Optimal mesh generation for a blade passage using deep reinforcement learning

Innyoung Kim, Sejin Kim, Donghyun You

Department of Mechanical Engineering, Pohang University of Science and Technology, 77 Cheongam-Ro, Nam-Gu, Pohang, Gyeongbuk 37673, South Korea

Abstract

A mesh generation method that can generate an optimal mesh for a blade passage at a single attempt is developed using deep reinforcement learning (DRL). Unlike the conventional methods, where meshing parameters must be specified by the user or iteratively optimized from scratch for a newly given geometry, the developed method employs DRL-based multi-condition (MC) optimization to define meshing parameters for various geometries optimally. The method involves the following steps: (1) development of a base algorithm for structured mesh generation of a blade passage; (2) formulation of an MC optimization problem to optimize meshing parameters introduced while developing the base algorithm; and (3) development of a DRL-based mesh generation algorithm by solving the MC optimization problem using DRL. As a result, the developed algorithm is able to successfully generate optimal meshes at a single trial for various blades.

Keywords: Mesh generation, Multi-condition optimization, Deep reinforcement learning, Structured mesh

*Corresponding author.
Email address: dhyou@postech.ac.kr (Donghyun You)
1. Introduction

A computational mesh is a crucial component for a numerical simulation of fluid flow. The quality of the mesh significantly affects the accuracy, stability, and computational cost of the simulation [1, 2]. However, generating a high-quality mesh typically requires considerable time and effort. Consequently, research has been conducted to develop algorithms that can automatically generate high-quality meshes [3–9].

Using the automatic mesh generation algorithm, users only need to specify meshing parameters required in the algorithms, such as the number of cells in the streamwise and the spanwise direction or the expansion and the clustering ratio of their distribution. Although the algorithm reduces significant time and effort in the mesh generation process, it has a limitation in that the quality of the generated mesh depends on the meshing parameters. Well-determined meshing parameters produce a high-quality mesh, while poorly-specified parameters can result in a low-quality mesh. Hence, users typically adjust the meshing parameters according to their simulation problems. However, such user dependency in the mesh generation process degrades the reliability of the simulation results.

Efforts have been made to optimize such parameters to alleviate the user dependency in mesh generation and generate an optimal mesh [10, 12]. Dittmer [10], Ahmad et al. [11], and Islam et al. [12] optimized meshing parameters for an airfoil, a ground vehicle, and a marine propeller, respectively. By incorporating optimization into the mesh generation, it was possible to optimally determine the meshing parameters and generate an optimal mesh.
However, due to the nature of the optimization, iterative mesh generation processes are inevitable. Even if the optimal parameters are obtained, they are only valid for a specific geometry. The entire process must be repeated from scratch for a new geometry, which is inefficient where numerous systematical simulations are needed, such as a design process. To overcome the limitations, a method that can generate an optimal mesh at a single attempt for various geometries is necessary.

Recently, Kim et al. [13] have established a concept of multi-condition (MC) optimization and developed an efficient method to solve the MC optimization problem using deep reinforcement learning (DRL). DRL-based MC optimization can obtain optimal solutions as a function of the condition by learning the correlations between conditions and optimal solutions. Hence, it is possible to obtain the optimal solutions for varying conditions without additional optimization. This is in contrast to the conventional optimization methods where optimization is performed under a specific condition and must be conducted again whenever the condition varies.

The present study aims to develop a DRL-based mesh generation method that can generate an optimal mesh for various geometries at a single attempt. The key idea is to exploit the DRL-based MC optimization method in developing a mesh generation algorithm. In the present method, meshing parameters introduced while developing the mesh generation algorithm are optimized considering various geometries using MC optimization. Then, it is possible to determine the optimal meshing parameters for varying geometries without additional optimization.

The proposed method is applied to generate structured meshes for turbo-
machinery blades. Designing high-performance blades typically requires a lot of simulations with various shapes, while generating a high-quality structured mesh is not trivial. The cells on the upper and lower periodic boundaries must be perfectly matched, with smooth resolution change while the complex curvature of the blade exists in the domain. The detailed procedure of the present method includes following steps. Firstly, a base algorithm for automatic mesh generation of turbomachinery blades is developed. Secondly, an MC optimization problem is formulated to optimize meshing parameters introduced while developing the base algorithm. Finally, a DRL-based mesh generation algorithm that can generate an optimal mesh at a single attempt is developed by solving the MC optimization problem using DRL. The capability of the developed DRL-based mesh generation algorithm is examined by generating meshes for various turbomachinery blades.

2. Background

2.1. Multi-condition optimization

The conventional single-condition (SC) optimization problem is defined as follows:

$$\max_x f(x)$$

subject to $x \in \Omega$,  \hspace{1cm} (1)

where $x$ is a decision vector, and $f$ is a real-valued objective function. $\Omega$ is the decision space which is defined as the set $\{x| g_m(x) \leq 0, m = 1, 2, \ldots, n\}$. $g_m(x)$ is a constraint function that determines the feasible region of $x$, and $n$ is the number of the constraints of $x$. The goal of a SC optimization is to find an optimal solution $x^*$ that maximizes $f$. 

4
In contrast to SC optimization, a condition vector $c$ and the condition space $\Phi$ are introduced in MC optimization. An MC optimization problem is defined as follows:

$$\max_x f(x, c)$$

subject to $x \in \Omega, c \in \Phi$, where $\Phi$ is defined as the set \{ $c$ | $h_p(c) \leq 0$, $p = 1, 2, \ldots, q$ \}. $h_p(c)$ is a constraint function that determines the feasible region of $c$, and $q$ is the number of the constraints of $c$. The goal of an MC optimization is to find optimal solutions $x^*(c)$ that maximize $f$ as a function of $c$.

