Design Optimization for Self-propulsion of a Bulk Carrier Hull Using a Discrete Adjoint Method

Ping He, Grzegorz Filip, Joaquim R. R. A. Martins, and Kevin J. Maki
Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, 48109

Abstract

Computational fluid dynamics (CFD) based optimization is becoming increasing popular in hydrodynamic design of ship hulls because it provides a fully automatic framework with a shorter design cycle than a human-supervised design tool. Despite the above advantage, CFD-based optimization requires careful attention to relevant design considerations, such that the final design is useful in practice. These considerations include all relevant objectives (such as drag and wake distortion) and constraints (such as volume, thickness, and curvature). Although constraints have been included in previous hull shape optimization studies, these studies have typically considered only one objective. To address this shortcoming, we conduct design optimization for self-propulsion by simultaneously considering drag and propeller-wake distortion. We use a gradient-based optimization framework that includes a discrete adjoint method for efficient derivative computation, which allows us to use a large number of design variables to parameterize the complex hull shape and thus gain a large amount of freedom for geometric modification. We impose appropriate geometric constraints (volume, thickness, and curvature) on the hull surface to ensure a practical design. In addition, we use a weighted objective function that includes drag and wake distortion to construct a Pareto front with five optimizations. We also consider hull-propeller interaction by comparing optimization results with and without a propeller. We use the Japan bulk carrier (JBC) as the baseline model and focus on optimizing the stern region. We find that optimizing for only one objective results in a large penalty on the other objective, whereas a weighted objective balances the penalty and achieves simultaneous improvement in drag and wake distortion. Moreover, we observe that the suction effect of the propeller suppresses the flow separation near the bilge tube and smooths out the velocity distortion at the propeller plane; these are effects that would end up affecting the optimized shapes. Our results demonstrate that it is necessary to simultaneously consider drag and wake distortion in hull-shape-optimization studies, and that constrained shape optimization with a large number of design variables is possible with the discrete-adjoint method.

Keywords
Hull shape optimization; drag; wake distortion; hull-propeller interaction; discrete adjoint; OpenFOAM

1 Introduction

Hull hydrodynamic performance is an important aspect of ship design because it determines their economic viability. Traditional hull shape designs rely heavily on the designers' experience. The process typically involves manual geometry modification followed by performance evaluation. With advances in computing techniques, however, the hull design process can be automated using the computational fluid dynamics (CFD) method integrated with an optimization algorithm [1]. The main benefit of using automated design optimization is that it reduces the length of the design cycle while achieving satisfactory design quality compared with human-supervised design tools. This advantage has been quantitatively demonstrated for aerospace applications [2], and it is plausible that the same benefit applies to ship designs.

Owing to this advantage, CFD-based hull shape optimization has received increased interest in recent years. Optimization algorithms can be divided into two categories: gradient-free and gradient-based. Gradient-free methods (e.g., genetic algorithm, particle swarm algorithm, and differential evolution) require only the values of the objective and constraint functions. Therefore, they can treat the CFD code as a black-box and are generally easy to implement. In addition, some gradient-free algorithms perform global exploration of the design space and have a higher chance of finding a point close to the global minimum compared with gradient-based algorithms; a favorable feature for handling multimodal problems. Gradient-free methods have been used in various hull shape optimization studies [3–9]. To further improve optimization efficiency, hybrid optimization strategies were developed using both global and local explorations [1, 10, 11], variable-fidelity methods were used that involved high- and low-fidelity CFD models [1, 10], and Karhunen–Lòeve expansion method was proposed to reduce the dimension of design space [12, 13].

Gradient-based methods require not only the values but also the derivatives of objective functions and constraints. Gradient-based methods typically perform local exploration of design space by starting from the baseline design and using the gradient (derivative) information to find the most promising direction for improvement. To efficiently compute the derivatives, one can use the adjoint method, the computational cost of which is independent of the number of design variables [14–16]. This salient feature allows a gradient-based method to handle complex design problems with a large amount of freedom for geometric modification. In addition, the derivatives provide extra information on the function behavior, which allows an optimizer to converge to the optimum more efficiently. Although gradient-based methods are only guaranteed to converge to local minima, we have found that the design space in aerodynamic shape optimization is for the most part unimodal [17, 18]. Because unimodality is impossible to prove but easy to disprove, we assume that the design space is unimodal until we find multiple local minima. Therefore, we opt to use gradient-based optimization for hull shape design, because of its ability to handle large-scale complex design problems.

The combination of gradient-based optimization and adjoint derivative computation has been widely used in aircraft [19–23], ground vehicles [24, 25], hydrofoil [26, 27], and turbomachinery [28, 29] design optimization, as well as in hydrodynamic optimization of ship hulls [30–36]. The challenges of adjoint-based hull shape optimization include handling the naturally unsteady nonlinear interaction of ship waves and wave breaking, formulating the optimization problems with the relevant design considerations (e.g., drag, wake distortion) as well as imposing appropriate geometric and physical constraints [31]. To handle the free surface, early adjoint-based hull optimization focused on potential flow [30, 32]. Recently, Kröger et al. [35] used the volume-of-fluid method to handle the free surface in an adjoint-based hull shape optimization framework, and imposed geometric constraints to maintain the main dimension and displacement of the hull. In terms of formulating design considerations, drag [30–33] and propeller-wake distortion [34, 37] have been individually used as the objective function in hull design optimization.

Nelson et al. [38] used a low-fidelity, multidisciplinary design optimization model to optimize a hull-propeller system, and concluded that simultaneously considering multiple objectives is critical due to their tight coupling. This is because a low drag design can cause high wake distortion, which is characterized by alternate low- and high-speed wake regions in the circumferential direction at the propeller plane. When the propeller blades pass through the low-speed-wake region, blade loading decreases and cavitation can occur [37, 39]. Similarly, a low-wake-distortion design can induce a large penalty in drag. Thus, there is a need to consider both drag and wake distortion in hull design optimization.

Driven by the above motivation, we conduct a hull-shape optimization in towed and self-propelled modes using a high-fidelity, gradient-based hydrodynamic shape optimization framework. We simultaneously consider drag and wake distortion in the hull shape designs and use the adjoint method to efficiently compute the derivatives; this allows a large number of design variables to parameterize the complex hull shape and provides correspondingly broad freedom in geometric modifications. We use the discrete adjoint method because its derivatives are fully consistent with flow solutions, as discussed in our previous studies [25]. We impose geometric constraints on the hull surface (volume, thickness, and curvature) to ensure that the final designs are practical.

The baseline geometry is the Japan bulk carrier (JBC) [40] at model scale, and we focus on optimizing the stern region of the hull. This design is a capesize bulk carrier that has been the focus of propulsive efficiency improvement studies [41–44]. The design Froude number is relatively low, and our focus is hull-propeller interaction. Free-surface effects are not included; while the wave field can influence propulsive performance (it certainly alters the flow near the water surface), our objective is to demonstrate the ability to consider a large number of design variables for hull-propeller interaction with the discrete-adjoint method and RANS-based CFD. Another important focus of this work is to demonstrate the importance of optimization in self-propulsion configuration, rather than in the towed condition. To this end, we use the actuator disk method to mimic the effect of the propeller. This simplification introduces simulation errors compared to unsteady simulations with dynamic meshes for the propeller. However, it enables RANS-based optimization and significantly reduces the computational cost compared with an unsteady adjoint approach.

The rest of the paper is organized as follows. In Section 2, we introduce the optimization framework and its components. The hull shape optimization results are presented and discussed in Section 3 and we summarize our findings in Section 4.

2 Methods

In this study, we adapt our aerodynamic optimization framework [25] to perform hydrodynamic design of ship hulls. We use the gradient-based optimization approach coupled with the adjoint method to efficiently compute the total derivative $\mathrm{d}f/\mathrm{d}\boldsymbol{x}$, where f is the function of interest (drag, wake distortion, or a combination of both), and $\boldsymbol{x}$ represents the design variables that control the design surface geometry. In this section, we first introduce the overall optimization framework, followed by a description of its modules: geometry parameterization, mesh deformation, flow simulation, adjoint derivative computation, and optimization problem formulation; these descriptions are brief because a detailed description of these modules has been previously published [25]. Then, we elaborate the flow and optimization configurations for hull shape design. Finally, we compare our simulation results with experimental data to verify the CFD solver, followed by evaluation of the performance of adjoint implementation.

2.1 DAFoam: a discrete adjoint framework for gradient-based optimization

We use an open-source adjoint framework (DAFoam[1]) to perform gradient-based optimization. DAFoam consists of two major layers: OpenFOAM and Python. There are three solvers in the OpenFOAM layer: `simpleFoam`, `adjointSolver`, and `coloringSolver`. The OpenFOAM's built-in solver `simpleFoam` is used to simulate the flow. Instead of using the OpenFOAM's built-in continuous adjoint solver, we developed a discrete adjoint solver [25]. Based on the converged flow field, we then use `adjointSolver` to compute the total derivative $\mathrm{d}f/\mathrm{d}\boldsymbol{x}$. The `adjointSolver` module also uses graph coloring information, generated by `coloringSolver` [25], to accelerate the partial derivative computation.

The Python layer is a high-level interface to control the module interaction, where we take the computed objective function and its derivatives and call multiple in-house Python modules to conduct optimization. These modules are `pyGeo`, `IDWarp`, and `pyOptSparse`, and they are all open source.

[1]https://github.com/mdolab/dafoam

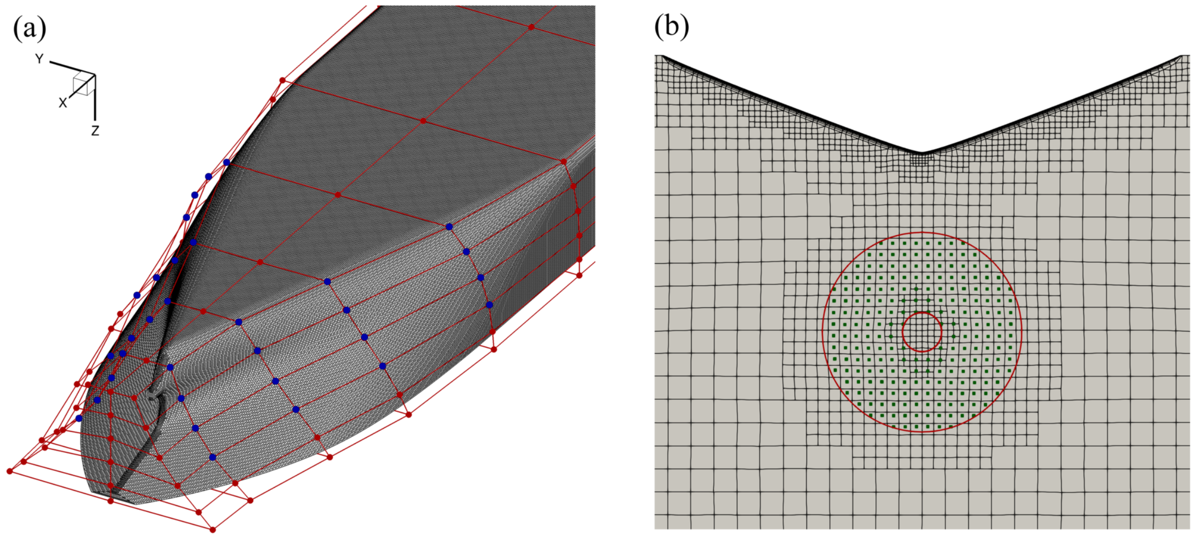

Figure 1: (a) Surface mesh and FFD points for the JBC hull. The blue points move as design variables while the red points remain fixed during optimization. (b) 2D mesh structure at the S4 plane ($x/L_{pp} = 0.98428$). We refine the propeller region to better represent the wake distortion metric σ_U^2. The red annulus is the propeller space projected onto the S4 plane, and the green squares are the sampling points for computing σ_U^2.

`pyGeo` uses the free-form deformation (FFD) approach [45] to parameterize the design surface geometry. [2] This method embeds the design surface into a tri-variate B-spline volume, as shown in Fig. 1(a). We then manipulate the surface geometry by moving the FFD points at the surface of that volume. We also use `pyGeo` to compute the values and derivatives of the geometric constraints ($\boldsymbol{c}$ and $\mathrm{d}\boldsymbol{c}/\,\mathrm{d}\boldsymbol{x}$, respectively; see examples in Sec. 2.4).

`IDWarp` uses the analytic inverse distance algorithm to deform the volume mesh, [3] similar to that described by Luke et al. [46]. The advantage of this approach is that it is applicable to both structured and unstructured meshes. Moreover, this approach better preserves mesh orthogonality in the boundary layer, compared with the methods based on the radial basis function [46]. The surface parameterization (`pyGeo`) and mesh-deformation (`IDWarp`) operations are fully parallel and typically require less than 0.1% of the CFD simulation time.

`pyOptSparse` is an extension of pyOpt [47] and provides a high-level Python interface for setting constrained nonlinear optimization problems. [4] In this paper, the SNOPT [48] optimizer is used, which adopts the sequential quadratic programming (SQP) algorithm for optimization. The constraints are handled by formulating them into the Lagrangian function, and the Hessian of the Lagrangian is approximated by using a Broyden–Fletcher–Goldfarb–Shanno update [49].

[2]https://github.com/mdolab/pygeo
[3]https://github.com/mdolab/idwarp
[4]https://github.com/mdolab/pyoptsparse

For a given optimization, we start with an initial design $\boldsymbol{x}^{(0)}$, and input it to `pyGeo`. `pyGeo` then deforms the surface mesh based on $\boldsymbol{x}^{(0)}$, and outputs the updated surface mesh coordinates $\boldsymbol{x}_s$ to `IDWarp`. `pyGeo` also computes the geometric constraints ($\boldsymbol{c}$) and their derivatives ($\mathrm{d}\boldsymbol{c}/\,\mathrm{d}\boldsymbol{x}$). To ensure a smooth mesh transition and avoid negative cell volumes, `IDWarp` deforms the volume mesh and outputs the updated volume mesh coordinates $\boldsymbol{x}_v$ to the flow solver `simpleFoam`. `simpleFoam` solves the flow and outputs the converged state variables $\boldsymbol{w}$ to `adjointSolver`. `simpleFoam` also computes the objective function f. Based on $\boldsymbol{w}$, `adjointSolver` computes the total derivative $\mathrm{d}f/\,\mathrm{d}\boldsymbol{x}$ using the discrete adjoint approach. `adjointSolver` also uses the graph coloring information from `coloringSolver` to accelerate Jacobian assembling [25]. Finally, `pyOptSparse` receives $\boldsymbol{c}$, $\mathrm{d}\boldsymbol{c}/\,\mathrm{d}\boldsymbol{x}$, f, and $\mathrm{d}f/\,\mathrm{d}\boldsymbol{x}$, solves the SQP problem, updates the design variables $\boldsymbol{x}$, and inputs $\boldsymbol{x}$ to `pyGeo` again. The above process is repeated until the optimization converges.

Next, we briefly describe two key components in the optimization process: flow simulation and adjoint derivative computation.

2.2 Flow simulation

In this paper, the OpenFOAM standard solver `simpleFoam` is used to simulate three-dimensional, steady turbulent flow. The flow is governed by the incompressible Navier–Stokes (NS) equations:

$$\nabla \cdot \boldsymbol{U} = 0, \tag{1}$$

$$\nabla \cdot \boldsymbol{U}\boldsymbol{U} + \nabla p - \nabla \cdot [(\nu + \nu_t)(\nabla \boldsymbol{U} + \nabla \boldsymbol{U}^T)] = 0, \tag{2}$$

where p is the pressure, $\boldsymbol{U}$ is the velocity vector $\boldsymbol{U} = [u, v, w]$, ν is the kinematic viscosity, and ν_t is the turbulent eddy viscosity. As mentioned in the introduction, we do not consider free-surface effects.

