of the zeroth iterate are the minimum drag coefficient and corresponding weight shown in Fig. 11

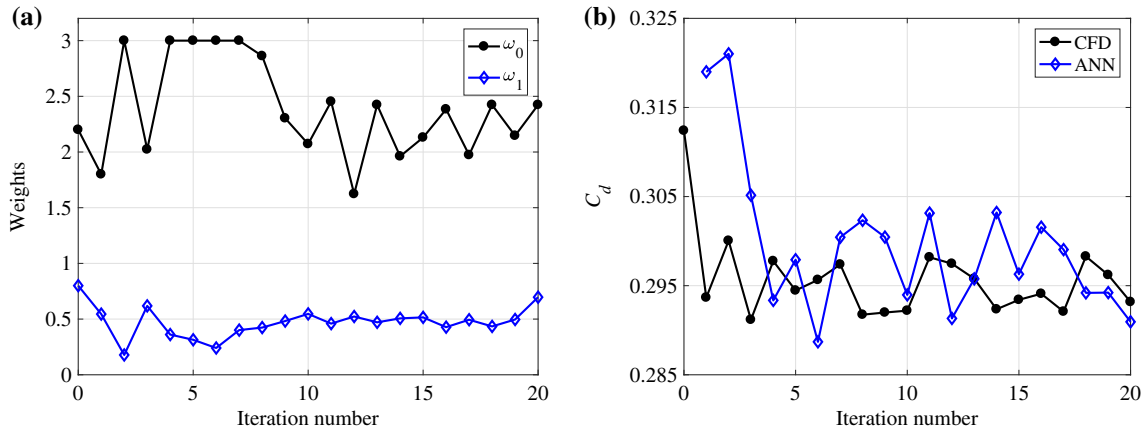


Fig. 16 Same as in Fig. 15, but for the 3 baseline train (with 2 degrees of freedom) optimization

prediction error of the ANN at the ith iteration step can be defined as

$$E_i^{predict} = \frac{|C_{d,i} - \overline{C}_{d,i}|}{C_{d,i}}. \tag{14}$$

The prediction errors of the ANNs as functions of the iteration step are presented in Fig. 17. The prediction errors tend to decrease in the first few iteration steps, and then not improve with further iteration. The reason for why the trains with the least drag occurred in Figs. 15 and 16 at the sixth and third iterations respectively is unclear. The reason could be due to the behavior of $E_i^{predict}$ or due to "lucky" choices of the genetic algorithm at the sixth and third iterations or due to some other effect. However, we suspect that the reason might be due to a flaw in our ANN. In addition to $E_i^{predict}$, there is another measure for how well an ANN computes the drag. In the study with two baseline trains, in the last (10th) iter-

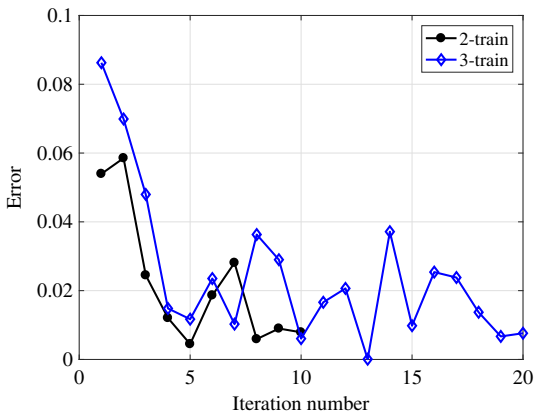


Fig. 17 Prediction error $E_i^{predict}$ of the ANN-computed drag coefficient as a function of optimization iteration number. The prediction errors of the 2-baseline train optimization (with 1 degree of freedom) and the 3-baseline train optimization (with 2 degrees of freedom) are plotted in black and blue, respectively

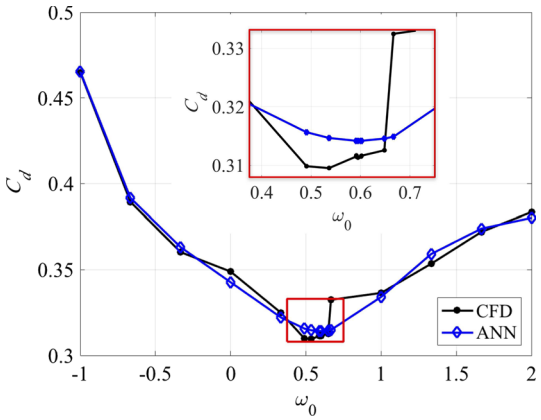


Fig. 18 ANN-computed and CFD-computed drag coefficients for the 2-baseline train (with 1 degree of freedom) as a function of the weight ω_0. The results are for the ANN trained with 19 design points, which is the ANN that was used for the last iteration in Fig. 15. The design points (black circles) are the design points of the training set (computed with CFD). The blue diamonds are the drag coefficients computed with the ANN at the same values of ω_0 of the black circles. The optimal weights found with the CFD and the ANN differ. The inset box shows that the ANN fails to capture the abrupt change in the drag near $\omega_0 = 0.65$

ation there were 19 design points in the training set of the ANN. (10 are from the originally selected design of experiment, and 9 more are from the previous 9 iterations.) Using the ANN trained with these 19 design points, Fig. 18 shows the ANN-computed drag coefficients evaluated at those 19 points along with the CFD-computed drag evaluated at the same design points. The ANN-computed drag coefficients are a smoother function of ω_0 than the CFD-computed drag coefficients are. Perhaps this is not surprising because it is known that the properties of the solutions to the equations of motion that govern fluid dynamics are not smooth functions of the control parameters. This is famously true for drag coefficients, which often undergo a "crisis" or abrupt change of value as the properties of the flow change. The drag computed by the ANN knows nothing of the properties of the governing equations of fluid motion, and perhaps the way in which the ANN is constructed or trained attempts to make the drag coefficient too smooth a function of parameters such as ω_0. A better ANN, in which an abrupt change in drag occurs near $\omega_0 = 0.65$, might allow design-by-morphing to produce designs with smaller drags as the iteration number in Fig. 15b increases, resulting in an overall improved train.