2.2. Deep-reinforcement-learning-based multi-condition optimization method

DRL is a process of learning optimal behavior for a complex decision-making process [14]. At each discrete step $t$, an action $a_t \in A$ is determined according to a deterministic policy $\pi : S \rightarrow A$, for a given state $s_t \in S$. Here, $S$ and $A$ denote the state space and the action space, respectively. Through the execution of the action $a_t$, a reward $r_t(s_t, a_t) \in \mathbb{R}$ is acquired, and the next state $s_{t+1}$ is given according to a probability distribution $p(s_{t+1}|s_t, a_t) : S \times A \rightarrow \mathcal{P}(S)$. This process is repeated until the terminal step $T$. When the terminal step is reached, one episode ends. The return $R_t$ is defined as the sum of an immediate reward $r_t$ and discounted future rewards as follows:

$$R_t = \sum_{i=t}^{T} \gamma^{(i-t)} r_i(s_{i}, a_{i}),$$

where $\gamma \in [0, 1]$ is a discount factor that determines the weight between short-term and long-term rewards.

In DRL, the policy is parameterized by a non-linear neural network as $\pi_\phi$, with network parameters $\phi$. The goal of DRL is to find the optimal
policy $\pi_\phi$ that maximizes an objective function $J$ which is defined as an expectation of the return. Thus, $\pi_\phi$ can be updated by $\nabla_\phi J$ which can be calculated using the deterministic policy gradient algorithm \cite{15}, by applying the chain rule to the $J$ as follows:

$$\nabla_\phi J = \mathbb{E}_{s_t \sim p}[\nabla_\phi Q(s_t, a_t)]$$

$$= \mathbb{E}_{s_t \sim p}[\nabla_a Q(s_t, a_t)|a_t=\pi_\phi(s_t)\nabla_\phi \pi_\phi(s_t)].$$

Here, $Q(s_t, a_t)$ is the action value function which is defined as follows:

$$Q(s_t, a_t) = \mathbb{E}_{s_{t+1} \sim p}[R_t|s_t, a_t].$$

A DRL-based MC optimization method was proposed by Kim et al. \cite{13}. In the method, a single-step DRL-based optimization method \cite{16} is employed where one learning episode consists of a single step. If an action is determined from a given state, a reward is given accordingly and the episode ends without the next state. Since the future rewards in Eq. (3) do not exist, learning proceeds to maximize only the immediate reward. Thus by setting the state, the action, and the reward of DRL as $c$, $x$, and $f$, respectively, learning $x^*$ that maximizes $f$ as a function of $c$ becomes possible. As a result, $x^*(c)$, the solution of the MC optimization problem, can be efficiently obtained.

3. Deep-reinforcement-learning-based mesh generation method

In the present study, a method that can generate an optimal mesh for various geometries at a single attempt is developed. A systematic procedure of the method is described by developing a DRL-based mesh generation
algorithm for turbomachinery blades. Firstly, a base algorithm that automatically generates a structured mesh for a blade is developed (Section 3.1). Then, an MC optimization problem is formulated to determine the optimal meshing parameters in the base algorithm for various blade geometries (Section 3.2). Finally, the DRL-based mesh generation algorithm that can generate an optimal mesh for an arbitrary turbomachinery blade at a single attempt is developed by solving the MC optimization problem using DRL (Section 3.3).

3.1. Mesh generation algorithm for turbomachinery blades

In this section, a mesh generation algorithm for generating a structured mesh for a two-dimensional turbomachinery blade is developed. The algorithm generates an HOH-type mesh that adopts H-type meshes for the inlet and outlet sides and an O-type mesh near the blade, based on the geometric parameters about the blade and the meshing parameters specified by the user. The parameters and their descriptions are summarized in Table 1.

The schematic diagram of the algorithm is illustrated in Fig. 1. Firstly, the geometric parameters about the blade, consisting of the blade shape, blade spacing pitch, horizontal inlet position \(x_{\text{in}}\), and horizontal outlet position \(x_{\text{out}}\), are given to the algorithm. Then, the mesh boundary is determined based on three meshing parameters \(y_{\text{in}}, y_{\text{out}}, \) and \(\alpha_{\text{camber}}\). \(y_{\text{in}}\) and \(y_{\text{out}}\) are the positions of the inlet and outlet in the vertical direction, respectively. \(\alpha_{\text{camber}}\) determines the degree to which the curvature of the lower boundary
follows the camber line. The lower boundary \( y_{\text{low}}(x) \) is defined as follows:

\[
y_{\text{low}}(x) = \begin{cases} 
  y_{\text{le}} + y_{\text{in}} - \text{pitch}/2 & x_{\text{le}} - x_{\text{in}} \leq x < x_{\text{le}}, \\
  \alpha_{\text{camber}} \cdot y_{c}(x) + (1 - \alpha_{\text{camber}}) \cdot y_{l}(x) & x_{\text{le}} \leq x < x_{\text{te}}, \\
  y_{\text{te}} + y_{\text{out}} - \text{pitch}/2 & x_{\text{te}} \leq x < x_{\text{te}} + x_{\text{out}}.
\end{cases}
\] (6)

Here, \((x_{\text{le}}, y_{\text{le}})\) and \((x_{\text{te}}, y_{\text{te}})\) are the coordinates of the leading and the trailing edges. \( y_{c}(x) \) is a scaled camber line in the vertical direction defined as follows:

\[
y_{c}(x) = \frac{|y_{\text{le}} - y_{\text{te}}|}{|y_{\text{le}} + y_{\text{in}} - y_{\text{te}} - y_{\text{out}}|} y_{\text{camber}}(x) - \text{pitch}/2,
\] (7)

where \( y_{\text{camber}}(x) \) is the camber line of the blade. \( y_{l}(x) \) is a straight line linearly connecting two points \((x_{\text{le}}, y_{\text{le}} + y_{\text{in}} - \text{pitch}/2)\) and \((x_{\text{te}}, y_{\text{te}} + y_{\text{out}} - \text{pitch}/2)\).