To connect the turbulent viscosity to the mean flow variables, the Spalart–Allmaras (SA) model is used:

$$\nabla \cdot (\boldsymbol{U}\tilde{\nu}) - \frac{1}{\sigma}\nabla \cdot [(\nu + \tilde{\nu})\nabla\tilde{\nu}] + \frac{1}{\sigma}C_{b2}|\nabla\tilde{\nu}|^2 - C_{b1}\tilde{S}\tilde{\nu} + C_{w1}f_w\left(\frac{\tilde{\nu}}{d}\right)^2 = 0. \tag{3}$$

The turbulent eddy viscosity ν_t is computed from $\tilde{\nu}$ via:

$$\nu_t = \tilde{\nu}\frac{\chi^3}{\chi^3 + C_{v1}^3}, \quad \chi = \frac{\tilde{\nu}}{\nu}. \tag{4}$$

Spalart and Allmaras [50] provide a detailed description of the terms and parameters in this model. Unlike typical continuous adjoint and some discrete adjoint implementations, we do not assume frozen turbulence, and include all the turbulence variables in our adjoint implementation.

The governing equations (1)–(3) are discretized using the finite-volume method and solved using the semi-implicit method for pressure-linked equations (SIMPLE) algorithm [51]. The Rhie–Chow interpolation scheme is used to avoid the oscillatory pressure field (checkerboard pattern issue) on a collocated mesh [52]. We choose the second-order linear-upwind scheme [53] to differentiate the divergence terms, whereas the central differential scheme is selected for the diffusion terms.

2.3 Adjoint derivative computation

2.3.1 Adjoint equations

In the discrete adjoint approach, we assume that a discretized form of Eqs. (1)–(3) is available through the flow solver, and that the design vector $\boldsymbol{x} \in \mathbb{R}^{n_x}$ and the flow state variable vector $\boldsymbol{w} \in \mathbb{R}^{n_w}$ satisfy the discrete residual equations:

$$\boldsymbol{R}(\boldsymbol{x}, \boldsymbol{w}) = 0, \tag{5}$$

where $\boldsymbol{R} \in \mathbb{R}^{n_w}$ is the residual vector.

The functions of interest are then functions of both the design variables and the flow variables, that is,

$$f = f(\boldsymbol{x}, \boldsymbol{w}). \tag{6}$$

In general, we have multiple functions of interest (the objective and multiple design constraints), but in the following derivations, we consider f to be a scalar without loss of generality. As we will see later, each additional function requires the solution of another adjoint system.

To obtain the total derivative $\mathrm{d}f/\,\mathrm{d}\boldsymbol{x}$, we apply the chain rule as follows:

$$\underbrace{\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}}_{1\times n_x} = \underbrace{\frac{\partial f}{\partial \boldsymbol{x}}}_{1\times n_x} + \underbrace{\frac{\partial f}{\partial \boldsymbol{w}}}_{1\times n_w} \underbrace{\frac{\mathrm{d}\boldsymbol{w}}{\mathrm{d}\boldsymbol{x}}}_{n_w\times n_x}, \tag{7}$$

where the partial derivatives $\partial f/\partial \boldsymbol{x}$ and $\partial f/\partial \boldsymbol{w}$ are relatively cheap to evaluate because they only involve explicit computations. The total derivative $\mathrm{d}\boldsymbol{w}/\,\mathrm{d}\boldsymbol{x}$ matrix, on the other hand, is expensive, because $\boldsymbol{w}$ and $\boldsymbol{x}$ are implicitly determined by the residual equations $\boldsymbol{R}(\boldsymbol{w}, \boldsymbol{x}) = 0$.

To solve for $\mathrm{d}\boldsymbol{w}/\,\mathrm{d}\boldsymbol{x}$, we can apply the chain rule for $\boldsymbol{R}$. We then use the fact that the governing equations should always hold, independent of the values of design variables $\boldsymbol{x}$. Therefore, the total derivative $\mathrm{d}\boldsymbol{R}/\,\mathrm{d}\boldsymbol{x}$ must be zero:

$$\frac{\mathrm{d}\boldsymbol{R}}{\mathrm{d}\boldsymbol{x}} = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{x}} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}}\frac{\mathrm{d}\boldsymbol{w}}{\mathrm{d}\boldsymbol{x}} = 0. \tag{8}$$

Rearranging the above equation, we get the linear system

$$\underbrace{\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}}}_{n_w\times n_w} \underbrace{\frac{\mathrm{d}\boldsymbol{w}}{\mathrm{d}\boldsymbol{x}}}_{n_w\times n_x} = - \underbrace{\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{x}}}_{n_w\times n_x}. \tag{9}$$

We can then substitute the solution for $\mathrm{d}\boldsymbol{w}/\,\mathrm{d}\boldsymbol{x}$ from Eq. (9) into Eq. (7) to get

$$\underbrace{\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}}_{1\times n_x} = \underbrace{\frac{\partial f}{\partial \boldsymbol{x}}}_{1\times n_x} - \overbrace{\underbrace{\frac{\partial f}{\partial \boldsymbol{w}}}_{1\times n_w} \underbrace{\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}}^{-1}}_{n_w\times n_w}}^{\boldsymbol{\psi}^T} \underbrace{\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{x}}}_{n_w\times n_x}. \tag{10}$$

Now we can transpose the Jacobian and solve with $[\partial f/\partial \boldsymbol{w}]^T$ as the right-hand side, which yields the *adjoint equations*,

$$\underbrace{\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}}^T}_{n_w \times n_w} \underbrace{\boldsymbol{\psi}}_{n_w \times 1} = \underbrace{\frac{\partial f}{\partial \boldsymbol{w}}^T}_{n_w \times 1}, \tag{11}$$

where $\boldsymbol{\psi}$ is the *adjoint vector*. Then, we can compute the total derivative by substituting the adjoint vector into Eq. (10):

$$\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \frac{\partial f}{\partial \boldsymbol{x}} - \boldsymbol{\psi}^T \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{x}}. \tag{12}$$

For each function of interest, we need to solve the adjoint equations only once, because the design variable is not explicitly present in Eq. (11). Therefore, its computational cost is independent of the number of design variables, but proportional to the number of objective functions. This approach is also known as the *adjoint method* and is advantageous for hull design, because we typically have only a few functions of interest but may use several hundred design variables.

To summarize, a discrete adjoint consists of four major steps: 1. Compute the partial derivatives $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]^T$ and $[\partial f/\partial \boldsymbol{w}]^T$; 2. Solve the linear equation (11) for the adjoint vector $\boldsymbol{\psi}$; 3. Compute the partial derivatives $\partial \boldsymbol{R}/\partial \boldsymbol{x}$ and $\partial f/\partial \boldsymbol{x}$; 4. Use Eq. (12) to compute the total derivative $\mathrm{d}f/\mathrm{d}\boldsymbol{x}$. These four steps indicate that an effective adjoint implementation requires efficient partial derivative computation and efficient and robust adjoint equation solution. We elaborate on the details of these two tasks in the following sections.

2.3.2 Partial derivative computation and adjoint equation solution

We use the finite-difference method to compute $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]^T$ and $[\partial f/\partial \boldsymbol{w}]^T$. However, naively computing these partial derivatives using finite differences requires calling the residual and objective computation routines n_w times, where n_w is the number of columns in $\partial \boldsymbol{R}/\partial \boldsymbol{w}$ and $\partial f/\partial \boldsymbol{w}$. This becomes computationally prohibitive for three-dimensional problems, because n_w is at least a few million for useful problems.

To circumvent the above issue, we accelerate partial derivative computation using the graph coloring method [54, 55]. Taking the state Jacobian $\partial \boldsymbol{R}/\partial \boldsymbol{w}$ as an example, we group all the states (columns) into different colors such that no two states impact the same residual (row) in each color. Using this coloring information, we can then simultaneously perturb multiple columns that have the same color and compute their partial derivatives by calling the residual function only once. In previous work, we developed a heuristic graph coloring algorithm that runs in distributed memory systems in parallel [25]. The central idea of this algorithm is to tentatively assign colors for the local meshes which are owned by local processors and then resolve conflicts by exchanging coloring information between local meshes through message passing interface (MPI) communication. Our coloring algorithm is applicable for structured and unstructured meshes and assembles the entire state Jacobian matrix by calling the residual computation routine $\mathcal{O}(1000)$ times, instead of n_w times. In addition, the number of colors only weakly depends on the size of the meshes or on the number of CPU cores when running coloring in parallel [25]. For example, the number of colors for the coarse (2.2 million cells), medium (6.6 million cells), and fine (19.7 million cells) meshes used in this study (see Table 3) are 2622, 2350, and 2454, respectively.

We do not use the coloring scheme for $\partial \boldsymbol{R}/\partial \boldsymbol{x}$ and $\partial f/\partial \boldsymbol{x}$; we use a brute-force finite-difference method instead, because it is fast and accurate enough [25]. We compute $\mathrm{d}\boldsymbol{x_v}/\,\mathrm{d}\boldsymbol{x}$ by first perturbing each $\boldsymbol{x}$ and then deforming the surface and volume meshes using `pyGeo` and `IDWarp`. We then pass $\mathrm{d}\boldsymbol{x_v}/\,\mathrm{d}\boldsymbol{x}$ to `adjointSolver` to compute $\partial \boldsymbol{R}/\partial \boldsymbol{x}$ and $\partial f/\partial \boldsymbol{x}$ directly. The computational cost for $\partial \boldsymbol{R}/\partial \boldsymbol{x}$ and $\partial f/\partial \boldsymbol{x}$ scales with the number of design variables, whereas the cost for $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]^T$ and $[\partial f/\partial \boldsymbol{w}]^T$ scales with the number of colors. Computing $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]^T$ is the most expensive part of the method, because we typically have $\mathcal{O}(100)$ design variables but $\mathcal{O}(1000)$ colors for $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]^T$ [25].

After computing $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]^T$ and $[\partial f/\partial \boldsymbol{w}]^T$, Portable, Extensible Toolkit for Scientific Computation (PETSc) software library [56–58] is used to solve the adjoint equations (11). We choose the generalized minimal residual (GMRES) method as the top-level linear solver and adopt a nested preconditioning strategy. To be more specific, the additive Schwartz method with one level of overlap is used as the global preconditioner. The additive Schwartz method divides the linear system into sub-blocks and allows us to solve them in parallel. For the local preconditioner in each sub-block, incomplete lower and upper (ILU) factorization is selected. To improve the effectiveness of ILU, one level of extra fill-in is used. A segregated algorithm (SIMPLE) is used to solve the nonlinear Navier–Stokes equations (1)–(3), whereas a fully coupled linear solver (GMRES) is used for the adjoint equations (11).

2.4 Flow and optimization configurations

The OpenFOAM standard solver `simpleFoam` is used to conduct the flow simulations on the JBC hull at model scale, which was tested by National Maritime Research Institute [40]. The simulation domain size is $8L_{pp}$ by $2L_{pp}$ by $2L_{pp}$ in the streamwise (x), lateral (y), and vertical (z) directions, respectively, where L_{pp} is the length between perpendiculars. We generate an unstructured hex-mesh with 6 631 221 cells using the OpenFOAM built-in snappyHexMesh utility, with an average y^+ of 0.93. The surface mesh for the JBC hull is shown in Fig. 1(a). We refine the mesh near the propeller for a more accurate computation of wake distortion, as shown in Fig. 1(b). The hull is at model scale, with a Reynolds number of 7.46×10^6. The SA turbulence model is used for all the simulations and optimizations. Although other turbulence models, such as the $k - \omega$ shear stress transport (SST) model, have been shown to have better performance at predicting the vortex structures [59], we choose the SA model because it provides the most robust convergence for the adjoint equations, especially for the self-propulsion case, where the flow is complex.

As mentioned in the introduction, we perform flow simulations and optimization for both towed and self-propulsion configurations. To mimic the impact of propellers, we use actuator disk theory and add a source term in the momentum equation [60]. This source term acts to add streamwise (thrust) and circumferential (torque) body forces for cells inside an annulus space occupied by the propeller. The annulus space centers at $x_0/L_{pp} = 0.98743$, $y_0/L_{pp} = 0$, and $z_0/L_{pp} = -0.040414$; its width is $w/L_{pp} = 0.0021429$ and the inner and outer radii are $R_{\text{hub}}/L_{pp} = 0.0029571$ and $R_{\text{tip}}/L_{pp} = 0.014500$, respectively. The radial distribution of the thrust $T(\overline{R})$ is,

$$T(\overline{R}) = T_{\text{c}}\overline{R}(1-\overline{R})^{0.2}, \quad \overline{R} = \frac{R - R_{\text{hub}}}{R_{\text{tip}} - R_{\text{hub}}}, \quad R_{\text{hub}} \le R \le R_{\text{tip}}, \tag{13}$$

where $\overline{R}$ is the normalized radial location, T is the thrust, and T_{c} is the thrust coefficient. To obtain T_{c}, we run a self-propulsion simulation and adjust T_{c} such that the total thrust (obtained by integrating Eq. (13) over the propeller volume) is equal to the drag of the baseline hull. We then fix the value of T_{c} during the optimization. This simplification facilitates the Jacobian computation because dynamically adjusting T_{c} during the optimization would make the residuals and objective functions depend on all the cells inside the space occupied by the propeller. As a result of the increased dependency, $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]^T$ and $[\partial f/\partial \boldsymbol{w}]^T$ become much denser, which would increase the number of required colors. To compute the corresponding torque distribution $Q(\overline{R})$, we use [60]

$$Q(\overline{R}) = \frac{T(\overline{R})P/D}{\pi R/R_{\text{tip}}}, \tag{14}$$

where $P/D = 0.75$ is the propeller pitch ratio.

The functions of interests are drag coefficient and wake distortion. The drag coefficient is defined as

$$C_D = \frac{D}{0.5\rho U_0^2 S}, \tag{15}$$

where D is the drag force, ρ is the density, U_0 is the free stream velocity (1.179 ms^{-1}), and S is the wetted surface area. Since the Froude number (0.142) is relatively low, we ignore wave-making resistance in the drag calculation. That being said, the free surface is also ignored, and a double-body boundary condition is imposed on the calm-water plane ($z/L_{pp} = 0$).

To quantify wake distortion, we use the variance of streamwise velocity [37],

$$\sigma_U^2 = \frac{1}{N}\sum_{i=1}^{N}(U_p^i - \overline{U}_p)^2, \tag{16}$$

where U_p^i is the discrete streamwise velocity at the sampling points and $\overline{U}_p$ is the averaged streamwise velocity for these points. The sampling points are located within the projected annulus area on the S4 plane, which are shown in Fig. 1(b) as the green squares.

As mentioned above, we focus on optimizing the stern region of the JBC hull. The objective function is a weighted function of the drag coefficient and the wake distortion,

$$f = W_{C_D}(C_D/{C_D}^B) + W_{\sigma_U^2}(\sigma_U^2/{\sigma_U^2}^B) \tag{17}$$

Table 1: Summary of optimization cases. The objective function is defined in Eq. 17. We run five cases with different weights and physical constraints. In Opt2, W_{C_D} is 0.98 and 0.97 for the towed and self-propulsion configurations, respectively. For all other cases, W_{C_D} and $W_{\sigma_U^2}$ are the same across the towed and self-propulsion configurations.

Cases	Weights ($W_{C_D} + W_{\sigma_U^2} = 1$)	σ_U^2 constraint	Objective function
Opt0	$W_{C_D} = 1.00$	No	Drag
Opt1	$W_{C_D} = 0.98$ (towed), 0.97 (propulsion)	No	Weighted objective
Opt2	$W_{C_D} = 1.00$	Yes	Drag
Opt3	$W_{C_D} = 0.90$	No	Weighted objective
Opt4	$W_{C_D} = 0.00$	No	Wake distortion

where W_{C_D} and $W_{\sigma_U^2}$ are weights for the drag coefficient and wake distortion, respectively, and the superscript B denotes the reference values for the baseline design. The objective function weights must add up to one ($W_{C_D} + W_{\sigma_U^2} = 1$). We conduct five optimizations with different weights (Opt0 to Opt4), as shown in Table 1.

The optimization configurations are summarized in Table 2. We set 32 FFD control points to manipulate the hull shape, as shown in Fig. 1(a). We allow the FFD points to move only in the lateral direction (y) such that the draft of the hull is unchanged. For the wake distortion constraint, we limit σ_U^2 in Opt2 to be no larger than its initial value. For all other optimization cases, we do not impose constraints for σ_U^2.

Table 2: Constrained shape optimization problem for the JBC hull.