7.2 Physical properties of the optimal designs

Figure 19 shows the static pressure fields around the heads of the leading cars of the three baseline trains and the two optimal designs. At the fronts of the two optimal trains, the high-pressure regions are smaller than the corresponding high-pressure regions of the baseline **0** and **2** trains. The decrease in size of these high-pressure regions reduces the drag on the optimal trains. Furthermore, both optimal trains have smaller high pressure regions near their windshields than does baseline **1**, which also reduces their drag. Overall, the static pressures at the fronts of the optimal train heads are lower than those of the baseline trains, showing a good compromise of the aerodynamic features of baselines **0** and **1**. Figure 20 shows the static pressure fields around the trailing car's of the baseline trains and of the two optimal trains. Baseline trains **1** and **2** have low-pressure zones at their trailing edges that induce drag, while the two optimal designs do not have corresponding low-pressure regions.

Figure 21 shows stream lines in the trailing wake vortices. Baseline trains **1** and **2** have tangled stream lines beneath their cow-catchers, which induce drag, but neither optimal train has a similar tangle. The stream lines of the optimal train

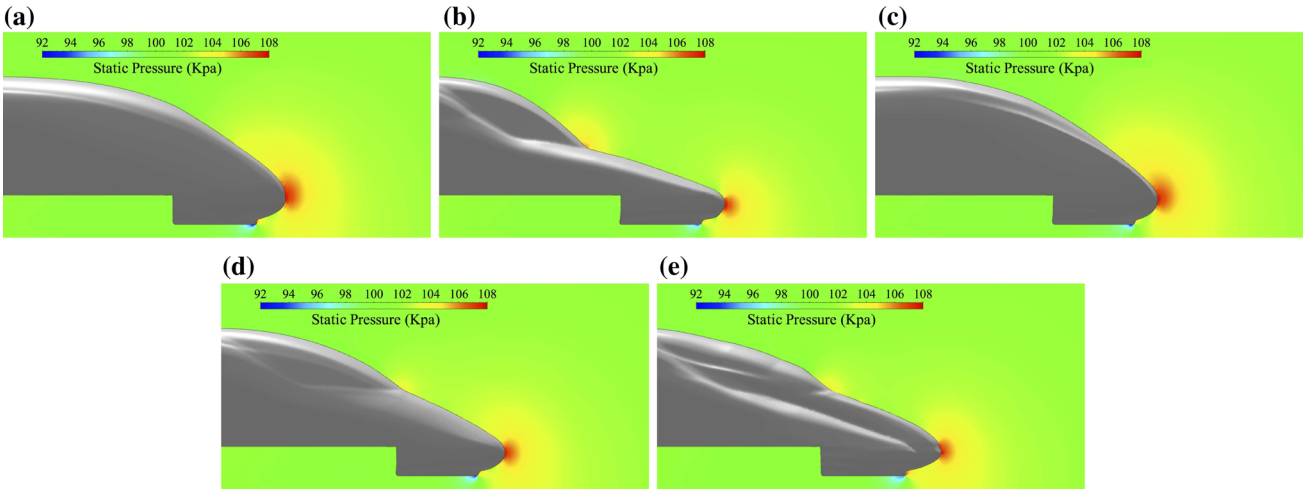


Fig. 19 The static pressure at the symmetry-plane (i.e., at $\phi = 0$, using the spherical coordinate system shown in Fig. 3 of the baseline and optimal trains' leading heads. **a** baseline train **0**; **b** baseline train **1**; **c** baseline train **2**; **d** optimal train from the 2-baseline train optimization; and **e** optimal train from the 3-baseline train optimization. The train is moving from left to right. (Color figure online)