The upper boundary is determined by shifting the lower boundary vertically by \( \text{pitch} \) to meet periodicity. Then, the locations of the interface between the O-type mesh and the H-type mesh at the inlet and outlet sides are linearly determined by \( \beta_{\text{in}}^o \) and \( \beta_{\text{out}}^o \) as follows:

\[
\begin{align*}
  x &= x_{\text{le}} - \beta_{\text{in}}^o x_{\text{in}}, \\
  x &= x_{\text{te}} + \beta_{\text{out}}^o x_{\text{out}}.
\end{align*}
\] (8)

Next, nodes of the O-type mesh are distributed according to \( N_t, N_n, \gamma_{\text{le}}, \gamma_{\text{te}}, \) and \( \Delta n_1 \). \( N_t \) and \( N_n \) are the numbers of nodes of the O-type mesh in tangential and normal directions to the blade surface, respectively. Along the blade surface, \( N_t \) is divided so that the numbers of nodes at the suction and pressure sides are proportional to their lengths. At each side, the nodes are clustered at the leading and the trailing edges to resolve them as shown in Fig. 2a. Equally distributed nodes are transformed using the hyperbolic tangent function. The transformed nodes are linearly scaled and distributed.
along each pressure and suction side. $\gamma_{le}$ and $\gamma_{te}$ determine the degree of clustering at the leading and the trailing edges. Then, edges are generated by extending the nodes on the blade surface to the outer boundary of the O-type mesh in an outward normal direction. After that, the nodes at the outer boundary are adjusted so that nodes at the upper and the lower boundaries should be periodically matched. Along each edge connected from the blade surface to the outer boundary of the O-type mesh, $N_n$ nodes are distributed. The nodes are clustered at the blade surface following the hyperbolic tangent function as shown in Fig. 2b. The degree of clustering is determined such that the height of the first cell at the blade surface is equal to the specified height of the first cell $\Delta n_1$.

After distributing the nodes of the O-type mesh, the elliptic mesh generation method [17, 18] is applied to adjust the node distribution. The elliptic mesh generation method used in this study is proposed by Hsu & Lee [19], which enables generating a mesh with high orthogonality near the blade surface without slope discontinuity inside the mesh domain. The method uses the Poisson equations, the elliptic-type partial differential equations, defined as follows:

$$
\frac{\partial^2 \xi}{\partial x^2} + \frac{\partial^2 \xi}{\partial y^2} = P_1(\xi, \eta),
\frac{\partial^2 \eta}{\partial x^2} + \frac{\partial^2 \eta}{\partial y^2} = P_2(\xi, \eta),
$$

(9)

where $(\xi, \eta)$ is the curvilinear coordinate such that the domain of $(x, y)$ is transformed to a rectangular domain. Boundaries of the rectangular domain are denoted as $\xi = 0$, $\xi = \xi_{max}$, $\eta = 0$, and $\eta = \eta_{max}$. The terms $P_1$ and $P_2$ are control functions that control the node distribution. By interchanging the
dependent and the independent variables, Eq. (9) is transformed as follows:

\[
\begin{align*}
A_1 \frac{\partial^2 x}{\partial \xi^2} - 2A_2 \frac{\partial^2 x}{\partial \xi \partial \eta} + A_3 \frac{\partial^2 x}{\partial \eta^2} &= -A_4^2 (P_1 \frac{\partial x}{\partial \xi} + P_2 \frac{\partial x}{\partial \eta}), \\
A_1 \frac{\partial^2 y}{\partial \xi^2} - 2A_2 \frac{\partial^2 y}{\partial \xi \partial \eta} + A_3 \frac{\partial^2 y}{\partial \eta^2} &= -A_4^2 (P_1 \frac{\partial y}{\partial \xi} + P_2 \frac{\partial y}{\partial \eta}),
\end{align*}
\]

(10)

where

\[
\begin{align*}
A_1 &= (\frac{\partial x}{\partial \eta})^2 + (\frac{\partial y}{\partial \eta})^2, \\
A_2 &= \frac{\partial x}{\partial \xi} \frac{\partial x}{\partial \eta} + \frac{\partial y}{\partial \xi} \frac{\partial y}{\partial \eta}, \\
A_3 &= (\frac{\partial x}{\partial \xi})^2 + (\frac{\partial y}{\partial \xi})^2, \\
A_4 &= \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} + \frac{\partial x}{\partial \eta} \frac{\partial y}{\partial \xi}.
\end{align*}
\]

(11)

By applying orthogonal boundary conditions, \(P_1(\xi,0), P_2(\xi,0), P_1(\xi,\eta_{max}),\) and \(P_2(\xi,\eta_{max})\) are determined by back-solving Eq. (10). Then control functions inside the domain are calculated by the interpolation based on power-law functions as follows:

\[
\begin{align*}
P_1(\xi, \eta) &= P_1(\xi,0)[1 - (\eta/\eta_{max})]^3 + P_1(\xi,\eta_{max})(\eta/\eta_{max})^3, \\
P_2(\xi, \eta) &= P_2(\xi,0)[1 - (\eta/\eta_{max})]^3 + P_2(\xi,\eta_{max})(\eta/\eta_{max})^3.
\end{align*}
\]

(12)

After generating the O-type mesh with the elliptic grid generation method, an HOH-type mesh is eventually generated by adding H-type meshes at the inlet and outlet sides.