	Function or variable	Description	Quantity
minimize	$W_{C_D} \dfrac{C_D}{{C_D}^B} + W_{\sigma_U^2} \dfrac{\sigma_U^2}{{\sigma_U^2}^B}$	Weighted drag and wake distortion	
with respect to	Δy	FFD movement in the y direction	**32**
subject to	$\sigma_U^2 \le \sigma_{U0}^2$	Distortion not larger than its initial value (Opt2 only)	1
	$\nabla = \nabla_0$	Hull displacement volume remains unchanged	1
	$\Delta y_{\text{port}} = -\Delta y_{\text{starboard}}$	Hull shape symmetric constraint	16
	$t \le t_{\text{beam}}$	Thickness constraint to keep maximum beam	40
	$t \ge t_{\text{tube}}$	Thickness constraint for shaft installation	25
	$\kappa^{\max} \le 1.1\kappa_0^{\max}$	Maximum mean-curvature relative to baseline	1
	$-0.5 \text{ m} < \Delta y < 0.5 \text{ m}$	Design variable bounds	
		Total constraints	**84**

Imposing proper geometric constraints is critical in order to ensure a practical hull shape. To ensure a symmetric hull shape in the y direction, we set 16 linear constraints to link the displacements of FFD points between the starboard and port sides. To maintain the water line, we impose a volume constraint to ensure that hull displacement volume remains unchanged during the optimization. To this end, we generate a 2D mesh of 50 streamwise by 25 vertical points and project them to the hull surface mesh to form a 3D hexahedral domain that represents the displacement volume.

To keep the beam of the hull, we limit the y direction thickness at any location to be no larger than the original beam. The thickness constraint is imposed by projecting sampling points in a 2D mesh to the starboard and port sides, where the projection distances represent the thickness. The 2D mesh contains 5×8 points within the following ranges: $0.71 < x/L_{pp} < 0.89$ and $-0.059 < z/L_{pp} < 0$. We also impose a thickness constraint to ensure that there is sufficient space to install the propeller shaft. To this end, we setup a 5×5 mesh covering the propeller shaft near the bilge tube. The values and derivatives of volume and thickness constraints are computed by `pyGeo`, as mentioned in Sec. 2.1.

To consider manufacturing cost, we impose a mean-curvature constraint to avoid kinks and sharp changes in the design surface, similar to the approach taken in our previous work [25]. For a parametric surface with the surface coordinates $\boldsymbol{x}_S = \boldsymbol{x}_S(u, v)$ and parameterization variables u and v, the mean curvature H is defined as

$$H = \frac{EN - 2FM + GL}{2(EG - F^2)}, \tag{18}$$

where

$$E = \boldsymbol{x}_u \cdot \boldsymbol{x}_u, \quad F = \boldsymbol{x}_u \cdot \boldsymbol{x}_v, \quad G = \boldsymbol{x}_v \cdot \boldsymbol{x}_v, \quad L = \boldsymbol{x}_{uu} \cdot \boldsymbol{n}, \quad M = \boldsymbol{x}_{uv} \cdot \boldsymbol{n}, \quad N = \boldsymbol{x}_{vv} \cdot \boldsymbol{n}, \tag{19}$$

are the coefficients from the first and second fundamental forms. The vector $\boldsymbol{n}$ is the surface unit normal and $\boldsymbol{x}_u$ is the first-order derivative of $\boldsymbol{x}$ with respect to u. We use the finite-difference method to compute these derivatives. Given Eqs. (18) and (19), we derive $\mathrm{d}H/\,\mathrm{d}\boldsymbol{x}_S$ analytically and compute its value in a similar manner. We need a special treatment to facilitate the finite-difference derivative computation in Eq. (19). This is because the above curvature formulations assume a u-v parametric surface, whereas we use an unstructured mesh whose surface pattern is irregular. To circumvent this issue, we generate a new patch of 2D structured mesh for the hull surface, covering only $0.74 < x/L_{pp} < 0.97$ and $-0.051 < z/L_{pp} < 0$. We then use `pyGeo` to compute H and $\mathrm{d}H/\,\mathrm{d}\boldsymbol{x}_S$ based on this patch instead of the unstructured mesh used in CFD. During the optimization process, `pyGeo` applies the same morphing for the unstructured mesh and this patch, such that the computed curvature and its derivatives match the values on the actual hull surface. Finally, we use `pyGeo` to map the surface coordinate derivatives ($\mathrm{d}H/\,\mathrm{d}\boldsymbol{x}_S$) to the design variable derivatives ($\mathrm{d}H/\,\mathrm{d}\boldsymbol{x}$).

Instead of constraining the averaged H on a design surface, as in our previous work [25], we limit the maximal H on the hull surface to be no larger than 10% of baseline maximum H. This is done by aggregating all the local H on the hull surface using the Kreisselmeier–Steinhauser (KS) function [61, 62],

$$c_H = \frac{1}{\rho}\ln\left(\sum_{i=1}^{n} e^{\rho H_i}\right), \tag{20}$$

where H_i is the local curvature on the ith discrete surface mesh and ρ=6000. In total, we have set 84 constraints. The symmetry, volume, and thickness constraints are always satisfied during the optimization process. The wake distortion and curvature constraints can become infeasible when the optimizer explores the design space; however, they are eventually satisfied when the optimizations converge (see Fig. 3).

2.5 CFD validation and uncertainty analysis

We validate the CFD solver by comparing the numerical simulation results with experimental data measured by National Maritime Research Institute (NMRI) [40]. For the towed case, the simulated C_D is 4.006×10^{-3}, which is 6.6% lower than the measured value of 4.289×10^{-3}. This drag underestimation is probably due to the double-body boundary condition and the exclusion of wave-making resistance in our simulations. However, for the self-propulsion configuration, no such underestimation is observed, and the simulated C_D (4.810×10^{-3}) agrees well with the measured value 4.811×10^{-3}, an error of 0.02%. This better agreement does not necessarily indicate that the self-propulsion simulations are more accurate than the towed case or that wave-making resistance is negligible, because there are errors in the actuator disk approximation. For optimization, the CFD's ability to correctly predict trends is more important than its ability to predict absolute values; we elaborate on this in Sec. 3.3.

Figure 2 shows the streamwise velocity contour at the planes corresponding to x/L_{pp} values of 0.96250 (S2), 0.98428 (S4), and 1 (S7). Overall, the simulated velocity distribution qualitatively agrees with the measurements. However, the measured low-velocity region ($y/L_{pp} \approx -0.175$, $z/L_{pp} \approx -0.04$), associated with the longitudinal bilge vortex, is not quantitatively captured in the simulated velocity field at the S2 plane. We speculate that this is because the SA turbulence model tends to underestimate the strength of the longitudinal bilge vortex, as was also observed by Larsson et al. [59]. As shown in Fig. 2, the velocity measured at the S4 plane is discrete and has many missing data; therefore, we are not able to compute an accurate experimental σ_U^2 for comparison.

Because CFD is subject to errors, we evaluate the numerical uncertainty (U_N) of the simulation results, following Stern et al. [63]. For steady-state CFD simulations, U_N includes errors due to poor flow convergence and errors due to spatial discretization. As we show in Sec. 2.6, our CFD simulations are well converged, and therefore the convergence error is negligible. To evaluate discretization errors, we compare the simulated C_D and σ_U^2 using meshes with 2.2 (coarse), 6.6 (medium), and 19.7 (fine) million cells. The numerical uncertainty is then computed as:

$$U_N = \frac{F_S|\delta_{RE}|}{V_m} \times 100\%, \tag{21}$$

where F_S=1.25 is the factor of safety [64], V_m is the value of objective function (C_D or σ_U^2) given by the medium mesh, and δ_{RE} is the error computed from the Richardson extrapolation given by

$$\delta_{RE} = \frac{V_m - V_f}{r^p - 1}, \quad p = \frac{\ln\left(\dfrac{V_c - V_m}{V_m - V_f}\right)}{\ln(r)}, \tag{22}$$

where r is the spacing ratio for mesh refinement, and subscripts c and f represent the coarse and fine meshes, respectively.

The above formulations are used to compute U_N for C_D. As shown in Tables 3, 6, and 7, the σ_U^2 changes between the meshes is much larger than C_D, and mesh convergence

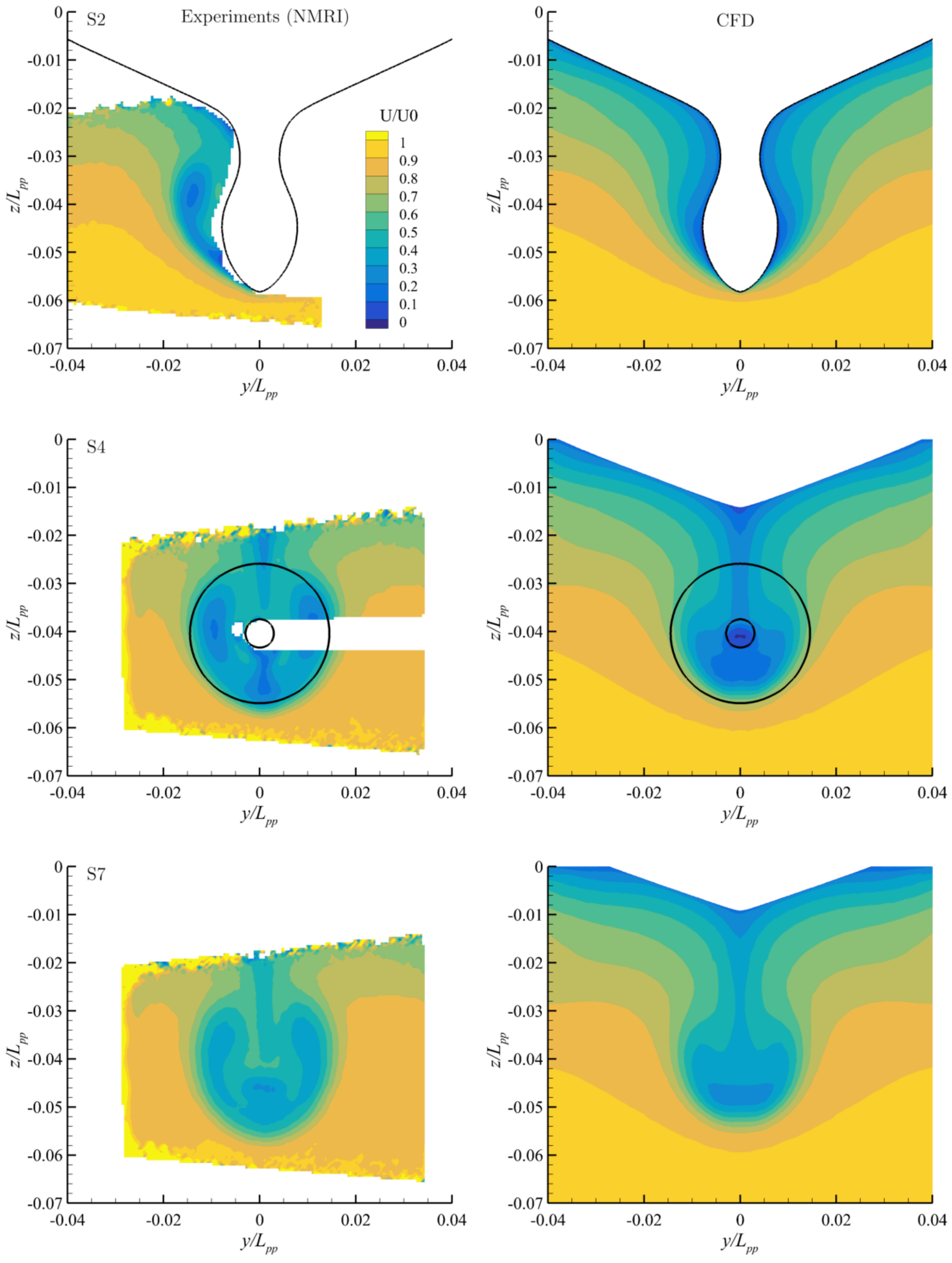

Figure 2: The simulated velocity contours qualitatively agree with the experimental data [40]. However, the strength of the longitudinal bilge vortex is underestimated.

is not always achieved. Therefore, for σ_U^2, the uncertainty is computed using half of the difference between the largest and smallest values in the samples,

$$U_N = \frac{1}{2}\left(V_U - V_L\right), \tag{23}$$

where the subscripts L and U are the upper and lower bounds of the objective function values obtained from the three mesh densities, respectively.

As shown in Table 3, the uncertainty in C_D is less than 0.3%, indicating that good mesh convergence is achieved. In contrast, for σ_U^2, the simulation results are more sensitive to the mesh size, resulting in much larger uncertainties. This is expected for unstructured mesh simulations, because changing the mesh density changes the number and distribution of σ_U^2 sampling points at the S4 plane (see Fig. 1b for an example of sampling points), which increases the uncertainty in σ_U^2 computation.

Table 3: Impact of mesh sizes on the simulation results. The numerical uncertainty (U_N) for σ_U^2 is much higher than for C_D. Moreover, the uncertainty for the self-propulsion case is larger than for the towed case.

Mesh	Mesh cells (million)	C_D	σ_U^2
Towed			
Coarse	2.2	0.004220	0.02388
Medium	6.6	0.004006	0.02966
Fine	19.7	0.003984	0.03036
U_N		0.1%	10.9%
Self-propulsion			
Coarse	2.2	0.005011	0.02540
Medium	6.6	0.004810	0.02880
Fine	19.7	0.004769	0.03390
U_N		0.3%	17.8%

Combing numerical and experimental uncertainties, we compute validation uncertainty using $U_V = (U_N^2 + U_D^2)^{1/2}$, where $U_D = 1\%$ is the experimental uncertainty [40]. For the self-propulsion case, U_V in C_D (1.0%) is larger than the error (0.02%) and thus we consider the CFD validated within the uncertainty. However, for the towed case, U_V (1.0%) is smaller than the error (6.6%), and thus validation is not achieved. As discussed by Campana et al. [1], a validated CFD simulation does not necessarily lead to an improved design in the optimization, and vice versa. Therefore, to verify the optimization results, it is more appropriate to use trends other than absolute values (see Sec. 3.3).

2.6 Adjoint performance

Each optimization is run using 192 CPU cores on 8 Skylake nodes of the Stampede 2 system [65]. The Skylake nodes are equipped with Intel Xeon Platinum 8160 CPUs running at 2.1 GHz; each node has 48 CPU cores and 196 GB of memory. The towed optimizations take between 11 and 25 iterations to converge, whereas the self-propulsion cases require 7 to 17 iterations, as shown in Fig. 3. The values of C_D and σ_U^2 converge well except for Opt4 (towed). As explained in Sec. 3, this optimized shape is not practical because it has a wavy hull surface and a large penalty to drag. Therefore, we terminate the optimization at the 25th iteration to avoid wasting computational time.

We run the flow simulations for 3000 steps and the residuals decrease by 10 orders of magnitude. Each flow simulation takes 403 s and uses 46 GB of memory. For Opt0 and Opt4, each adjoint derivative computation takes 1961 s, whereas for Opt1 to Opt3, each adjoint computation takes 2818 s. The longer runtimes for Opt1 to Opt3 are because we need to solve the adjoint equations twice (once for C_D and once for σ_U^2). The performance is broken down in Table 4. We solve the adjoint equations until its residual reduces by 6 orders of magnitude. Overall, the optimizations can be done in an overnight cycle.

As expected, we need a relatively large memory (707 GB) for adjoint computation, because we store all the partial derivative matrices in memory. However, this is not a severe limitation for the current paper because modern high-performance computing systems commonly have more than 128 GB memory per node. We can further reduce memory usage by using a Jacobian-free adjoint strategy, as elaborated by Kenway et al. [66].

Table 4: Runtime and memory usage for flow simulation and adjoint derivative computation for a 6.6 million cell mesh using 192 CPU cores on 8 Skylake nodes. The adjoint total runtime is broken down into Jacobian assembly and adjoint equation solution.