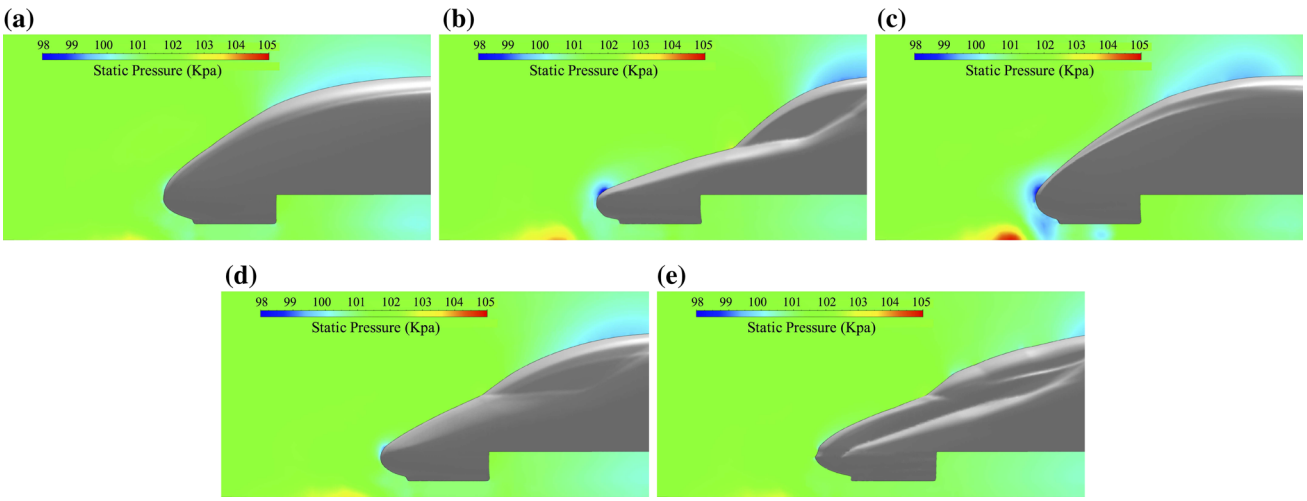


Fig. 20 Same as in Fig. 19, but for the trailing head of the train

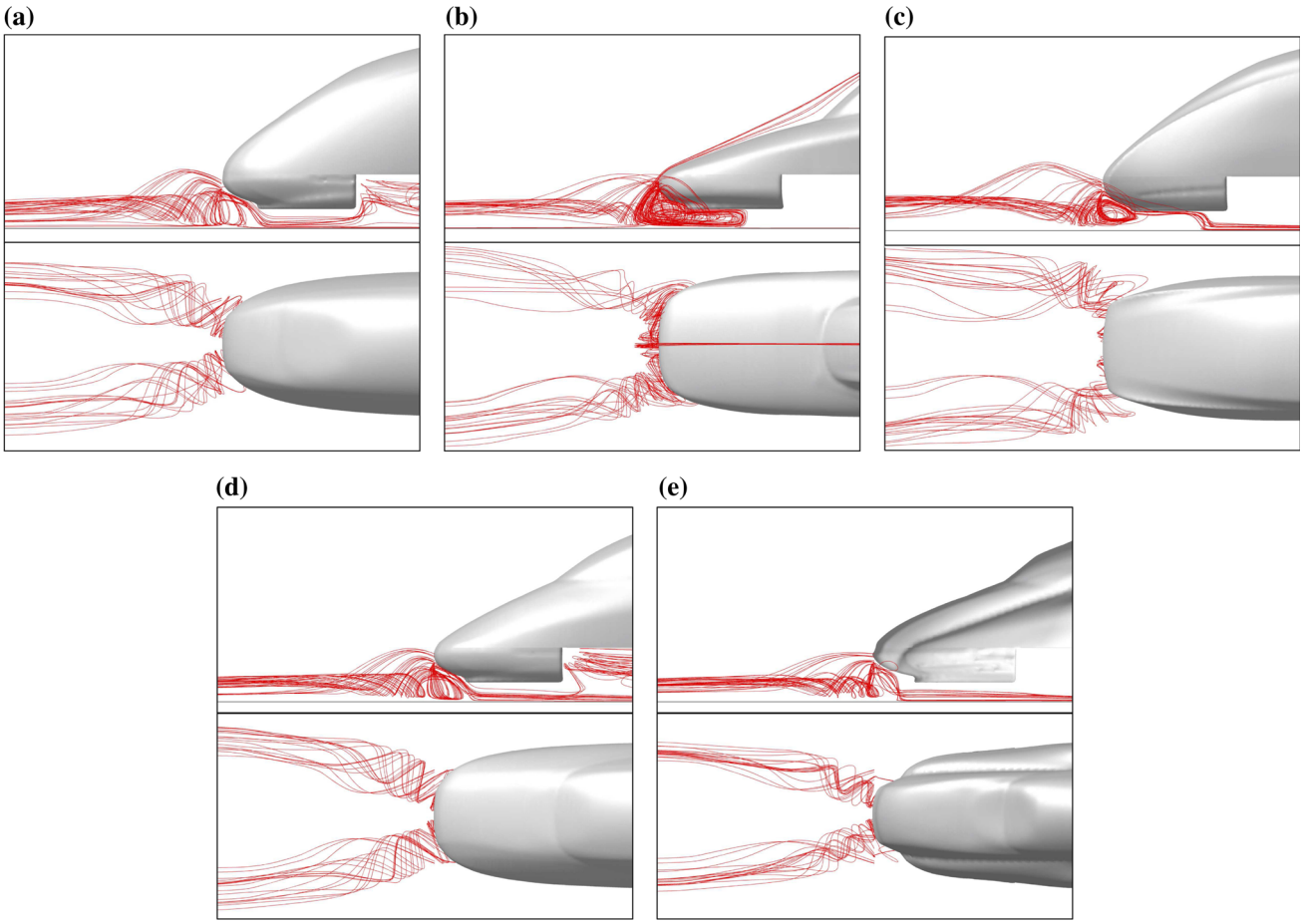


Fig. 21 Side and top views of the stream lines at the trailing cars, indicating the shed trailing wake vortices from the baseline and optimal trains. **a** baseline train **0**; **b** baseline train **1; c** baseline train **2**; **d** optimal train from the 2-baseline train optimization; and **e** optimal train from the 3-baseline train optimization

based on 3-baseline trains are exceptionally smooth beneath its cow-catcher. The vertical thicknesses of the shed vortex wakes in the optimal trains in panels (d) and (e) are noticeably thinner than in the baseline trains in panels (a) and (c); other researchers [2,20] noted that decreasing the vertical thickness of the shed vortex wake reduces drag. The optimal

train from the 3-baseline train optimization study in panel (e) has a pair of trailing vortices that are more narrowly separated from each other compared to the trailing vortices shed from baseline trains **1** and **2**. Duriez et al. [20] and Aider et al. [2] in their study of TGV trains and other bluff bodies also found that drag is decreased by modifying the trailing car shape to narrow the distance between the two shed vortices. The wake of the optimal train in panel (e) has a smaller momentum deficit than any of the baseline trains, and the smaller deficit also leads to less drag.