3.2. Multi-condition optimization

3.2.1. Mesh quality

In this section, mesh quality that can quantitatively evaluate the status of the generated mesh is defined. Generally, mesh quality is evaluated by two factors, the consistency of spatial distribution of the cells and the resolution of the cells. The indicators to assess these are \textit{a priori} mesh quality and \textit{a posteriori} mesh quality, respectively, depending on whether the simulation
solution is reflected or not. *A priori* mesh quality does not reflect the simulation solution. Instead, it is defined as geometrical characteristics of the mesh, such as distortion of the cell, the size change with the adjacent cells, and cell arrangement. As *a priori* quality can be evaluated before performing the simulation, a lot of efforts have been made to properly define and optimize the quality metrics to generate meshes that can result in more accurate and stable simulations [20–24]. The present study employs *a priori* metrics to determine mesh quality.

*A priori* mesh quality should be designed to minimize the numerical error caused by the geometrical defect of the mesh; the spatial distribution of the cells should be as uniform as possible. To consider such challenges, both the determinant ratio of the Jacobian matrix $Q_J$ and skewness $Q_S$ are considered simultaneously to define mesh quality. Fig. 3 shows how to calculate $Q_J$ and $Q_S$ for a cell. $Q_J$ is calculated as follows:

$$ J_{i,j} = \left| \begin{array}{cc} \frac{\partial x}{\partial i} & \frac{\partial x}{\partial j} \\ \frac{\partial y}{\partial i} & \frac{\partial y}{\partial j} \end{array} \right| = \frac{x_{i+1,j} - x_{i-1,j}}{2} \frac{x_{i,j+1} - x_{i,j-1}}{2}, \quad (13) $$

$$ Q_J = \min(J_{i,j}, J_{i+1,j}, J_{i+1,j-1}, J_{i,j-1}) \cdot \max(J_{i,j}, J_{i+1,j}, J_{i+1,j-1}, J_{i,j-1}), \quad (14) $$

$x$ and $y$ are the coordinates of the node, and $i$ and $j$ are indices of the node as shown in Fig. 3a. First, the determinant of the Jacobian matrix $J_{i,j}$ at each node $(x_{i,j}, y_{i,j})$ is calculated. Then, $Q_J$ is calculated by dividing the minimum $J$ by the maximum $J$ among the four nodes constituting the cell. As $J_{i,j}$ is the area of the blue quadrangle, $Q_J$ represents the area change among adjacent cells. Higher $Q_J$ indicates that the area of the cell is more constant with neighboring cells resulting in a smooth resolution change.
around the cell.

\( Q_S \) is calculated as follows:

\[
Q_S = 1 - \max \left( \frac{90^\circ - \min(\theta_1, \theta_2, \theta_3, \theta_4)}{90^\circ}, \frac{\max(\theta_1, \theta_2, \theta_3, \theta_4) - 90^\circ}{90^\circ} \right).
\] (15)

\( \theta \) is the interior angle of the cell as shown in Fig. 3b. \( Q_S \) indicates how distorted the cell is. Thus, \( Q_S \) represents the uniformity in the cell arrangement with adjacent cells. Higher \( Q_S \) indicates that the shape of the cell is close to a rectangle, resulting in a more uniform alignment with neighboring cells and higher orthogonality at shared edges. Since \( Q_J \) and \( Q_S \) consider uniformity of the distribution of both the area and alignment of the cells, they are regarded as representative quality metrics among other metrics in the case of a quadrilateral mesh [25]. In fact, metrics that can consider area change among adjacent cells or distortion of a cell are significant considerations in generating a high-quality quadrilateral mesh, although the exact formulation might be different [9, 26, 27].

Mesh quality \( Q \) is defined as follows:

\[
Q = \frac{(Q_J)_{\text{min}} + (Q_J)_{\text{avg}} + (Q_S)_{\text{min}} + (Q_S)_{\text{avg}}}{4}.
\] (16)

( )\(_{\text{min}} \) and ( )\(_{\text{avg}} \) denotes the minimum and average values over all cells in the O-type mesh, respectively. Since cells in the H-type meshes are fixed in a rectangular shape, the quality remains high during the mesh generation process. On the other hand, cells in the O-type mesh where the turbine exists should be distorted to fit the blade shape, which can significantly degrade mesh quality. Accordingly, only the cells in the O-type mesh are included in evaluating mesh quality. Considering the entire cells would result in a mesh where a ratio of the H-type meshes is excessively large in the optimization.
process. Furthermore, the minimum values are considered as the simulation can be problematic due to the cell with the lowest quality, even though the qualities of the other cells are fine. The average values are introduced to reflect the overall quality distribution. Note that $(Q_J)_{\text{min}}$, $(Q_J)_{\text{avg}}$, $(Q_S)_{\text{min}}$, and $(Q_S)_{\text{avg}}$ lie between 0 and 1 to match the scale.