	Opt0 and Opt4	Opt1 to Opt3
Runtime (s)		
Flow	533	533
Adjoint	1961	2818
Jacobian assembly	1046	1070
Adjoint solution	915	1748
Adjoint-flow ratio	3.7	5.3
Peak memory (GB)		
Flow	46	46
Adjoint	707	707
Adjoint-flow ratio	15.4	15.4

Finally, we verify the accuracy of adjoint derivatives. We compute the reference total derivatives by using the finite-difference method. We perform a finite-difference step size study by adding various perturbation magnitudes to the FFD points and then compare their total derivative values. We find that 10^{-4} m is the best step size, and we conduct similar step size studies for the partial derivatives. For the adjoint computation, we use a normalized step size of $\varepsilon_n = 10^{-7}$ to compute the partial derivatives $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]^T$ and $[\partial f/\partial \boldsymbol{w}]^T$. The actual finite-difference perturbation for each state variable is $\varepsilon_w = w_{ref}\varepsilon_n$. The state variable vector $\boldsymbol{w}$ contains U, p, ϕ, and $\tilde{\nu}$, where ϕ is the face flux. The reason to include ϕ in $\boldsymbol{w}$ was explained in our previous work [25]. For U, p, and $\tilde{\nu}$,

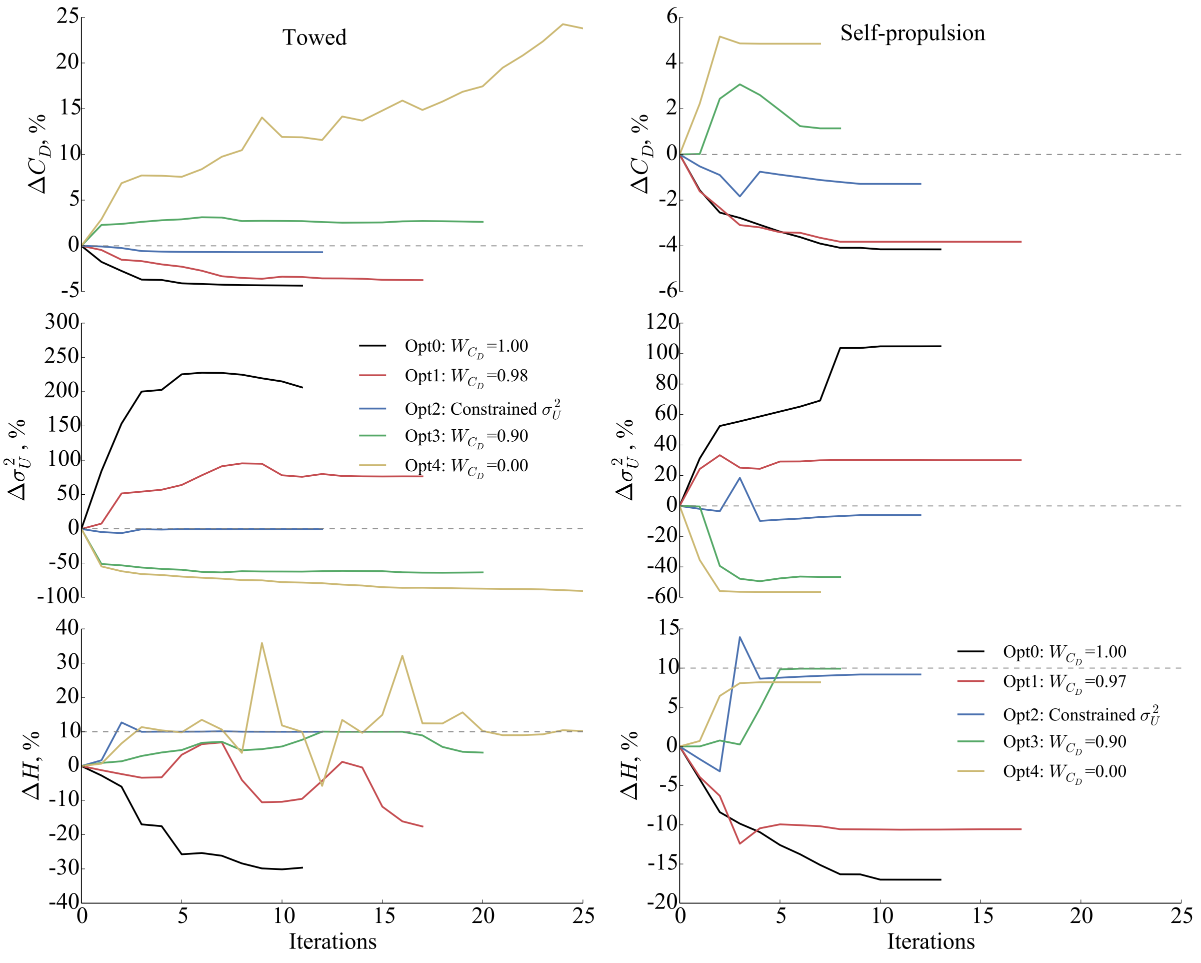

Figure 3: Optimization convergence history for the drag coefficient (C_D), wake distortion (σ_U^2), and curvature constraint (H). The towed optimizations take between 11 and 25 iterations to converge, whereas the self-propulsion cases require 7 to 17 iterations.

we use their far-field values as w_{ref}, whereas for ϕ, w_{ref} is the face area. We need the above treatment because the magnitudes of values in the $\boldsymbol{w}$ vector vary significantly, especially for $\tilde{\nu}$ and ϕ, as elaborated on in He et al. [25]. For computing $\partial \boldsymbol{R}/\partial \boldsymbol{x}$ and $\partial f/\partial \boldsymbol{x}$, the step size is 10^{-4} m. We use the baseline JBC configuration (towed; 6.6 million cells) and compare the derivatives for the first five design variables, as shown in Table 5. The adjoint derivatives agree well with the reference values, with an average error of 0.27%.

3 Results and Discussion

In this section, we present the optimization results, in terms of both drag and wake distortion at the propeller plane. We construct a Pareto front using five optimizations with different combinations of weights for drag and distortion. We analyze and interpret the optimization results by comparing shapes and flow structures between the single-

Table 5: The adjoint derivatives agree well with the reference values, with an average error of 0.27%. The reference values are computed using the finite-difference method, with a step size of 10^{-4}. The results are based on the baseline JBC configuration (towed; 6.6 million cells).

	Reference	Adjoint	Error
$\mathrm{d}C_D/\,\mathrm{d}y \times 10^{-4}$			
FFD1	-1.8844968	$-1.88\underline{56200}$	0.06%
FFD2	-1.8392580	$-1.8\underline{441308}$	0.26%
FFD3	-1.0785991	$-1.0\underline{870582}$	0.78%
FFD4	-0.8740341	$-0.8\underline{823941}$	0.96%
FFD5	-3.8162678	$-3.816\underline{6974}$	0.01%
$\mathrm{d}\sigma_U^2/\,\mathrm{d}y \times 10^{-2}$			
FFD1	2.7861729	$2.78\underline{77556}$	0.06%
FFD2	2.1410451	$2.14\underline{07703}$	-0.01%
FFD3	0.2380415	$0.23\underline{71916}$	-0.36%
FFD4	0.0821301	$0.08\underline{19794}$	-0.18%
FFD5	8.9950908	$8.99\underline{05096}$	-0.05%

and multi-objective cases. We use the JBC hull as our baseline geometry, and both towed and self-propulsion configurations are considered. We also evaluate the impact of the propeller on the optimization results. To verify our optimization results, we compare the relative magnitudes for design improvement and numerical uncertainty. Because we use a model-scale hull, we also discuss the potential impact of the Reynolds number on the optimization results.

3.1 Optimization for towed configuration

Based on the five cases summarized in Table 1, we construct a Pareto front of drag coefficient (C_D) and wake distortion (σ_U^2), as shown in Fig. 4.

For the drag-only case (Opt0), we obtain 4.4% reduction in C_D. Comparing the baseline and optimized hull shapes (Figs. 5a, 5b, and 6a), hull bilge thickness decreases and a V-shaped hull is formed. It is known that the V-shaped hull generates relatively weak longitudinal vorticity and therefore is preferable for drag reduction over the U-shaped hull in the baseline geometry [37]. However, the velocity distribution at the propeller plane (Fig. 7b) shows that the V-shaped hull generates a thin, V-shaped wake at the propeller plane. This V-shaped wake induces a large velocity distortion (σ_U^2 increases by 208.0%; see Fig. 4), which is undesirable for the propeller.

For the wake-only case (Opt4), σ_U^2 decreases by 90.6%. However, as shown in Figs. 5(f) and 6(e), we observe a wavy hull surface even though we impose a curvature constraint in the optimization. More specifically, a large bump appears near the hull bilge. Since there is no drag penalty for the Opt4 case, the optimizer creates this bump to smooth out the low-velocity region at the bottom of the propeller annulus region, as shown in Figs. 7(a) and 7(f). Similar large bumps were also observed in the wake-only case by Duvigneau et al. [37]. However, this optimized shape is impractical

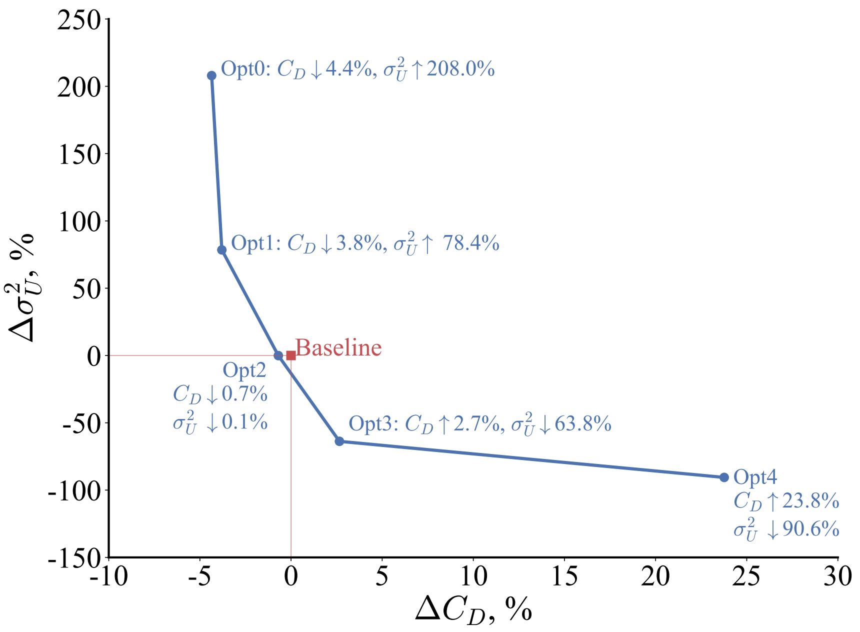

Figure 4: Pareto front for C_D and σ_U^2 (towed). Optimizing for only one objective penalizes the other objective (Opt0 and Opt4), whereas considering both C_D and σ_U^2 results in more balanced designs (Opt1, Opt2, and Opt3).

for structural and manufacturing considerations. Although the wake distortion increases at the propeller plane, as shown in Fig. 7(f), the drag increases by 23.8% (Fig. 4).

The above results (Opt0 and Opt4) confirm that simultaneously considering C_D and σ_U^2 (Opt1 to Opt3) is important in hydrodynamic design of ship hulls. To be more specific, in Opt1, we obtain 3.8% drag reduction, 0.6% lower than Opt0. However, σ_U^2 increases by only 78.4%; the σ_U^2 penalty is 2.7 times lower than Opt0. Similarly, in Opt3, we obtain 63.8% reduction in σ_U^2 with a penalty of only 2.7% drag increase, compared with the 23.8% drag increase in Opt4. Opt2 is a special optimization case where we optimize C_D while constraining σ_U^2 to be no more than its baseline value, as explained in Sec. 2.4. We obtain simultaneously improvements—C_D decreases by 0.7% and σ_U^2 decreases by 0.1%—and the hull maintains a U-shape with no wavy surfaces.

Next, we analyze the limiting streamlines (Fig. 8) to interpret the simulation results. For the baseline hull, we observe a flow bifurcation in the upper hull region, based on the limiting streamlines in Fig. 8(a). The upper branch of the bifurcation flows upward to the waterline and the lower branch flows towards the bilge tube. Near the keel, the flow moves upward and meets with the lower branch of the flow bifurcation, forming a clear convergence line. There is a flow recirculation zone above the convergence line near the bilge tube. The simulated flow convergence line and recirculation zone are qualitatively similar to the experimental measurement for the bulk carrier KVLCC2 [67, Fig. 3(b)].

Figure 9 shows the pressure distribution on the hull. There is a low pressure region upstream of the flow convergence line near the keel (Fig. 9a). As discussed by Larsson et al. [59], the location and extent of the flow convergence line and recirculation zone are

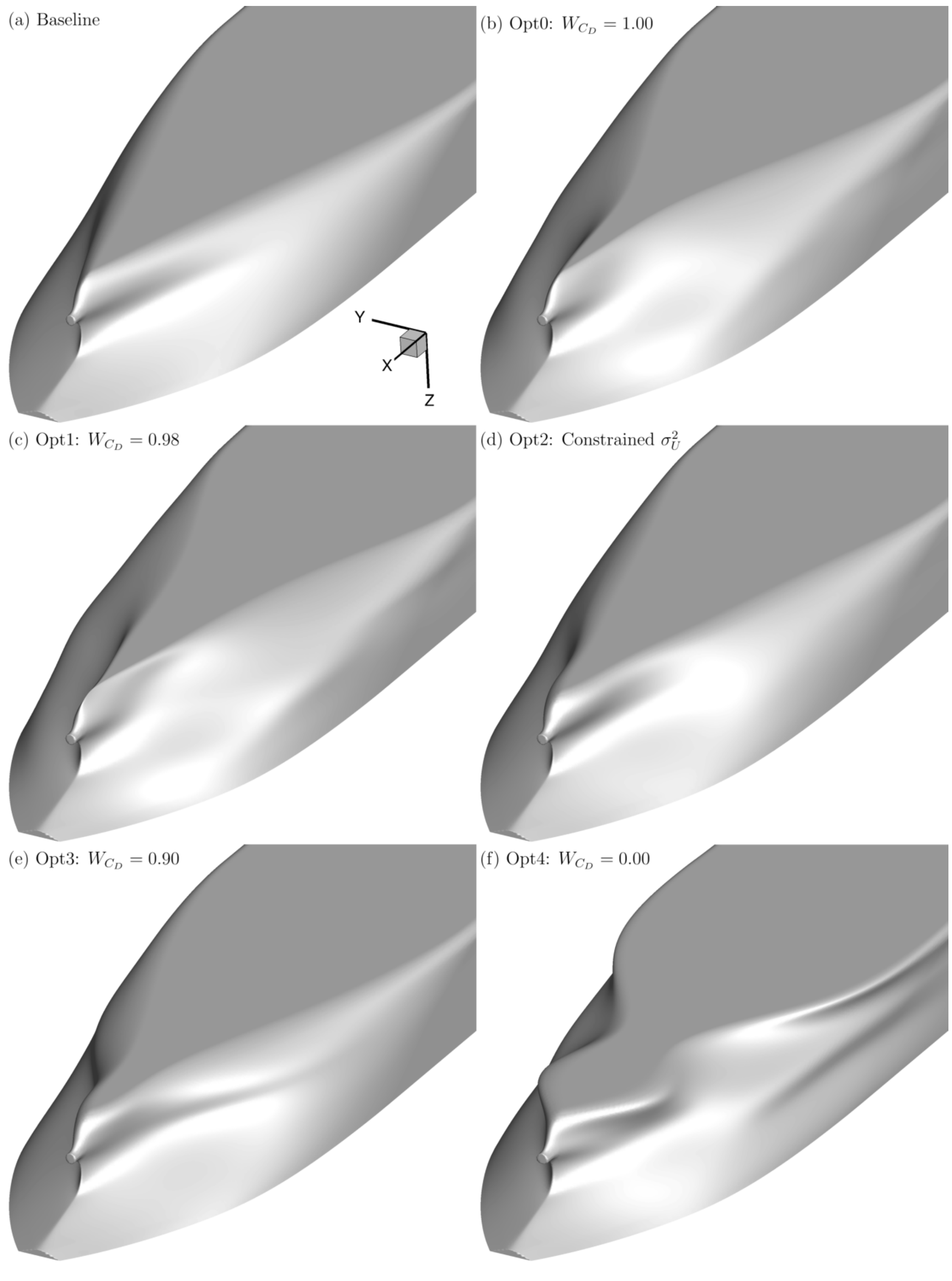
(a) Baseline
(b) Opt0: $W_{C_D} = 1.00$
Y
X
Z
(c) Opt1: $W_{C_D} = 0.98$
(d) Opt2: Constrained σ_U^2
(e) Opt3: $W_{C_D} = 0.90$
(f) Opt4: $W_{C_D} = 0.00$

Figure 5: 3D view of the baseline and optimized shapes (towed). The drag-only case (Opt0) results in a V-shaped hull, whereas the wake-only case (Opt4) has a large bump near the bilge.