8 Conclusion and discussion

8.1 Summary and conclusions

We presented a new, general design method, called *design-by-morphing* for an object whose performance is determined by its shape due to hydrodynamic, aerodynamic, structural, or thermal requirements. To illustrate the method, we minimized a cost function that was set equal to the drag of a train traveling at 360 km/hr. The overarching concept of the method is to create a new shape by morphing the shapes of different objects. Here, we designed a new leading-and-trailing car of a train by morphing existing baseline leading-and-trailing cars. Our objective was to find the optimal weights of each of the baseline shapes so that the morphed shape had minimum drag. In our first attempt at optimization we used two baseline trains, and in our second attempt we use three. The morphing was done by representing the shapes of the trains in two different ways: with polygonal meshes and spectrally with a truncated series of spherical harmonics. The morphed shapes were constrained such that the head of the morphed leading-and-trailing car seamlessly joined to its passenger compartment (so that the locations of the surfaces and the slopes of its tangent planes were continuous) and to its cow-catcher (so that the locations of the surfaces were continuous at the joint, but the slopes of its tangent planes were allowed to be discontinuous there). The constraints were imposed in a way that reduced the number of degrees of freedom of the design space from 3 to 2 for the optimization using 3 baseline trains and from 2 to 1 for the optimization using 2 baseline trains. Optimal weights were found by setting a range of weights to be explored, using CFD to compute the drag of the trains in the design-of-experiment, and using an artificial neural network to create a reconstructed response surface and to allow a genetic algorithm to find the optimal weights without repeatedly using an expensive CFD code. The optimal designs found with the artificial neural network were validated with CFD. We found that with only two baseline trains that mimic current high-speed trains with low drag (so that the design space had only one degree of freedom) that the drag of the optimal train was reduced by 8.04% with respect to the baseline train with the smaller drag. The optimal shape exploited the best features of both baseline designs. When we repeated the optimization by adding a third baseline train with high drag to the morph, we were surprised by the results. Although we expected the new, optimally morphed train to have a negative morphing weight for the third baseline train, we did not anticipate that the drag would be reduced by 13.46% with respect to the baseline train with the smallest drag. This near doubling of the drag reduction meant that the optimization not only de-emphasized the poor features of the third baseline, but also actively negated them. *Bad examples of design are as useful as good examples in determining an optimal design*.

8.2 Discussion of future work

We believe that *design-by-morphing* is practical for design spaces with degrees of freedom greater than 2, but probably, due to computational limitations in training the artificial neural network, fewer than 100 if the cost function must be evaluated with CFD, another type of computationally expensive code, or with laboratory experiments. The question for future work is: how should we best use those degrees of freedom? We believe that the answer is not necessarily to use more baseline objects to include in the morphing. Rather, we believe that those degrees of freedom should be used in breaking up the baseline objects into sub-objects. It is likely that each baseline object has some features that perform well and others that perform poorly. Rather than penalize an entire baseline object for one poorly performing part, it is probably better to just penalize that part. We have already begun to explore this concept in a PhD thesis [43] and plan to expand upon it. Part of the thesis dealt with optimizing the lift-to-drag ratio of a plane by using a morph with two baseline planes. In this work, one of the baselines was an idealized version of an air force SR-71 and the other was a fictional plane inspired by a "Naboo starfighter" from Star Wars. Each plane was broken into 15 sub-objects (Fig. 22), so, in principle, there could be 15 independent morphing weights for each baseline plane, making a total of 30 weights and 30 degrees of freedom.

A problem encountered when breaking a single object into many sub-objects is that although the baseline sub-objects fit seamlessly together in the baseline object, the independently morphed sub-objects, in general, have discontinuities where they are re-joined in the morphed object. To solve this problem, in the thesis work, all of the sub-objects were represented spectrally. The surfaces of the objects were then made to obey a temporal diffusion equation and were integrated forward in time with one step. For a diffusion equation to be well-posed, boundary conditions must be imposed. We imposed boundary conditions such that the morphed sub-objects seamlessly fit together. This capability of design-by-morphing allows us

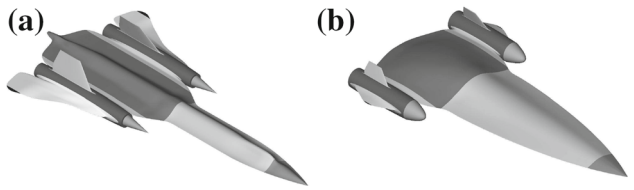


Fig. 22 A single aircraft is broken into 15 independent sub-objects. Each sub-object can be given its own morphing weight, so that a new morphed plane made from the sub-objects of 2 planes has 30 independent weights or degrees of freedom. Each sub-object is shown with a different color. **a** An idealized air force SR-71; **b** a fictional "Naboo starfighter" inspired from *Star Wars*

to break the baseline objects (such as planes) into sub-objects (such as wings, fuselages, tails, etc.) so that the algorithm can incorporate the best performing pieces of the baseline objects and negate the poorly performing pieces.