$Q_J$ of cells at the boundary of the O-type mesh is calculated with additional treatments. To calculate $Q_J$ of a cell, it is necessary to calculate the determinant of the Jacobian matrix at the four nodes constituting the corresponding cell. However, to calculate the determinant of the Jacobian matrix $J_{i,j}$, the coordinates of the neighboring nodes are required, which do not exist for the nodes at the boundary. Therefore, $J_{i,j}$ for such nodes are calculated as follows. For nodes at the boundaries meeting the H-type mesh, $J_{i,j}$ are calculated with the coordinates of the adjacent nodes in the H-type mesh. For nodes at the periodic boundaries, $J_{i,j}$ are calculated with the coordinates of the nodes at the opposite periodic boundary, as if the same mesh is periodically attached. The treatments not only enable the calculation of $Q_J$ for the cells at the boundary of the O-type mesh but also take into account smooth resolution changes at the interfaces between the O-type mesh and the H-type meshes and at the periodic boundaries.

3.2.2. Optimization formulation

An MC optimization problem is formulated to obtain optimal meshing parameters of the base mesh generation algorithm for various blade geometries. To parameterize the shape of a two-dimensional turbomachinery blade, the method developed by Agromayor et al. [28] is employed in the present study. The method can generate a blade profile with continuous curvature by using
shape parameters based on Non-Uniform Rational Basis Spline curves \[29\]. The method is capable of generating various turbine blades, such as axial gas turbines, supersonic impulse turbines, and axial compressors. In particular, the method can reversely extract shape parameters from the blade shape defined by coordinates of points. It was confirmed that the method can successfully re-parameterize various types of the existing blades using 22 parameters. For the convenience of notation, let a vector $\mathbf{BSP}$ denote the 22 blade shape parameters in the parameterization method. A detailed description of the parameterization method can be found in \[28\].

An MC optimization problem is defined as follows:

$$\max_{\mathbf{x}} f(\mathbf{x}, \mathbf{c})$$

subject to $\mathbf{x} \in \Omega, \mathbf{c} \in \Phi$, \hspace{1cm} (17)

where

$$f(\mathbf{x}, \mathbf{c}) = Q,$$

$$\Omega = \{(y_{in}, y_{out}, \alpha_{camber}, \beta_{in}^o, \beta_{out}^o, N_t, \gamma_{le}, \gamma_{te}) | -0.5C \leq y_{in} \leq 0.5C, \ -0.5C \leq y_{out} \leq 0.5C, \ 0 \leq \alpha_{camber} \leq 1, \ 0.1 \leq \beta_{in}^o \leq 0.9, \ 0.1 \leq \beta_{out}^o \leq 0.9, \ 100 \leq N_t \leq 1000, \ 0 \leq \gamma_{le} \leq 5, \ 0 \leq \gamma_{te} \leq 5\},$$

$$\Phi = \{((\mathbf{BSP}, \text{pitch}, x_{in}, x_{out}, N_o, \Delta n_1) | \mathbf{BSP} \in \mathbf{BSP}_\text{range}, \ 0.3C \leq \text{pitch} \leq 1.0C, \ 0 \leq x_{in} \leq C, \ 0 \leq x_{out} \leq C, \ 10000 \leq N_o \leq 50000, \ 2 \times 10^{-5}C \leq \Delta n_1 \leq 2 \times 10^{-4}C\}.\$$

Here, $C$ is the chord length of the blade, and $N_o = N_t \times N_n$ is the number of nodes of the O-type mesh. $\mathbf{x}$ includes the meshing parameters in the base mesh generation algorithm. The ranges of components in $\mathbf{x}$ are defined wide enough so that the base algorithm can generate various meshes. $\mathbf{c}$
is determined to consider both the blade geometry and the flow condition. $BSP_{\text{range}}$ and the ranges of pitch, $x_{\text{in}}$, and $x_{\text{out}}$ are set to cover a wide variety of blade geometries. $N_o$, which determines the resolution of the entire mesh, and $\Delta n_1$, which determines the resolution near the blade surface, are included in $c$ to consider the flow conditions. The range of $N_o$ is set based on the number of cells generally used in Reynolds-averaged Navier–Stokes (RANS) simulations [30–33]. The range of $\Delta n_1$ is set to satisfy $n^+ = 1$ at $10^5 \leq Re \leq 10^6$ calculated by the flat-plate boundary layer theory [34], where $n^+$ is the dimensionless wall distance and $Re$ is the Reynolds number. By solving the MC optimization problem, optimal meshing parameters that maximize mesh quality are obtained as a function of $c$.

3.3. Deep-reinforcement-learning-based mesh generation algorithm

3.3.1. Deep-reinforcement-learning-based multi-condition optimization

In this section, the MC optimization problem is solved by the DRL-based MC optimization method to develop a DRL-based mesh generation algorithm that can generate an optimal mesh for various blade geometries at a single attempt. For DRL, the actor-critic method [35] is employed. An actor $\pi_\phi$ and a critic $Q_\theta$ are functions parameterized by non-linear neural networks with network parameters $\phi$ and $\theta$, respectively. The actor network $\pi_\phi(s)$ determines an optimal action that maximizes the expectation of the return in Eq. (3) according to a given state. The critic network $Q_\theta(s,a)$ predicts the action value function in Eq. (5) depending on a state and an action. Note that as a single-step DRL is employed, the return is identical to the reward. The state $s$, the action $a$, and the reward $r$ of DRL are defined as $c$, $x$, and $Q$, respectively. All variables in $s$ and $a$ are linearly scaled to range between
Algorithm 1 summarizes the learning procedure of DRL-based MC optimization. For every episode, a state \( s \) is randomly given, the action is determined by \( a = \pi_{\phi}(s) \) and the base algorithm generates a mesh. Then, mesh quality \( Q \) of the generated mesh is evaluated, and the reward \( r \) is obtained. Based on the data of \((s, a, r)\), learning proceeds to maximize \( Q \) which is set as the reward. Thus, as the learning continues, the data are accumulated, and the base algorithm is reinforced to generate higher-quality meshes by adapting its meshing parameters. After sufficient learning, the base algorithm converges into a DRL-based mesh generation algorithm that can generate a mesh with optimal quality. Detailed descriptions of evaluating convergence and validating the optimality of the final algorithm are explained in Section 3.3.2.