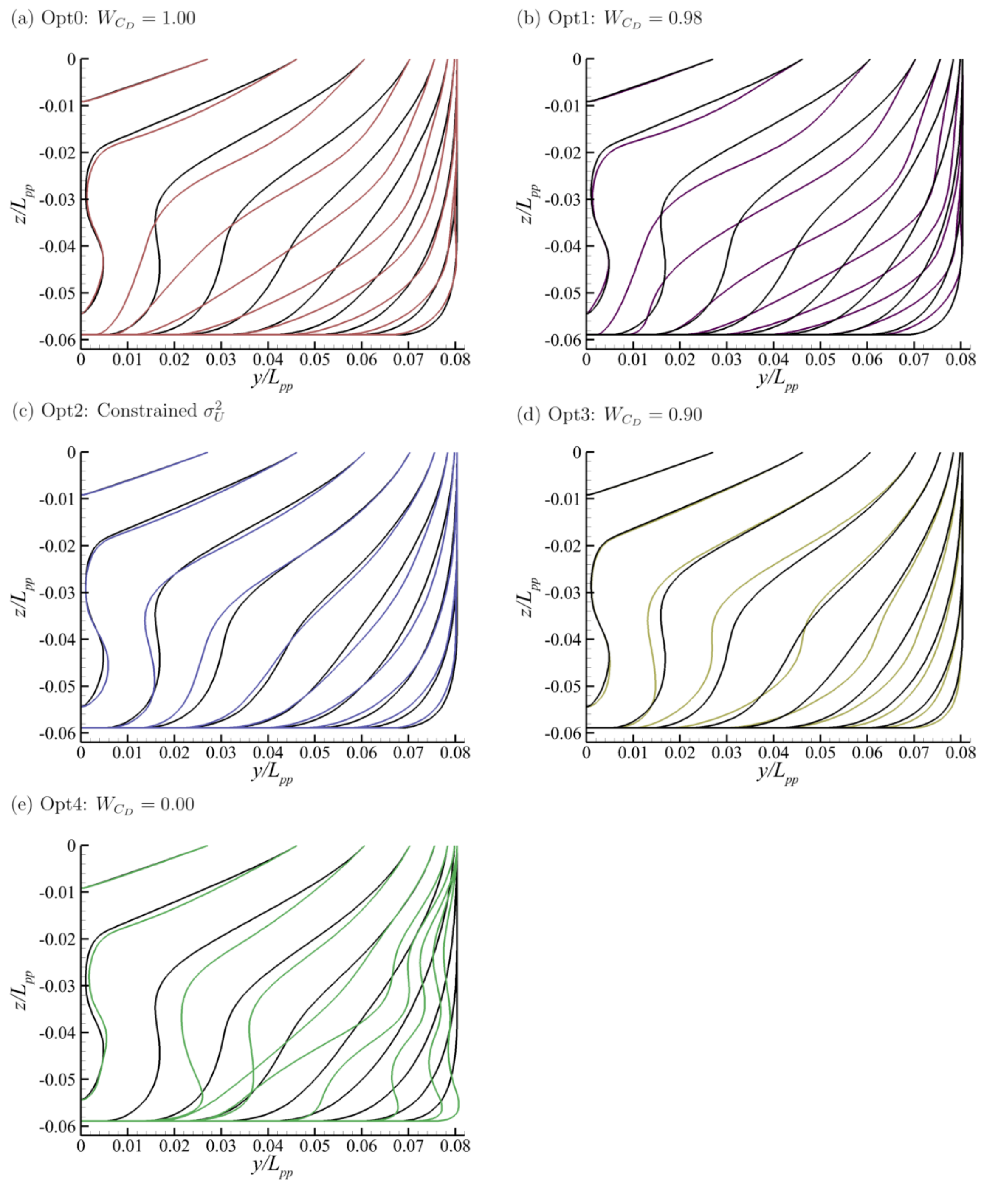

Figure 6: Comparison between the baseline (black) and optimized (colored) hull profiles (towed configuration) from $x/L_{pp} = 0.73$ to 1 with intervals $\Delta x/L_{pp} = 0.03$. The drag-only case (Opt0) results in a V-shaped hull, whereas the wake-only case (Opt4) has a large bump near the bilge.

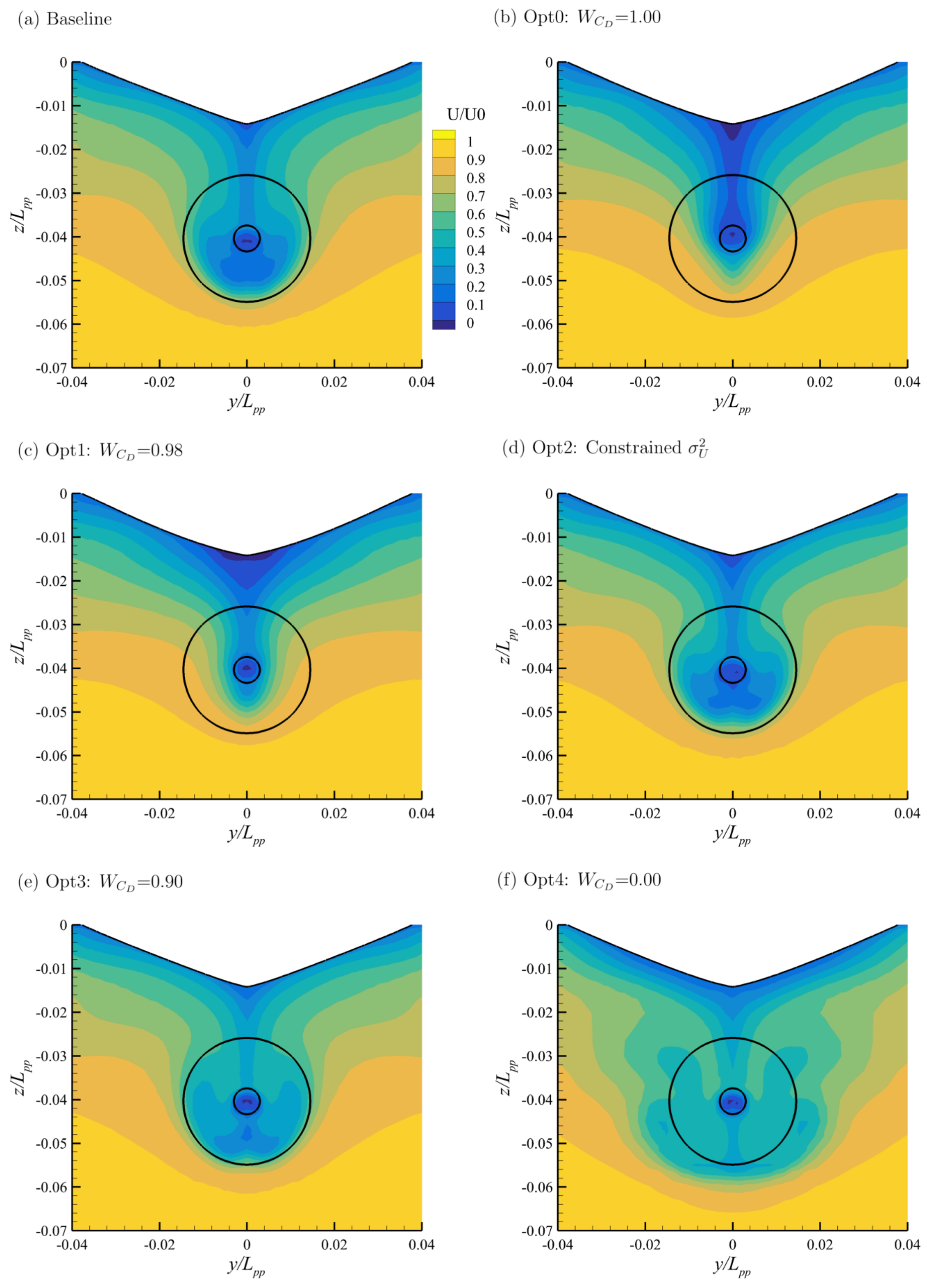

Figure 7: Streamwise velocity contour at the propeller plane (S4; towed). The drag-only case (Opt0) results in a V-shaped wake that increases the wake distortion, whereas the wake-only case (Opt4) smooths out the velocity field.

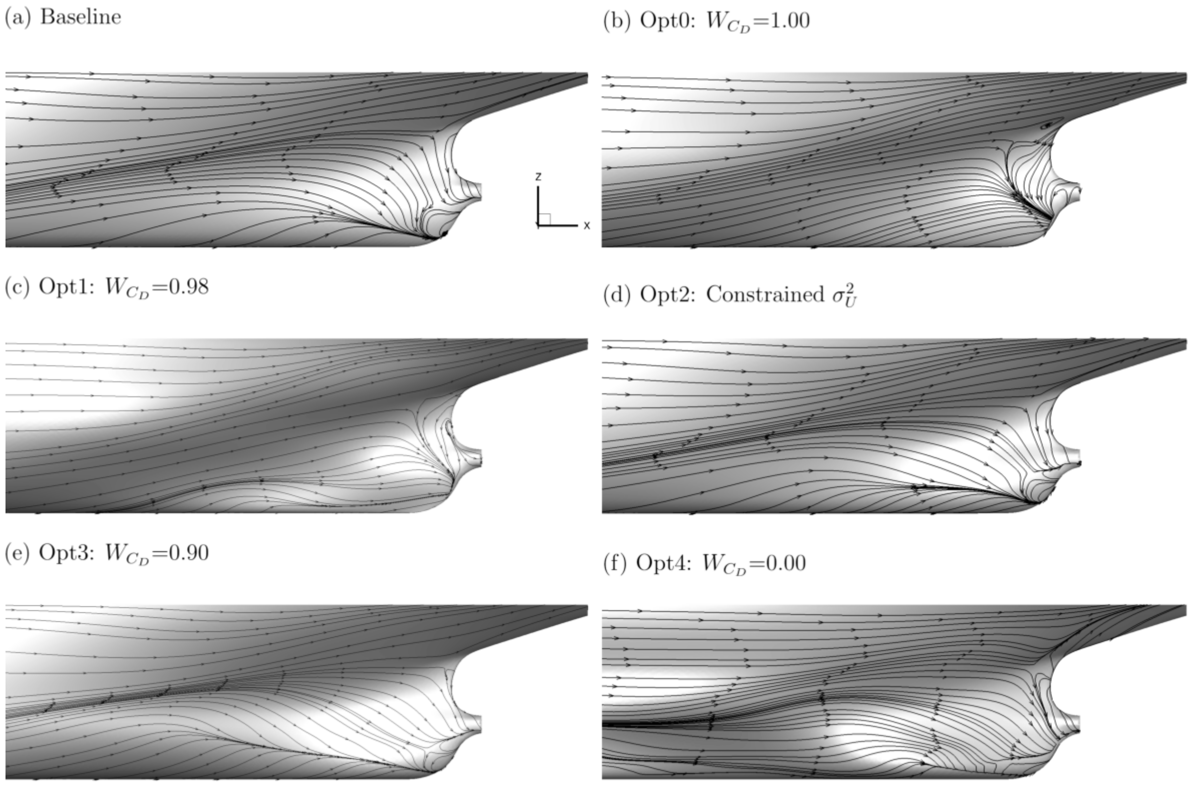

Figure 8: Limiting streamline on the hull surface for the baseline and optimized shapes (towed). The drag-only case (Opt0) results in a tilted, upward-moving flow convergence line, whereas the wake-only case (Opt4) pushes the flow convergence line downwards.

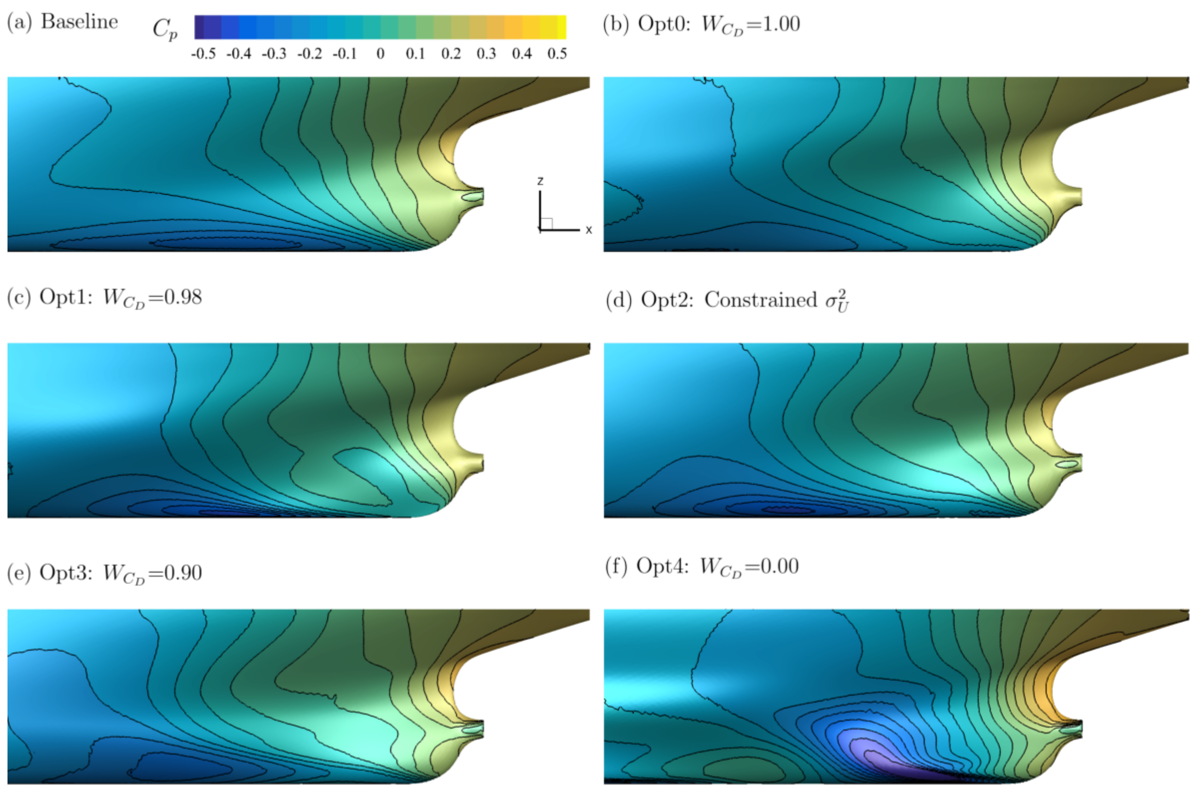

Figure 9: Pressure distribution on the hull surface for the baseline and optimized shapes (towed). The drag-only case (Opt0) results in a smoother streamwise pressure gradient, whereas the wake-only case (Opt4) induces a sharp adverse pressure gradient near the bilge.

strongly related to the onset and evolution of longitudinal bilge vortices, and therefore related to wake distortion as well. The flow convergence line is an indication of local flow separation, which eventually impacts the pressure distribution and the associated drag increase.

We now elaborate on how the near-surface flow structure and pressure distribution are modified in the optimized shapes. In Opt0, owing to the impact of the V-shaped hull, we observe a less intense adverse pressure gradient along the streamwise direction, as seen in Fig. 9(b), and the flow separation is delayed. In addition, the flow convergence line moves upward, becoming less elongated and more tilted. This upward-moving flow convergence line contributes to the V-shaped low-speed region and therefore to the large velocity distortion at the propeller plane as well, as shown in Fig. 7(b).

In Opt4, the large bump near the keel pushes the bilge vortex downwards and reduces its intensity, as shown in Fig. 8(f). This eventually increases the velocity uniformity at the propeller plane. However, the wavy bump creates a large adverse pressure gradient, as shown in Fig. 9(f), which increases the pressure drag.