To be practical, a systematic methodology for optimizing the shape of an object will likely require multi-objective optimization, rather than single-objective optimization, as studied here. Here, we have considered only single-objective optimization by minimizing the drag on a train because this is the first study of design-by-morphing, and we are more concerned here with the proof of its concept rather than its practicality. One way of doing multi-objective optimization is with a Pareto optimal solution (c.f., Nemec et al. [42] and Jiaqi and Feng [30]), which could be used with design-by-morphing. However, for many applications, an easier way of allowing design-by-morphing to be more practical and to carry out multi-objective optimization is to define a cost function in terms of several criteria that need to be maximized or minimized, as was done in the study by Jakubek and Wagner [27]. For example, to decrease both the drag on a train and the amount of its rocking due to vortex shedding, a cost function to be minimized could be defined to be the sum of one weighting coefficient multiplied by the drag and another weighting coefficient multiplied by the amplitude of the rocking, where the values of the weights are a subjective choice of the designer. If it were difficult to compute efficiently the amplitude of the rocking, the cost function could be defined as a function of the torque induced on the train by a shedding vortex, where the function of the torque serves as a proxy for the amplitude of the rocking. There are many costs that might be difficult to compute numerically but need to be minimized. For example, the manufacturability and maintenance of an object are often major costs, and therefore of practical concern in an optimization. The surface of the optimal train in Fig. 14 has small features that might be difficult to manufacture, lead to large stresses and diminished lifetimes, or correspond to small grooves that might be expensive to clean. A proxy for those costs might be the minimum radius of curvature of the shape, which could be incorporated into the cost function.

Although some geometric constraints can be directly written into the algorithm that defines an object's shape, such as the constraints we imposed in Sect. 5.3 where the back of the nose was forced to fit to the front of the passenger car, some cannot. For example, a train may need to pass through a tunnel, which would require that its maximum width and height could not be greater than imposed values. The easiest way to enforce these types of constraints is to include them in the cost function as penalties. The cost function would be defined to increase very rapidly if the maximum width or height of a morphed shape came close to or exceeded the imposed values (c.f., Demeulenaere et al. [15,16]).

Our original philosophy behind design-by-morphing was that shapes of objects that have stood the test of time and that are currently used, must have good features. By blending or morphing those features, a better design can be produced. Therefore, we chose our baseline shapes to be existing shapes, and, for example, not CAD shapes or their unions. The study presented here has already shown a flaw in this philosophy by demonstrating that the inclusion of baseline shapes with bad features teaches our design algorithm features to avoid.

It is currently beyond anyone's ability to determine the optimal set of baseline shapes to use in design-by-morphing, but we are exploring two avenues for baselines shapes that are not shapes of existing objects. In Oh [43], we considered the optimal design of a sharp heeled draft tube. Only one of the baseline shapes was an existing object because no others were readily available. We created additional baseline shapes using CAD tools and intuition. We then morphed those shapes to find an optimal design (which differs from creating a new design by changing the radii, angles, heights, etc. of CAD shapes as in Demeulenaere et al. [15,16]). We found improvements in the performance of the draft tube of 11% with design-by-morphing using these additional baseline shapes. A second avenue for developing baseline shapes is to allow a neural network to create them. Just as a neural network can be taught to determine whether an object is a cat or a dog from its appearance, we have been able to teach a neural network whether a 3D shape is an airplane. Once a neural network has that knowledge, it can create new airplane baseline shapes. Numerical experiments with these baselines are currently underway.

Due to the versatility of design-by-morphing, we believe that it can be extended to many engineering problems. This automated method, with its ability to extrapolate as well as interpolate existing designs, can find optimal designs that are radical departures from existing ones that might not be considered "intuitive" by engineers nor be obtained using traditional design methods.

Acknowledgements We thank Alain Demeulenaere and David Gutzwiller for valuable contributions to the science and engineering

results and to NUMECA, USA, Inc. for software and computational support. Partial support was supplied by NSF Grants AST-1009907 and AST-1510703 and by NASA PATM Grants NNX10AB93G and NNX13AG56G. Partial support for computational work was provided by NSF XSEDE (NSF OCI-1053575) and NASA-HEC.

Appendix

We require that all of the designs of the leading-and-trailing cars that we create by morphing baseline leading-and-trailing cars have continuous surfaces. In particular, we require that at the interface labeled I_p in Fig. 3 where the nose joins the passenger compartment that the surfaces are sufficiently smooth that (I) the locations of the surfaces of the passenger compartment and the nose at I_p are continuous, and (II) that the slopes of the tangent planes of the surfaces at I_p also be continuous. Here we show that sufficient conditions for this smoothness are

1. The passenger compartments of all of the baseline leading-and-trailing cars have the same shape.
2. At the interface I_p, all the baseline shapes satisfy conditions (I) and (II) listed above.
3. The sum of the weights ω_i used in the morph in Eq. 10 sum to unity.