The network structures and the hyperparameters are set according to the work by Kim et al. \[13\]. This setting has the advantage of avoiding local minima and finding the optimal solutions precisely for optimization problems with nonlinear characteristics. Both networks are parameterized as fully-connected networks with four hidden layers of 512, 256, 256, and 128 neurons. The standard deviation of the exploration noise \( \sigma \) is set as follows:

\[
\sigma = \begin{cases} 
1 & \text{episode } \leq 1000, \\
0.05(\cos(\frac{2\pi}{1000} \times \text{episode}) + 1) & \text{episode } > 1000.
\end{cases}
\] (18)

Large \( \sigma \) in early episodes allows for gathering a variety of data. Then, a cosine function is employed to balance exploration and exploitation by periodically changing the magnitude of the noise. For stable learning, the actor network is updated every two episodes, while the critic network is updated every
episode \textsuperscript{36}. The mini-batch size $N_b$ is set to 100.

3.3.2. Convergence and validation

Since a condition is chosen randomly from the condition space for every episode and the corresponding optimal quality is not known, it is difficult to judge the convergence and optimality of the DRL-based mesh generation algorithm. Instead, four test conditions are selected to represent the condition space, and their optimal qualities are obtained in advance. By comparing the qualities of the meshes generated by the algorithm with the optimal mesh qualities for the test conditions, convergence and optimality of the algorithm are judged.

The test conditions are defined using the existing turbomachinery blades. Four blades, the LS89 \textsuperscript{37}, the T106A \textsuperscript{38}, the SIRT \textsuperscript{39}, and the STD10 \textsuperscript{40}, which have different geometries and applications, are adopted to represent various blade geometries. The LS89 and the T106A are axial gas turbine blades for high and low pressure, respectively. The SIRT is a blade of a supersonic impulse turbine, and the STD10 is a axial compressor blade. Based
on the four blades, test conditions are designated as follows:

\[
\begin{align*}
  c_{\text{LS89}} &= (BSP_{\text{LS89}}, \text{pitch}_{\text{LS89}}, x_{\text{in}} = 0.5C, x_{\text{out}} = 0.5C, \\
  &N_o = 25000, \Delta n_1 = 1.1 \times 10^{-4}C), \\
  c_{\text{T106A}} &= (BSP_{\text{T106A}}, \text{pitch}_{\text{T106A}}, x_{\text{in}} = 0.5C, x_{\text{out}} = 0.5C, \\
  &N_o = 25000, \Delta n_1 = 1.1 \times 10^{-4}C), \\
  c_{\text{SIRT}} &= (BSP_{\text{SIRT}}, \text{pitch}_{\text{SIRT}}, x_{\text{in}} = 0.5C, x_{\text{out}} = 0.5C, \\
  &N_o = 25000, \Delta n_1 = 1.1 \times 10^{-4}C), \\
  c_{\text{STD10}} &= (BSP_{\text{STD10}}, \text{pitch}_{\text{STD10}}, x_{\text{in}} = 0.5C, x_{\text{out}} = 0.5C, \\
  &N_o = 25000, \Delta n_1 = 1.1 \times 10^{-4}C).
\end{align*}
\]

The values of \(x_{\text{in}}, x_{\text{out}}, N_o\), and \(\Delta n_1\) in the test conditions are the median of each variable range in the condition space. The optimal mesh quality for each test condition is obtained by performing SC optimization as the same procedure described in Algorithm 1 but only targeting the corresponding test condition.

Fig. 4 shows convergence and optimality of the algorithm as the MC optimization proceeds. The light blue, blue, and black lines show the results of SC optimization performed for each test condition. Although the SC optimization shows convergence at the 10000th episode, it is performed for sufficient episodes to obtain the optimal quality more precisely. The light red and red lines show the qualities of meshes generated by the algorithm trained over a defined condition space. Similar trends are observed in all test conditions. In the early episodes, the algorithm often generates low-quality meshes and large oscillations are observed. However, as learning progresses, the oscillations disappear, and the algorithm generates meshes with higher
qualities. Eventually, the algorithm converges at the 1000000th episode. All the converged mesh qualities are comparable with the optimal qualities obtained by SC optimization, confirming that the algorithm converges optimally. Since each test condition is just one condition that does not affect training, like other conditions in the condition space, it is expected that the developed algorithm is able to generate an optimal mesh for all conditions in the condition space.

4. Mesh generation for various blades

Fig. 5 shows the meshes generated by the present algorithm for $c_{LS89}$, $c_{T196A}$, $c_{SIRT}$, and $c_{STD10}$. High-quality meshes are generated in all the test conditions. The leading and the trailing edges are well resolved, and high orthogonality near the blade surface is observed. The periodic boundaries are perfectly matched, and the mesh resolution changes smoothly across the boundaries. The O-type mesh and the H-type mesh are smoothly connected. The smooth resolution changes are due to the additional treatments in calculating $Q$ at the boundary of the O-type mesh, as discussed in Section 3.2.1. In particular, the present algorithm successfully generates a high quality mesh for $c_{SIRT}$, which is difficult due to the highly cambered blade shape with small pitch.

Besides the test conditions, meshes are generated for various other conditions to identify the capability of the developed algorithm. In situations where the geometry changes frequently, such as a design process, the mesh generation is needed for the changed condition. As shown in Figs. 6, 7, 8, and 9 the present algorithm successfully generates optimal meshes for a wide
variety of conditions at a single attempt.