In contrast, Opt1, Opt2, and Opt3 achieve more balanced designs by not having the large adverse pressure gradient that increases drag or the intense, upward-moving flow

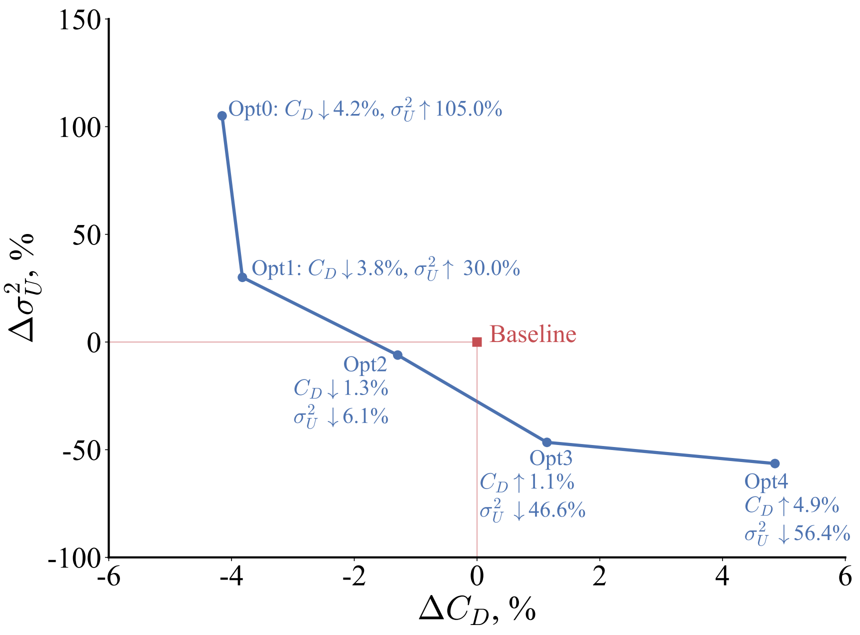

Figure 10: Pareto front for C_D and σ_U^2 (self-propulsion). Optimizing for only one objective has large penalty to the other objective (Opt0 and Opt4), whereas considering both C_D and σ_U^2 results in more balanced designs (Opt1, Opt2, and Opt3).

convergence line that distorts the wake, as shown in Figs. 8(c) to 8(e), and Figs. 9(c) to 9(e).

3.2 Optimization for self-propulsion configuration

In this subsection, we analyze the impact of the propeller on the optimization results. Figure 10 shows the Pareto front of C_D and σ_U^2 for the self-propulsion configuration. Similar to the towed configuration (Fig. 4), considering only one objective induces large penalty to the other objective in the optimization. More specifically, the C_D reduction for Opt0 is 0.4% lower than Opt1; however, its σ_U^2 increase is 74.8% higher. Moreover, going from Opt3 to Opt4, we obtain only 9.8% more σ_U^2 reduction, whereas the drag penalty is 3.8% higher. One special case is Opt2, where we obtain simultaneously improvement for C_D (1.3% reduction) and σ_U^2 (6.1% reduction). Again, the above results underscore the necessity of considering both drag and wake quality in hull-form optimization.

Next, we compare the baseline and optimized shapes, as shown in Figs. 11 and 12. Similar to the towed configuration, Opt0 forms a V-shaped hull to minimize the drag. However, a major difference is that, in contrast to the towed configuration, Opt1 and Opt2 also result in the V-shaped hulls. This is because the suction effect of the propeller accelerates the flow near the bilge tube. As a result, the flow separation is suppressed and the velocity distortion is reduced.

More specifically, by comparing Fig. 9(a) and Fig. 13(a), we observe that the pressure near the bilge tube is reduced for the self-propulsion configuration; this lower pres-

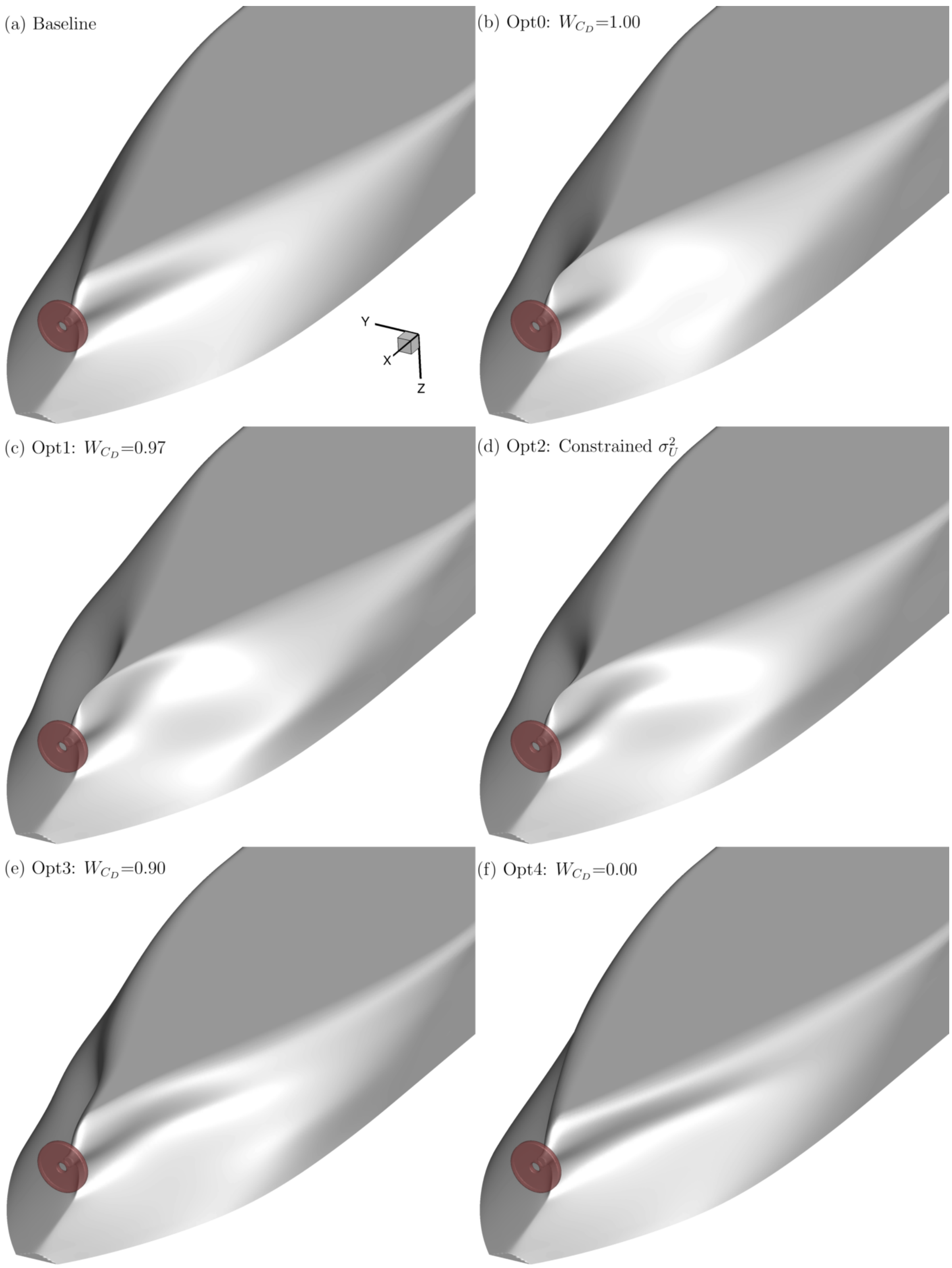

Figure 11: Baseline and optimized shapes (self-propulsion). In contrast to the towed configuration, Opt1 and Opt2 result in V-shaped hulls, whereas the large bump is absent in Opt4.

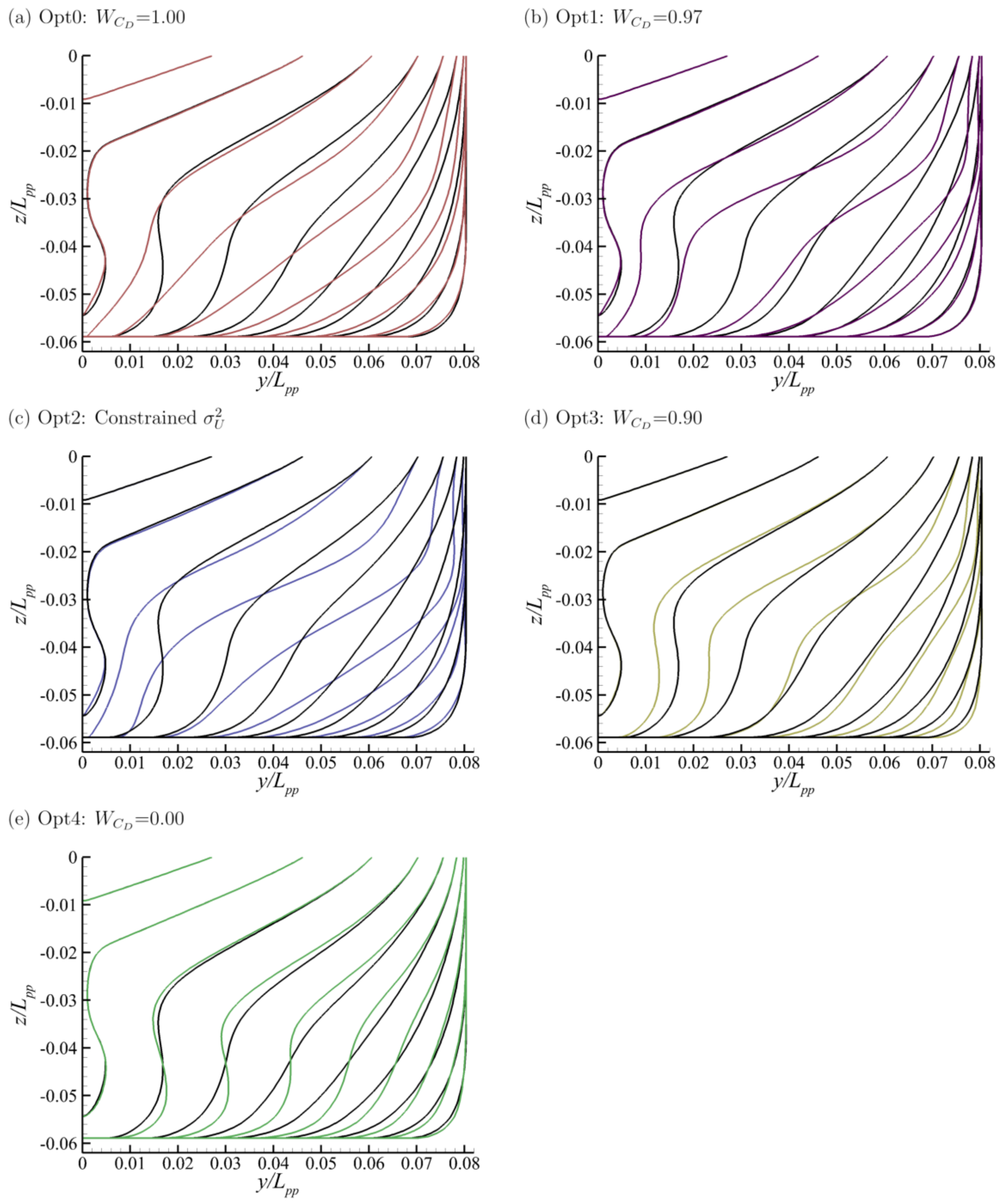

Figure 12: Comparison between the baseline (black) and optimized (colored) hull profiles (self-propulsion) from $x/L_{pp} = 0.73$ to 1.00 with interval $\Delta x/L_{pp} = 0.03$. In contrast to the towed configuration, a V-shaped hull is also observed in Opt1 and Opt2, whereas the large bump is absent in Opt4.

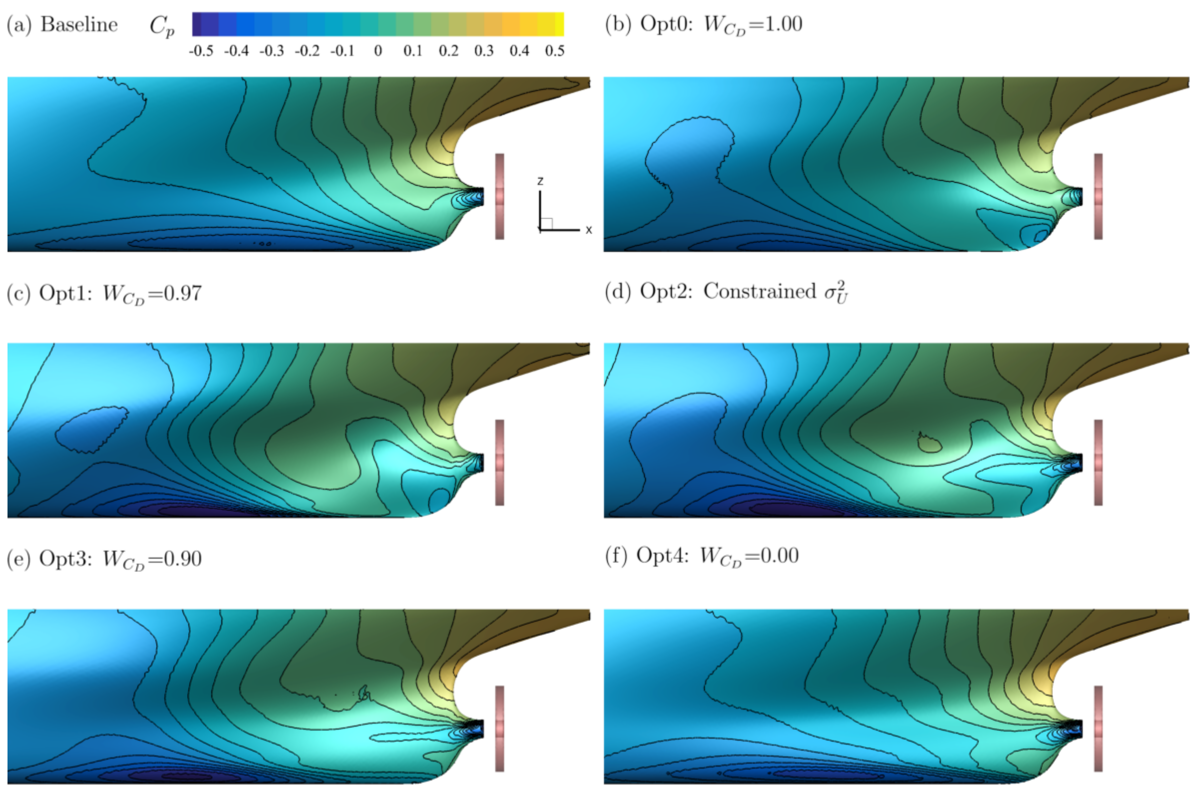

Figure 13: Pressure distribution on the hull surface for the baseline and optimized shapes (self-propulsion). Compared with the towed configuration, the pressure near the bilge tube decreases due to the suction effect.

sure suppresses the flow separation. We see this more clearly in the surface limiting streamlines by comparing Fig. 8(a) and Fig. 14(a). Although we observe similar flow convergence lines near the bilge, the flow circulation zone near the bilge tube is absent in the self-propulsion configuration, compared with the towed configuration.

By comparing Fig. 7(a) and Fig. 15(a), we observe that the low-velocity region at the bottom of the propeller annulus region is absent in the self-propulsion configuration, primarily owing to the acceleration of the flow. This accelerated flow with the resultant stronger mixing smooths out the wake distortion even for a V-shaped hull, an effect we can see by comparing Fig. 7(b) and Fig. 15(b).

Another major difference between the towed and self-propulsion configuration is the Opt4 case. For the self-propulsion configuration, we do not observe a large bump near the bilge in the optimized shape, compared with the towed configuration. Again, this is primarily because the low-velocity region at the bottom of the propeller annulus region is absent, so there is no benefit of using a large bump to smooth the wake distortion in this region.