Because the train's is symmetric we need only concern ourselves with $\pi/2 \geq \phi \geq 0$ and $\pi/2 \geq \theta \geq 0$ (see Fig. 3). Let $R_k(\theta, \phi)$ be the surface of the kth baseline nose. I_p is located on the $x = 0$ plane at $\phi = \pi/2, \pi/2 \geq \theta \geq 0$. The location, $\widetilde{R}(\theta)$ of the interface I_p of each of the N baseline leading-and-trailing cars is the the same because all of the passenger compartments are identical:

$$\widetilde{R}(\theta) = R_k(\theta, \pi/2), \quad k = 0, 1, \ldots, N-1, \tag{15}$$

The radius of a morphed train nose $R(\theta, \phi)$ is obtained from Eqs. 9 and 10, so it can be rewritten as

$$R(\theta, \phi) = \sum_{k=0}^{N-1} \omega_k R_k(\theta, \phi). \tag{16}$$

Therefore, from Eqs. 15 and 16, if $1 = \sum_{k=0}^{N-1} \omega_k$, the location $R(\theta, \pi/2)$ of the morphed nose at I_p is $\widetilde{R}(\theta)$, so condition (I) is satisfied.

The slopes of the surfaces of two adjacent objects are continuous at their interface if the tangent planes of the two objects are the same at each each location of their interface. The latter will be true if the unit normal vectors of the tangent planes are the same at each location of the interface. Let $\mathbf{X}_k(\theta, \phi)$ be the vector from the origin (in Fig. 3) to $R_k(\theta, \phi)$ and $\mathbf{X}(\theta, \phi)$ be the vector from the origin to $R(\theta, \phi)$. So,

$$\mathbf{X} = \sum_{k=0}^{N-1} \omega_k \mathbf{X}_k. \tag{17}$$

A standard geometrical result [14,18] shows that at I_p:

$$\mathbf{n}_k(\theta) = \frac{\partial \mathbf{X}_k}{\partial \theta}\Big|_{\phi=\pi/2} \times \frac{\partial \mathbf{X}_k}{\partial \phi}\Big|_{\phi=\pi/2}, \tag{18}$$

$$\mathbf{n}(\theta) = \frac{\partial \mathbf{X}}{\partial \theta}\Big|_{\phi=\pi/2} \times \frac{\partial \mathbf{X}}{\partial \phi}\Big|_{\phi=\pi/2}, \tag{19}$$

where $\mathbf{n}_k(\theta)$ and $\mathbf{n}(\theta)$ are the unnormalized normal vectors to the tangent planes of the noses of the kth baseline nose and of the morphed nose, respectively, at θ at I_p. These normal vectors can be normalized (indicated with "hats") to provide unit vectors:

$$\hat{\mathbf{n}}_k(\theta) = \mathbf{n}_k(\theta)/||\mathbf{n}_k(\theta)||, \tag{20}$$

$$\hat{\mathbf{n}}(\theta) = \mathbf{n}(\theta)/||\mathbf{n}(\theta)||. \tag{21}$$

Substituting Eqs. 17 into 19 gives at I_p:

$$\mathbf{n}(\theta) = \sum_{j=0}^{N-1} \omega_j \frac{\partial \mathbf{X}_j}{\partial \theta}\Big|_{\phi=\pi/2} \times \sum_{k=0}^{N-1} \omega_k \frac{\partial \mathbf{X_k}}{\partial \phi}\Big|_{\phi=\pi/2} \tag{22}$$

Assumptions (1) and (2) in the first paragraph of the Appendix require that at I_p

$$\hat{\mathbf{n}}_0 = \hat{\mathbf{n}}_1 = \hat{\mathbf{n}}_2 = \cdots = \hat{\mathbf{n}}_{N-1} \tag{23}$$

and

$$\boldsymbol{\chi}(\theta) \equiv \frac{\partial \mathbf{X}_0}{\partial \theta}\Big|_{\phi=\pi/2} = \frac{\partial \mathbf{X}_1}{\partial \theta}\Big|_{\phi=\pi/2} = \cdots = \frac{\partial \mathbf{X}_{N-1}}{\partial \theta}\Big|_{\phi=\pi/2}. \tag{24}$$

With the definition of $\boldsymbol{\chi}$ in Eq. 24, Eq. 22 at I_p can be rewritten:

$$\mathbf{n}(\theta) = \sum_{j=0}^{N-1} \omega_j \boldsymbol{\chi} \times \sum_{k=0}^{N-1} \omega_k \frac{\partial \mathbf{X}_k}{\partial \phi}\Big|_{\phi=\pi/2} \tag{25}$$

$$\mathbf{n}(\theta) = \sum_{j=0}^{N-1} \omega_j \sum_{k=0}^{N-1} \omega_k \Big(\boldsymbol{\chi} \times \frac{\partial \mathbf{X}_k}{\partial \phi}\Big|_{\phi=\pi/2} \Big) \tag{26}$$

$$\mathbf{n}(\theta) = \sum_{j=0}^{N-1} \omega_j \sum_{k=0}^{N-1} \omega_k \mathbf{n}_k, \tag{27}$$

where the last equality comes from the fact that the quantity in parenthesis in Eq. 26 is $\mathbf{n}_k$. From Eqs. 20 and 23, $\mathbf{n}_k$ can be written as

$$\mathbf{n}_k(\theta) = a_k \mathbf{n}_0(\theta), \tag{28}$$

where a_k is constant and $a_0 \equiv 1$. Substituting Eq. 28 into 27, at I_p we obtain

$$\mathbf{n}(\theta) = \mathbf{n}_0(\theta) \sum_{j=0}^{N-1} \omega_j \sum_{k=0}^{N-1} \omega_k a_k. \tag{29}$$

Substituting Eq. 29 into 21 shows that the unit normal vector of the tangent plane of the morphed train nose at I_p is equal to that of baseline **0** nose at I_p, which itself is equal to that of all of the other baseline noses at I_p due to Eq. 23. Thus, condition (II) is satisfied.