Fig. 6 shows meshes generated according to pitch where conditions are defined as follows:

\[ \mathbf{c} = (BSP_{T106A}, \text{pitch}, x_{in} = 0.5C, x_{out} = 0.5C, N_o = 25000, \Delta n_1 = 1.1 \times 10^{-4}C). \]  

Here, pitch varies from 0.3C to 0.9C while pitch_{T106A} = 0.799C. As pitch decreases, \( \alpha_{\text{camber}} \) that is the degree to which the curvature of the periodic boundaries follows the camber line of the blade increases. In addition, the locations of the interface between the O-type mesh and the H-type mesh are changed according to the pitch by adjusting \( \beta^o_{\text{in}} \) and \( \beta^o_{\text{out}} \) to make the distance between the blade surface and the O-type boundary as constant as possible.

Fig. 7 shows meshes generated according to \( x_{in} \) and \( x_{out} \) where conditions are defined as follows:

\[ \mathbf{c} = (BSP_{LS89}, \text{pitch}_{LS89}, x_{in}, x_{out}, N_o = 25000, \Delta n_1 = 1.1 \times 10^{-4}C). \]  

Here, \( x_{in} \) and \( x_{out} \) vary from 0.2C to 0.8C. As \( x_{in} \) and \( x_{out} \) increase, the locations of the interface between the O-type mesh and the H-type mesh move away from the blade and finally converge by adjusting \( \beta^o_{\text{in}} \) and \( \beta^o_{\text{out}} \) appropriately.

Fig. 8 shows meshes generated according to \( N_o \) where conditions are defined as follows:

\[ \mathbf{c} = (BSP_{STD10}, \text{pitch}_{STD10}, x_{in} = 0.5C, x_{out} = 0.5C, N_o, \Delta n_1 = 1.1 \times 10^{-4}C). \]  

Here, \( N_o \) varies from 10000 to 50000. According to \( N_o \), meshes with a similar topology are observed with gradually increasing overall mesh resolution.
Fig. 9 shows meshes generated by arbitrary sampled conditions. The developed algorithm successfully generates optimal meshes for various turbomachinery blades. Moreover, in practical usage, the blade shape is often represented by a set of scattered point coordinates, not the shape parameters such as $BSP$. The parameterization method proposed by Agromayor et al. [28] can extract shape parameters $BSP$ from point coordinates of the blade, as mentioned in Section 3.2.2. As the present algorithm employed the corresponding parameterization method, the algorithm can handle blade shapes represented by point coordinates.

5. Concluding remarks

A DRL-based mesh generation method has been developed to generate an optimal mesh for a blade passage at a single attempt. The developed method utilizes DRL-based MC optimization to determine optimal meshing parameters for various geometries to overcome the user dependency and inefficiency of the conventional mesh generation methods. In detail, the base algorithm that automatically generates a structured mesh has been developed. Next, the MC optimization problem has been formulated to optimize meshing parameters in the base algorithm for various geometries. Finally, the DRL-based mesh generation algorithm has been developed by solving the MC optimization problem using DRL. The capability of the developed algorithm has been confirmed to generate optimal meshes at a single trial for various turbomachinery blades.

The DRL-based mesh generation method proposed in the present study shows outstanding performance in optimal mesh generation for turboma-
chinery blades in the perspective of a priori mesh quality. As one of the future directions of the present research, integrating the developed method with simulation results to consider a posteriori mesh quality is considered. Furthermore, although the present study targets turbomachinery blades, the developed method can be applied with other automatic mesh generation algorithms targeting other applications \cite{5, 7, 9}. For another future direction, the extension of the developed method to the generation of an optimal mesh for an arbitrary geometry is also considered.

CRediT authorship contribution statement

**Innyoung Kim:** Conceptualization, Investigation, Methodology, Software, Validation, Visualization, Writing - Original Draft. **Sejin Kim:** Conceptualization, Investigation, Methodology, Software, Validation. **Donghyun You:** Conceptualization, Funding Acquisition, Supervision, Writing - Original Draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The work was supported by the National Research Foundation of Korea (NRF) under the Grant Number NRF-2021R1A2C2092146 and the Sam-
sung Research Funding Center of Samsung Electronics under Project Number SRFC-TB1703-51.
References

[1] M. Zandsalimy, C. Ollivier-Gooch, A novel approach to mesh optimization to stabilize unstructured finite volume simulations, Journal of Computational Physics 453 (2022) 110959. doi:https://doi.org/10.1016/j.jcp.2022.110959.

[2] Z. Ali, P. G. Tucker, S. Shahpar, Optimal mesh topology generation for CFD, Computer Methods in Applied Mechanics and Engineering 317 (2017) 431–457. doi:https://doi.org/10.1016/j.cma.2016.12.001.

[3] A. Milli, S. Shahpar, PADRAM: Parametric design and rapid meshing system for complex turbomachinery configurations, in: Turbo Expo: Power for Land, Sea, and Air, Vol. 44748, American Society of Mechanical Engineers, 2012, pp. 2135–2148. doi:https://doi.org/10.1115/GT2012-69030.

[4] R. Zagitov, A. Dushko, Y. N. Shmotin, Automatic three dimensional grid generation in turbo machine blade passages, in: Turbo Expo: Power for Land, Sea, and Air, Vol. 45615, American Society of Mechanical Engineers, 2014, p. V02BT39A044. doi:https://doi.org/10.1115/GT2014-27127.

[5] M. Costenoble, J. Baeder, Y. S. Jung, Automated mesh generation and solution analysis of arbitrary airfoil geometries, Journal of Aircraft (2022) 1–16 doi:https://doi.org/10.2514/1.C036574.