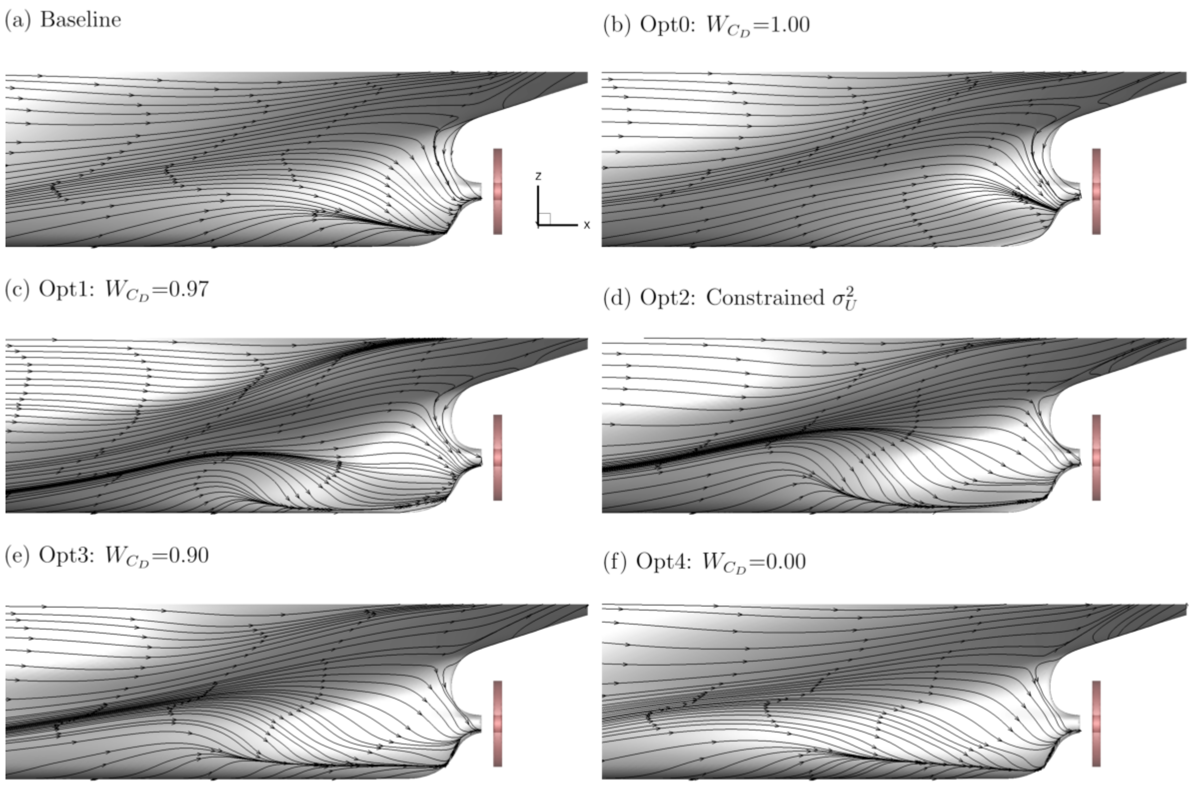

Figure 14: Limiting streamline on the hull surface for the baseline and optimized shapes (self-propulsion). Compared to the towed configuration, the circulation zone near the bilge tube is absent due to the suction effect.

3.3 Verification of optimization results

In Sec. 2.5, we evaluated the numerical uncertainty of the CFD simulations. In this section, we verify the optimization results. To this end, we compare coarse, medium, and fine mesh simulations based on the optimized shapes, as shown in Tables 6 and 7. The values in parentheses are the relative changes (design improvement or degradation) compared with baseline values at the same mesh density. For C_D, we achieve good mesh convergence for all the optimized shapes (Table 6). The numerical uncertainties are less than the relative changes; therefore, the trends in the optimizations are verified, according to Campana et al. [1]. Although the numerical uncertainties for σ_U^2 are much higher than those for C_D, they are generally lower than the relative changes (Table 7), which verifies the optimization results. Opt2 is an exception, since the uncertainties are higher than the relative changes. In this case, we optimize C_D while constraining σ_U^2 to be no larger than its baseline value. Given its high numerical uncertainty, it is challenging to obtain a verified trend for the constrained σ_U^2.

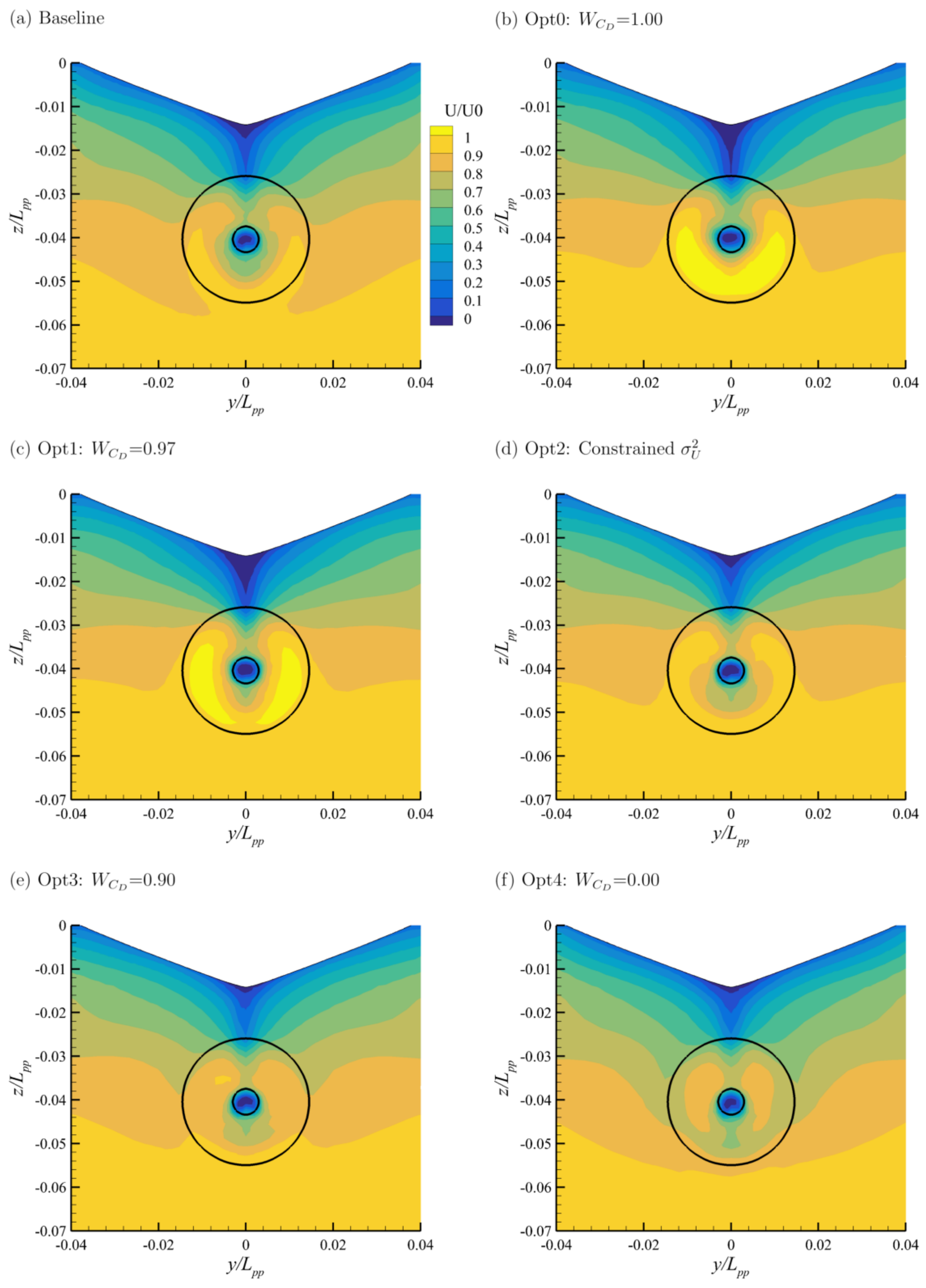

Figure 15: Streamwise velocity contour at the propeller plane (S4; self-propulsion). In contrast to the towed configuration, the V-shaped hull (Opt0, Opt1, and Opt2) has less adverse impact on wake distortion, primarily owing to the suction effect of the propeller.

Table 6: Verification of optimization results for C_D. The values in parentheses are the relative changes (design improvement or degradation) compared to baseline values at the same mesh density. The numerical uncertainties are lower than the relative changes, so that the optimizations are verified to have correct trends.

Mesh cells (million)	Opt0	Opt1	Opt2	Opt3	Opt4
Towed					
2.2	0.004031 (↓4.5%)	0.004058 (↓3.8%)	0.004191 (↓0.7%)	0.004342 (↑2.9%)	0.005172 (↑22.6%)
6.6	0.003832 (↓4.4%)	0.003854 (↓3.8%)	0.003979 (↓0.7%)	0.004113 (↑2.7%)	0.004959 (↑23.8%)
19.7	0.003828 (↓3.9%)	0.003843 (↓3.5%)	0.003954 (↓0.8%)	0.004060 (↑1.9%)	0.004888 (↑22.7%)
U_N	<0.1%	<0.1%	0.1%	0.5%	0.9%
Self-propulsion					
2.2	0.004820 (↓3.9%)	0.004810 (↓4.0%)	0.005002 (↓0.2%)	0.005110 (↑2.0%)	0.005970 (↑19.1%)
6.6	0.004610 (↓4.2%)	0.004626 (↓3.8%)	0.004748 (↓1.3%)	0.004864 (↑1.1%)	0.005043 (↑4.9%)
19.7	0.004569 (↓4.2%)	0.004579 (↓4.0%)	0.004676 (↓1.9%)	0.004801 (↑0.7%)	0.005002 (↑4.9%)
U_N	0.3%	0.4%	0.7%	0.6%	<0.1%

Table 7: Verification of optimization results for σ_U^2. The values in parentheses are the relative changes (design improvement or degradation) with respect to the baseline values at the same mesh density. Although the numerical uncertainties are higher than C_D, they are lower than the relative changes, except for Opt2.

Mesh cells (million)	Opt0	Opt1	Opt2	Opt3	Opt4
Towed					
2.2	0.08025 (↑236.1%)	0.05415 (↑126.8%)	0.02539 (↑6.3%)	0.00956 (↓59.9%)	0.00521 (↓78.2%)
6.6	0.09133 (↑208.0%)	0.05292 (↑78.4%)	0.02963 (↓0.1%)	0.01074 (↓63.8%)	0.00279 (↓90.6%)
19.7	0.13194 (↑334.6%)	0.08568 (↑182.2%)	0.02970 (↓2.2%)	0.01080 (↓64.4%)	0.00426 (↓86.0%)
U_N	28.3%	31.0%	7.3%	5.8%	43.3%
Self-propulsion					
2.2	0.04505 (↑77.4%)	0.03880 (↑52.8%)	0.02571 (↓1.2%)	0.01598 (↓37.1%)	0.00768 (↓69.8%)
6.6	0.05904 (↑105.0%)	0.03745 (↑30.0%)	0.02705 (↓6.1%)	0.01538 (↓46.6%)	0.01255 (↓56.4%)
19.7	0.06655 (↑96.0%)	0.05452 (↑60.6%)	0.03531 (↑4.0%)	0.02173 (↓36.0%)	0.01493 (↓56.0%)
U_N	18.2%	22.8%	17.7%	20.6%	28.9%

3.4 Impact of Reynolds number

In addition to the numerical uncertainty, the Reynolds number is an important parameter that affects simulation and optimization results. It has been documented that drag and vorticity structures computed at model scale can be significantly different from those of a full-scale hull [59], in which the Reynolds number is two to three orders of magnitudes higher.

For example, in the KVLCC2 studies, Larsson et al. [59] found that flow separation is reduced at full scale and that the longitudinal vortex becomes much weaker compared with the model-scale results. Duvigneau et al. [37] quantitatively compared the optimization results obtained at both model and full scales. They concluded that the optimized shapes at model and full scales had similar trends; however, the changes in drag and wake distortion are quantitatively different. For example, the drag penalty in wake-only optimization was 3% at model scale but became 13% when evaluated at full scale.

It is expected that the simulation and optimization results presented in this paper will quantitatively differ at full scale, for example based on how much the existence of propellers impacts the optimization results. For practical ship designs, we recommended running hull shape optimizations at full scale. However, in this paper we use a model-scale hull, because the experimental data are available for validating our CFD solver. This does not detract from one of the main messages of this paper: that it is necessary to simultaneously consider drag and wake distortion in hull shape optimization. The same proposed methodology could be applied to the full-scale hull.

4 Conclusions

In this paper, we have conducted hydrodynamic design optimization for a model-scale JBC hull, simultaneously considering drag and wake distortion. We used a gradient-based optimization framework coupled with an efficient discrete adjoint solver to compute derivatives. We used 32 design variables to parameterize the complex hull surface, which allowed us to have great freedom for geometric modification. In addition, we imposed geometric constraints (volume, thickness, and curvature) to ensure that the final design would be practical. We used a weighted objective function that included drag and wake distortion, and constructed a Pareto front by running five optimizations with different weights. Moreover, we considered both towed and self-propulsion configurations and evaluated the impact on the optimized shapes of adding a propeller.

We observe that optimizing for only one objective results in a large penalty on the other objective. For example, for the drag-only case (Opt0) in the towed configuration, we obtain 4.4% reduction in C_D, but its σ_U^2 increases by 208.0%. Similarly, for the wake-only case (Opt4), σ_U^2 decreases by 90.6% but at the cost of an increase in C_D of 23.8%. In contrast, the weighted objective cases achieve a design with more balance between drag and wake distortion. For example, by setting a 2% weight on σ_U^2, the Opt1 case achieves a 3.8% drag reduction (0.6% lower than Opt0) while σ_U^2 increases by only 78.4% (129.6% less than Opt0). For the drag-and-constraint-wake case (Opt2), we obtain simultaneous improvement in C_D and σ_U^2. For example, for the Opt2 case in the self-propulsion configuration, both C_D and σ_U^2 decrease, by 1.3% and 6.1%, respectively.

When comparing the baseline and optimized shapes for the towed configuration, we observe that the Opt0 case forms a V-shaped hull to minimize drag. However, it also generates a V-shaped wake at the propeller plane, with a large velocity distortion. The Opt4 case creates a large bump near the bilge to reduce the intensity of the longitudinal bilge vortex, which smooths out the wake velocity field. However, Opt4 has a wavy hull surface, which is impractical from the structural and manufacturing points of view. Moreover, it also induces a large adverse pressure gradient near the bilge tube, which increases the pressure drag. By using a weighted objective function, we obtain a more balanced design by avoiding large velocity distortions at the propeller plane and sharp adverse pressure gradient on the hull surface.

For the self-propulsion configuration, we observe similar behavior, where a single-objective optimization induces a large penalty to the other objective. The major difference in the self-propulsion case is that the V-shaped hull has less adverse impact on wake quality. This is primarily because the propeller accelerates the flow near the bilge tube; as a result, the pressure decreases near the bilge tube and the adverse pressure gradient decreases, which eventually reduces the intensity of the longitudinal bilge vortex. Moreover, the accelerated flow results in more mixing, which further smooths out wake distortion. Another major difference in the self-propulsion configuration is that no large bump is obtained near the bilge for the Opt4 case. Again, this is primarily due to the suction effect of the propeller, which is such that there is no benefit in using a large bump to smooth the low-speed region at the bottom of the propeller annulus region.

The numerical uncertainty study shows that simulated σ_U^2 is much more sensitive to mesh size than C_D. This is because we use unstructured meshes, so changing mesh size also changes the number and distribution of sampling points for σ_U^2. The numerical uncertainties associated with mesh sizes are generally lower than the relative change (design improvement or degradation) in the optimized shapes. Therefore, the optimizations are verified to have correct trends. There is an exception for σ_U^2 in Opt2, where the numerical uncertainty is higher than the relative change. It is challenging to obtain a verified trend for the constrained distortion case because of the high numerical uncertainty in σ_U^2.

Overall, our results demonstrate that it is necessary to simultaneously consider drag and wake distortion in hull shape optimization. Moreover, the coupled gradient-based optimization and discrete adjoint framework developed herein open the door to the integration of other disciplines in hull design optimization, such as structure analyses, and to performing hydrostructural optimization.

Acknowledgments

The computations were done in the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation Grant No. ACI-1548562.

References

[1] Campana, E. F., Peri, D., Tahara, Y., and Stern, F., "Shape optimization in ship hydrodynamics using computational fluid dynamics," *Computer Methods in Applied Mechanics and Engineering*, Vol. 196, No. 1-3, 2006, pp. 634–651.

[2] Puente, R., Corral, R., and Parra, J., "Comparison between aerodynamic designs obtained by human driven and automatic procedures," *Aerospace Science and Technology*, Vol. 72, 2018, pp. 443–454.

[3] Tahara, Y., Peri, D., Campana, E. F., and Stern, F., "Computational fluid

dynamics-based multiobjective optimization of a surface combatant using a global optimization method," *Journal of Marine Science and Technology*, Vol. 13, No. 2, 2008, pp. 95–116.