References

1. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M et al (2016) Tensorflow: large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467
2. Aider JL, Beaudoin JF, Wesfreid JE (2010) Drag and lift reduction of a 3d bluff-body using active vortex generators. Experiments in fluids 48(5):771–789
3. Anjum MF, Tasadduq I, Al-Sultan K (1997) Response surface methodology: a neural network approach. Eur J Op Res 101(1):65–73
4. Baker C, Cheli F, Orellano A, Paradot N, Proppe C, Rocchi D (2009) Cross-wind effects on road and rail vehicles. Veh Syst Dyn 47(8):983–1022
5. Inc Bambardier (2010) Aeroefficient optimised train shaping. Bambardier Inc., Montreal
6. Baş D, Boyacı İH (2007) Modeling and optimization i: usability of response surface methodology. J Food Eng 78(3):836–845
7. Boyd JP (2001) Chebyshev and fourier spectral methods. Courier Corporation, North Chelmsford
8. Brechbühler C, Gerig G, Kübler O (1995) Parametrization of closed surfaces for 3-D shape description. Comput Vis Image Underst 61(2):154–170
9. Canuto C, Hussaini M, Quarteroni A, Zang T (2007) Spectral methods: evolution to complex geometries and applications to fluid dynamics. Scientific computation. Springer, Berlin
10. Chen J, Shapiro V, Suresh K, Tsukanov I (2007) Shape optimization with topological changes and parametric control. Int J Numer Methods Eng 71(3):313–346
11. Chung MK, Worsley KJ, Nacewicz BM, Dalton KM, Davidson RJ (2010) General multivariate linear modeling of surface shapes using surfstat. NeuroImage 53(2):491–505
12. Cooper R (1981) The effect of cross-winds on trains. Journal of Fluids Engineering 103(1):170–178
13. Cybenko G (1989) Approximation by superpositions of a sigmoidal function. Math Control Signals Syst (MCSS) 2(4):303–314
14. De Berg M, Van Kreveld M, Overmars M, Schwarzkopf OC (2000) Computational geometry. In: Computational geometry, Springer, Berlin pp 1–17
15. Demeulenaere A, Ligout A, Hirsch C (2004) Application of multipoint optimization to the design of turbomachinery blades. In: ASME Turbo Expo 2004: power for land, sea, and air, american society of mechanical engineers, pp 1481–1489
16. Demeulenaere A, Bonaccorsi JC, Gutzwiller D, Hu L, Sun H (2015) Multi-disciplinary multi-point optimization of a turbocharger compressor wheel. In: ASME Turbo Expo 2015: turbine technical conference and exposition, American society of mechanical engineers, pp V02CT45A020–V02CT45A020
17. Desbrun M, Meyer M, Alliez P (2002) Intrinsic parameterizations of surface meshes. Computer Graphics Forum. Wiley, Hoboken, pp 209–218
18. do Carmo MP (1976) Differential geometry of curves and surfaces. Prentice-Hall
19. DuMouchel W, Jones B (1994) A simple bayesian modification of d-optimal designs to reduce dependence on an assumed model. Technometrics 36(1):37–47
20. Duriez T, Aider JL, Masson E, Wesfreid JE (2009) Qualitative investigation of the main flow features over a TGV. In: EUROMECH COLLOQUIUM 50, vol 509
21. Elef A, Mousa M, Nassar H (2014) An efficient technique for morphing zero-genus 3D objects. Int J Phys Sci 9(13):302–308
22. Feng J, Ma L, Peng Q (1996) A new free-form deformation through the control of parametric surfaces. Comput Gr 20(4):531–539
23. Gottlieb D, Orszag SA (1977) Numerical analysis of spectral methods: theory and applications, vol 26. Siam, Philadelphia
24. Hemida HN (2006) Large-eddy simulation of the flow around simplified high-speed trains under side wind conditions. PhD thesis, Chalmers University of Technology Goteborg, Sweden
25. Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. Neural Netw 2(5):359–366
26. Hughes TJ, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: cad, finite elements, nurbs, exact geometry and mesh refinement. Comput Methods Appl Mech Eng 194(39):4135–4195
27. Jakubek D, Wagner C (2016) Adjoint-based, cad-free aerodynamic shape optimization of high-speed trains. In: Dillmann A, Heller G, Krämer E, Wagner C, Breitsamter C (eds) New results in numerical and experimental fluid mechanics X. Springer, Berlin, pp 409–419
28. Jameson A (1989) Aerodynamic design via control theory. In: Chao CC, Orszag SA, Shyy W (eds) Recent advances in computational fluid dynamics. Springer, Berlin, pp 377–401
29. Jameson A, Pierce N, Martinelli L (1998) Optimum aerodynamic design using the Navier-stokes equations. In: 35th aerospace sciences meeting and exhibit, p 101
30. Jiaqi L, Feng L (2013) Multi-objective design optimization of a transonic compressor rotor using an adjoint equation method. AIAA Paper 2732:2013
31. Kang J (2014) Design of marine structures through morphing method and its supporting techniques. Marine Technol Soc J 48(2):81–89. doi:10.4031/MTSJ.48.2.7 cited By 0
32. Khuri AI, Mukhopadhyay S (2010) Response surface methodology. Wiley Interdiscip Rev Comput Stat 2(2):128–149
33. Kleijnen JP (2008) Response surface methodology for constrained simulation optimization: an overview. Simul Modell Pract Theory 16(1):50–64
34. Ku YC, Kwak MH, Park HI, Lee DH (2010) Multi-objective optimization of high-speed train nose shape using the vehicle modeling function. In: 48th AIAA aerospace sciences meeting. Orlando, USA
35. Li R, Xu P, Peng Y, Ji P (2016) Multi-objective optimization of a high-speed train head based on the FFD method. J Wind Eng Ind Aerodyn 152:41–49
36. Long C, Marsden A, Bazilevs Y (2014) Shape optimization of pulsatile ventricular assist devices using FSI to minimize thrombotic risk. Comput Mech 54(4):921–932
37. Lyu Z, Kenway GK, Martins JR (2014) Aerodynamic shape optimization investigations of the common research model wing benchmark. AIAA J 53(4):968–985
38. Marcus PS (1986) Description and philosophy of spectral methods. In: Winkler K-HA, NormanML (eds) Astrophysical Radiation Hydrodynamics. Springer, Berlin, pp 359–386