[4] Kim, H., and Yang, C., "A new surface modification approach for CFD-based hull form optimization," *Journal of Hydrodynamics, Ser. B*, Vol. 22, No. 5, 2010, pp. 520–525.

[5] Filip, G., Kim, D.-H., Sahu, S., de Kat, J., and Maki, K., "Bulbous Bow Retrofit of a Containership Using an Open Source Computational Fluid Dynamics (CFD) Toolbox," *SNAME Transactions*, Vol. 122, 2014, pp. 244–262.

[6] Li, D., Wilson, P. A., Guan, Y., and Zhao, X., "An Effective Approximation Modeling Method for Ship Resistance in Multidisciplinary Ship Design Optimization," *ASME 2014 33rd International Conference on Ocean, Offshore and Arctic Engineering*, American Society of Mechanical Engineers, 2014, pp. V002T08A023–V002T08A023.

[7] de Baar, J., Roberts, S., Dwight, R., and Mallol, B., "Uncertainty quantification for a sailing yacht hull, using multi-fidelity Kriging," *Computers & Fluids*, Vol. 123, 2015, pp. 185–201.

[8] Luo, W., and Lan, L., "Design Optimization of the Lines of the Bulbous Bow of a Hull Based on Parametric Modeling and Computational Fluid Dynamics Calculation," *Mathematical and Computational Applications*, Vol. 22, No. 1, 2017, p. 4.

[9] Zhang, S., Zhang, B., Tezdogan, T., Xu, L., and Lai, Y., "Computational fluid dynamics-based hull form optimization using approximation method," *Engineering Applications of Computational Fluid Mechanics*, Vol. 12, No. 1, 2018, pp. 74–88.

[10] Peri, D., and Campana, E. F., "High-fidelity models and multiobjective global optimization algorithms in simulation-based design," *Journal of Ship Research*, Vol. 49, No. 3, 2005, pp. 159–175.

[11] Serani, A., Fasano, G., Liuzzi, G., Lucidi, S., Iemma, U., Campana, E. F., Stern, F., and Diez, M., "Ship hydrodynamic optimization by local hybridization of deterministic derivative-free global algorithms," *Applied Ocean Research*, Vol. 59, 2016, pp. 115–128.

[12] Chen, X., Diez, M., Kandasamy, M., Zhang, Z., Campana, E. F., and Stern, F., "High-fidelity global optimization of shape design by dimensionality reduction, metamodels and deterministic particle swarm," *Engineering Optimization*, Vol. 47, No. 4, 2015, pp. 473–494.

[13] Diez, M., Campana, E. F., and Stern, F., "Design-space dimensionality reduction in shape optimization by Karhunen–Loève expansion," *Computer Methods in Applied Mechanics and Engineering*, Vol. 283, 2015, pp. 1525–1544.

[14] Jameson, A., “Aerodynamic Design via Control Theory,” *Journal of Scientific Computing*, Vol. 3, No. 3, 1988, pp. 233–260.

[15] Martins, J. R. R. A., Kenway, G. K. W., and Brooks, T. R., “Multidisciplinary Design Optimization of Aircraft Configurations—Part 2: High-fidelity aerostructural optimization,” Lecture series, Von Karman Institute for Fluid Dynamics, Rode Saint Genèse, Belgium, May 2016. ISSN0377-8312.

[16] Kennedy, G. J., and Martins, J. R. R. A., “A Laminate Parametrization Technique for Discrete Ply Angle Problems with Manufacturing Constraints,” *Structural and Multidisciplinary Optimization*, Vol. 48, No. 2, 2013, pp. 379–393. doi:10.1007/s00158-013-0906-9.

[17] Yu, Y., Lyu, Z., Xu, Z., and Martins, J. R. R. A., “On the Influence of Optimization Algorithm and Starting Design on Wing Aerodynamic Shape Optimization,” *Aerospace Science and Technology*, Vol. 75, 2018, pp. 183–199. doi:10.1016/j.ast.2018.01.016.

[18] Bons, N. P., He, X., Mader, C. A., and Martins, J. R. R. A., “Multimodality in Aerodynamic Wing Design Optimization,” *AIAA Journal*, Vol. 57, No. 3, 2019, pp. 1004–1018. doi:10.2514/1.J057294.

[19] Jameson, A., Martinelli, L., and Pierce, N. A., “Optimum Aerodynamic Design Using the Navier–Stokes Equations,” *Theoretical and Computational Fluid Dynamics*, Vol. 10, No. 1–4, 1998, pp. 213–237. doi:10.1007/s001620050060.

[20] Nielsen, E. J., and Anderson, W. K., “Aerodynamic Design Optimization on Unstructured Meshes Using the Navier–Stokes Equations,” *AIAA Journal*, Vol. 37, No. 11, 1999, pp. 1411–1419.

[21] Anderson, W. K., and Venkatakrishnan, V., “Aerodynamic Design Optimization on Unstructured Grids with a Continuous Adjoint Formulation,” *Computers and Fluids*, Vol. 28, No. 4, 1999, pp. 443–480.

[22] Kenway, G. K. W., and Martins, J. R. R. A., “Multipoint High-Fidelity Aerostructural Optimization of a Transport Aircraft Configuration,” *Journal of Aircraft*, Vol. 51, No. 1, 2014, pp. 144–160. doi:10.2514/1.C032150.

[23] Lyu, Z., Kenway, G. K. W., and Martins, J. R. R. A., “Aerodynamic Shape Optimization Investigations of the Common Research Model Wing Benchmark,” *AIAA Journal*, Vol. 53, No. 4, 2015, pp. 968–985. doi:10.2514/1.J053318.

[24] Othmer, C., “Adjoint methods for car aerodynamics,” *Journal of Mathematics in Industry*, Vol. 4, No. 1, 2014, p. 6. doi:10.1186/2190-5983-4-6.

[25] He, P., Mader, C. A., Martins, J. R. R. A., and Maki, K. J., "An Aerodynamic Design Optimization Framework Using a Discrete Adjoint Approach with OpenFOAM," *Computers & Fluids*, Vol. 168, 2018, pp. 285–303. doi:10.1016/j.compfluid.2018.04.012.

[26] Garg, N., Kenway, G. K. W., Lyu, Z., Martins, J. R. R. A., and Young, Y. L., "High-fidelity Hydrodynamic Shape Optimization of a 3-D Hydrofoil," *Journal of Ship Research*, Vol. 59, No. 4, 2015, pp. 209–226. doi:10.5957/JOSR.59.4.150046.

[27] Garg, N., Kenway, G. K. W., Martins, J. R. R. A., and Young, Y. L., "High-fidelity Multipoint Hydrostructural Optimization of a 3-D Hydrofoil," *Journal of Fluids and Structures*, Vol. 71, 2017, pp. 15–39. doi:10.1016/j.jfluidstructs.2017.02.001.

[28] Wang, D., and He, L., "Adjoint aerodynamic design optimization for blades in multistage turbomachines—Part I: Methodology and verification," *Journal of Turbomachinery*, Vol. 132, No. 2, 2010, p. 021011.

[29] He, P., Martins, J. R. R. A., Mader, C. A., and Maki, K., "Aerothermal Optimization of a Ribbed U-Bend Cooling Channel Using the Adjoint Method," *International Journal of Heat and Mass Transfer*, 2019, pp. 152–172. doi:10.1016/j.ijheatmasstransfer.2019.05.075.

[30] Ragab, S., "Shape optimization in free surface potential flow using an adjoint formulation-Surface ships," *15th AIAA Computational Fluid Dynamics Conference*, 2001, p. 3042.

[31] Soto, O., Lohner, R., and Yang, C., "An adjoint-based design methodology for CFD optimization problems," *41st Aerospace Sciences Meeting and Exhibit*, 2003, p. 299.

[32] Martinelli, L., and Jameson, A., "An adjoint method for design optimization of ship hulls," *Proc. of the 9th International Conference on Numerical Ship Hydrodynamics, Ann Arbor, Michigan*, 2007.

[33] Stück, A., Kröger, J., and Rung, T., "Adjoint-based hull design for wake optimisation," *Ship Technology Research*, Vol. 58, No. 1, 2011, pp. 34–44.

[34] Stück, A., "Adjoint Navier-Stokes methods for hydrodynamic shape optimisation," Ph.D. thesis, Technische Universität Hamburg, 2012.

[35] Kröger, J., Kühl, N., and Rung, T., "Adjoint volume-of-fluid approaches for the hydrodynamic optimisation of ships," *Ship Technology Research*, Vol. 65, No. 1, 2018, pp. 47–68.

[36] He, P., Filip, G., Martins, J. R. R. A., and Maki, K. J., "Hull form hydrodynamic design using a discrete adjoint optimization method," *13th International Marine Design Conference*, Helsinki, Finland, 2018.

[37] Duvigneau, R., Visonneau, M., and Deng, G. B., "On the role played by turbulence closures in hull shape optimization at model and full scale," *Journal of marine science and technology*, Vol. 8, No. 1, 2003, pp. 11–25.

[38] Nelson, M., Temple, D., Hwang, J. T., Young, Y. L., Martins, J. R. R. A., and Collette, M., "Simultaneous Optimization of Propeller-Hull Systems to Minimize Lifetime Fuel Consumption," *Applied Ocean Research*, Vol. 43, 2013, pp. 46–52. doi:10.1016/j.apor.2013.07.004.

[39] Chen, P.-F., and Huang, C.-H., "An inverse hull design problem in optimizing the desired wake of ship," *Journal of ship Research*, Vol. 46, No. 2, 2002, pp. 138–147.

[40] NMRI, "Tokyo 2015 A Workshop on CFD in Ship Hydrodynamics," Tech. rep., National Maritime Research Institute, 2015. Http://www.t2015.nmri.go.jp.

[41] Deng, G., Leroyer, A., Guilmineau, E., Queutey, P., Visonneau, M., Wackers, J., and del Toro Llorens, A., "Verification and validation of resistance and propulsion computation," *Proceedings of Tokyo 2015 Workshop on CFD in Ship Hydrodynamics*, 2015.

[42] Abbas, N., and Kornev, N., "Computations of the Japan bulk carrier using URANS and URANS/LES methods implemented into OpenFoam toolkit," *Tokyo 2015—a Workshop on CFD in Ship Hydrodynamics Google Scholar*, 2015.

[43] Korkmaz, K. B., "CFD Predictions of Resistance and Propulsion for the Japan Bulk Carrier (JBC) with and without an Energy Saving Device, Master Thesis, Chalmers University of Technology," , 2015.

[44] Queutey, P., Guilmineau, E., Visonneau, M., Wackers, J., and Deng, G. B., "RANS and Hybrid RANS-LES simulations around the Japan Bulk Carrier of the Tokyo 2015 CFD Workshop," , 2016.

[45] Kenway, G. K., Kennedy, G. J., and Martins, J. R. R. A., "A CAD-Free Approach to High-Fidelity Aerostructural Optimization," *Proceedings of the 13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, TX, 2010. doi:10.2514/6.2010-9231.

[46] Luke, E., Collins, E., and Blades, E., "A Fast Mesh Deformation Method Using Explicit Interpolation," *Journal of Computational Physics*, Vol. 231, No. 2, 2012, pp. 586–601. doi:10.1016/j.jcp.2011.09.021.

[47] Perez, R. E., Jansen, P. W., and Martins, J. R. R. A., "pyOpt: A Python-Based Object-Oriented Framework for Nonlinear Constrained Optimization," *Structural and Multidisciplinary Optimization*, Vol. 45, No. 1, 2012, pp. 101–118. doi:10.1007/s00158-011-0666-3.

[48] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Journal of Optimization*, Vol. 12, No. 4, 2002, pp. 979–1006. doi:10.1137/S1052623499350013.

[49] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Review*, Vol. 47, No. 1, 2005, pp. 99–131. doi:10.1137/S0036144504446096.

[50] Spalart, P., and Allmaras, S., "A One-Equation Turbulence Model for Aerodynamic Flows," *30th Aerospace Sciences Meeting and Exhibit*, 1992. doi:10.2514/6.1992-439.

[51] Patankar, S. V., and Spalding, D. B., "A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows," *International Journal of Heat and Mass Transfer*, Vol. 15, No. 10, 1972, pp. 1787–1806. doi:10.1016/0017-9310(72)90054-3.

[52] Rhie, C., and Chow, W. L., "Numerical study of the turbulent flow past an airfoil with trailing edge separation," *AIAA Journal*, Vol. 21, No. 11, 1983, pp. 1525–1532. doi:10.2514/3.8284.

[53] Warming, R., and Beam, R. M., "Upwind second-order difference schemes and applications in aerodynamic flows," *AIAA Journal*, Vol. 14, No. 9, 1976, pp. 1241–1249.

[54] Curtis, A. R., Powell, M. J., and Reid, J. K., "On the estimation of sparse Jacobian matrices," *J. Inst. Math. Appl*, Vol. 13, No. 1, 1974, pp. 117–120.

[55] Gebremedhin, A. H., Manne, F., and Pothen, A., "What Color Is Your Jacobian? Graph Coloring for Computing Derivatives," *SIAM Review*, Vol. 47, No. 4, 2005, pp. 629–705. doi:10.1137/S0036144504444711.

[56] Balay, S., Gropp, W. D., McInnes, L. C., and Smith, B. F., "Efficient Management of Parallelism in Object Oriented Numerical Software Libraries," *Modern Software Tools in Scientific Computing*, edited by E. Arge, A. M. Bruaset, and H. P. Langtangen, Birkhäuser Press, 1997, pp. 163–202.

[57] Balay, S., Buschelman, K., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Smith, B. F., and Zhang, H., "PETSc Web page," , 2009. Http://www.mcs.anl.gov/petsc.

[58] Balay, S., Abhyankar, S., Adams, M. F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W. D., Kaushik, D., Knepley, M. G., May, D. A., McInnes, L. C., Mills, R. T., Munson, T., Rupp, K., Sanan, P., Smith, B. F., Zampini, S., Zhang, H., and Zhang, H., "PETSc Users Manual," Tech. Rep. ANL-95/11 - Revision 3.10, Argonne National Laboratory, 2018. URL `http://www.mcs.anl.gov/petsc`.

[59] Larsson, L., Stern, F., and Visonneau, M., *Numerical ship hydrodynamics: an assessment of the Gothenburg 2010 workshop*, Springer, 2013.

[60] Hoekstra, M., "A RANS-based analysis tool for ducted propeller systems in open water condition," *International Shipbuilding Progress 53*, Vol. 53, 2006, pp. 205–227.

[61] Wrenn, G. A., "An Indirect Method for Numerical Optimization Using the Kreisselmeier–Steinhauser Function," Tech. Rep. CR-4220, NASA Langley Research Center, Hampton, VA, 1989.

[62] Lambe, A. B., Martins, J. R. R. A., and Kennedy, G. J., "An Evaluation of Constraint Aggregation Strategies for Wing Box Mass Minimization," *Structural and Multidisciplinary Optimization*, Vol. 55, No. 1, 2017, pp. 257–277. doi:10.1007/s00158-016-1495-1.

[63] Stern, F., Wilson, R. V., Coleman, H. W., and Paterson, E. G., "Comprehensive approach to verification and validation of CFD simulations—part 1: methodology and procedures," *Journal of Fluids Engineering*, Vol. 123, No. 4, 2001, pp. 793–802.

[64] Roache, P. J., *Verification and validation in computational science and engineering*, Hermosa Pub, 1998.

[65] Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., Hazlewood, V., Lathrop, S., Lifka, D., Peterson, G. D., Roskies, R., Scott, J. R., and Wilkins-Diehr, N., "XSEDE: Accelerating Scientific Discovery," *Computing in Science & Engineering*, Vol. 16, No. 5, 2014, pp. 62–74. doi:10.1109/MCSE.2014.80.

[66] Kenway, G. K. W., Mader, C. A., He, P., and Martins, J. R. R. A., "Effective Adjoint Approaches for Computational Fluid Dynamics," *Progress in Aerospace Sciences*, 2019. doi:10.1016/j.paerosci.2019.05.002, (In press).

[67] Lee, S.-J., Kim, H.-R., Kim, W.-J., and Van, S.-H., "Wind tunnel tests on flow characteristics of the KRISO 3,600 TEU containership and 300K VLCC double-deck ship models," *Journal of Ship Research*, Vol. 47, No. 1, 2003, pp. 24–38.