39. Mocanu BC (2012) 3d mesh morphing. PhD thesis, Institut National des Télécommunications
40. Muñoz-Paniagua J, García J, Crespo A (2014) Genetically aerodynamic optimization of the nose shape of a high-speed train entering a tunnel. J Wind Eng Ind Aerodyn 130:48–61
41. Munoz-Paniagua J, García J, Crespo A, Laspougeas F (2015) Aerodynamic optimization of the nose shape of a train using the adjoint method. J Appl Fluid Mech 8(3):601–612
42. Nemec M, Zingg DW, Pulliam TH (2004) Multipoint and multi-objective aerodynamic shape optimization. AIAA journal 42(6):1057–1065
43. Oh S (2016) Finding the optimal shape of an object using design-by-morphing. PhD dissertation, University of California, Berkeley
44. Peters J (1982) Optimising aerodynamics to raise IC performance. Railw Gaz Int 10:78–91
45. Pironneau O (1974) On optimum design in fluid mechanics. J Fluid Mech 64(1):97–110
46. Poole J, Allen C, Rendall T (2014) Application of control point-based aerodynamic shape optimization to two-dimensional drag minimization. In: 52nd AIAA aerospace sciences meeting, National Harbor, Maryland, pp 2014–0413
47. Praun E, Sweldens W, Schröder P (2001) Consistent mesh parameterizations. In: Proceedings of the 28th annual conference on computer graphics and interactive techniques, ACM, pp 179–184
48. Press WH, Flannery BP, Teukolsky SA, Vetterling WT et al (1989) Numerical recipes, vol 3. Cambridge University Press, cambridge
49. Samareh J (2004) Aerodynamic shape optimization based on free-form deformation. In: 10th AIAA/ISSMO multidisciplinary analysis and optimization conference, p 4630
50. Schaeffer N (2013) Efficient spherical harmonic transforms aimed at pseudospectral numerical simulations. Geochem Geophys Geosyst 14(3):751–758
51. Shen L, Ford J, Makedon F, Saykin A (2004) A surface-based approach for classification of 3D neuroanatomic structures. Intell Data Anal 8(6):519–542
52. Shen L, Farid H, McPeek MA (2009) Modeling three-dimensional morphological structures using spherical harmonics. Evolution 63(4):1003–1016
53. Shojaeefard MH, Mirzaei A, Babaei A (2014) Shape optimization of draft tubes for agnew microhydro turbines. Energy Convers Manag 79:681–689
54. Shyy W, Papila N, Vaidyanathan R, Tucker K (2001) Global design optimization for aerodynamics and rocket propulsion components. Prog Aerosp Sci 37(1):59–118
55. Sorkine O, Alexa M (2007) As-rigid-as-possible surface modeling. In: Symposium on geometry processing, vol 4
56. Styner M, Oguz I, Xu S, Brechbühler C, Pantazis D, Levitt JJ, Shenton ME, Gerig G (2006) Framework for the statistical shape analysis of brain structures using spharm-pdm. Insight J 1071:242
57. Sun Z, Song J, An Y (2010) Optimization of the head shape of the CRH3 high speed train. Sci China Technol Sci 53(12):3356–3364
58. Hq Tian (2009) Formation mechanism of aerodynamic drag of high-speed train and some reduction measures. J Cent South Univ Technol 16:166–171
59. Vanaja K, Shobha Rani R (2007) Design of experiments: concept and applications of plackett burman design. Clin Res Regul Aff 24(1):1–23
60. Vassberg J, Jameson A (2014) Influence of shape parameterization on aerodynamic shape optimization. In: Verstraete T, Periaux J (eds) Introduction to optimization and multidisciplinary design in aeronautics and turbomachinery. Von Karman Institute Sint-Genesius-Rode, pp 1–19
61. Viana FA, Venter G, Balabanov V (2010) An algorithm for fast optimal latin hypercube design of experiments. Int J Numer Methods Eng 82(2):135–156
62. Wang X, Hirsch C, Kang S, Lacor C (2011) Multi-objective optimization of turbomachinery using improved NSGA-II and approximation model. Comput Methods Appl Mech Eng 200(9):883–895
63. Yao S, Guo D, Sun Z, Yang G (2015) A modified multi-objective sorting particle swarm optimization and its application to the design of the nose shape of a high-speed train. Eng Appl Comput Fluid Mech 9(1):513–527
64. Zhang WH, Beckers P, Fleury C (1995) A unified parametric design approach to structural shape optimization. Int J Numer Methods Eng 38(13):2283–2292