[6] E. Marchandise, C. Geuzaine, J.-F. Remacle, Cardiovascular and lung mesh generation based on centerlines, International Journal for Nu-
[7] F. Lu, L. Qi, X. Jiang, G. Liu, Y. Liu, B. Chen, Y. Pang, X. Hu, NNW-GridStar: interactive structured mesh generation software for aircrafts, Advances in Engineering Software 145 (2020) 102803. doi:https://doi.org/10.1016/j.advengsoft.2020.102803.

[8] Y. Zhang, Y. Jia, 2D automatic body-fitted structured mesh generation using advancing extraction method, Journal of Computational Physics 353 (2018) 316–335. doi:https://doi.org/10.1016/j.jcp.2017.10.018.

[9] J. Zhu, O. Zienkiewicz, E. Hinton, J. Wu, A new approach to the development of automatic quadrilateral mesh generation, International Journal for Numerical Methods in Engineering 32 (4) (1991) 849–866. doi:https://doi.org/10.1002/nme.1620320411.

[10] J. P. Dittmer, C. G. Jensen, M. Gottschalk, T. Almy, Mesh optimization using a genetic algorithm to control mesh creation parameters, Computer-Aided Design and Applications 3 (6) (2006) 731–740. doi:https://doi.org/10.1080/16864360.2006.10738426.

[11] N. E. Ahmad, E. Abo-Serie, A. Gaylard, Mesh optimization for ground vehicle aerodynamics, CFD Letters 2 (1) (2010) 54–65.

[12] M. Islam, F. Jahra, M. Doucet, Optimization of RANS solver simulation setup for propeller open water performance prediction, in: International Conference on Offshore Mechanics and Arctic Engineering, Vol. 5648, 25
[13] S. Kim, I. Kim, D. You, Multi-condition multi-objective optimization using deep reinforcement learning, Journal of Computational Physics 462 (2022) 111263. doi:https://doi.org/10.1016/j.jcp.2022.111263

[14] R. S. Sutton, A. G. Barto, Reinforcement Learning: An Introduction, MIT Press, 2018.

[15] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller, Deterministic policy gradient algorithms, in: International Conference on Machine Learning, PMLR, 2014, pp. 387–395.

[16] J. Viquerat, J. Rabault, A. Kuhnle, H. Ghraieb, A. Larcher, E. Hachem, Direct shape optimization through deep reinforcement learning, Journal of Computational Physics 428 (2021) 110080. doi:https://doi.org/10.1016/j.jcp.2020.110080.

[17] J. F. Thompson, Elliptic grid generation, Applied Mathematics and Computation 10 (1982) 79–105. doi:https://doi.org/10.1016/0096-3003(82)90188-6

[18] J. L. Steger, R. L. Sorenson, Automatic mesh-point clustering near a boundary in grid generation with elliptic partial differential equations, Journal of Computational Physics 33. doi:10.1016/0021-9991(79)90165-7

[19] K. Hsu, S. L. Lee, A numerical technique for two-dimensional grid generation with grid control at all of the boundaries, Journal of Computa-
[20] Y. Kallinderis, C. Kontzialis, A priori mesh quality estimation via direct relation between truncation error and mesh distortion, Journal of Computational Physics 228 (3) (2009) 881–902. doi:https://doi.org/10.1016/j.jcp.2008.10.023

[21] Y. Kallinderis, S. Fotia, A priori mesh quality metrics for three-dimensional hybrid grids, Journal of Computational Physics 280 (2015) 465–488. doi:https://doi.org/10.1016/j.jcp.2014.09.036

[22] R. V. Garimella, M. J. Shashkov, P. M. Knupp, Triangular and quadrilateral surface mesh quality optimization using local parametrization, Computer Methods in Applied Mechanics and Engineering 193 (9-11) (2004) 913–928. doi:https://doi.org/10.1016/j.cma.2003.08.004

[23] S. Fotia, Y. Kallinderis, Quality index and improvement of the interfaces of general hybrid grids, Procedia Engineering 82 (2014) 416–427. doi:https://doi.org/10.1016/j.proeng.2014.10.401

[24] W. Lowrie, V. Lukin, U. Shumlak, A priori mesh quality metric error analysis applied to a high-order finite element method, Journal of Computational Physics 230 (14) (2011) 5564–5586. doi:https://doi.org/10.1016/j.jcp.2011.03.036

[25] X. Gao, J. Huang, K. Xu, Z. Pan, Z. Deng, G. Chen, Evaluating hex-mesh quality metrics via correlation analysis, Computer Graphics Forum 36 (5) (2017) 105–116. doi:https://doi.org/10.1111/cgf.13249
[26] Y. Zhang, C. Bajaj, Adaptive and quality quadrilateral/hexahedral meshing from volumetric data, Computer Methods in Applied Mechanics and Engineering 195 (9-12) (2006) 942–960. doi:https://doi.org/10.1016/j.cma.2005.02.016.

[27] P. M. Knupp, Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part I—a framework for surface mesh optimization, International Journal for Numerical Methods in Engineering 48 (3) (2000) 401–420. doi:https://doi.org/10.1002/(SICI)1097-0207(20000530)48:3<401::AID-NME880>3.0.CO;2-D.

[28] R. Agromayor, N. Anand, J.-D. Müller, M. Pini, L. O. Nord, A unified geometry parametrization method for turbomachinery blades, Computer-Aided Design 133 (2021) 102987. doi:https://doi.org/10.1016/j.cad.2020.102987.

[29] L. Piegl, On NURBS: A survey, IEEE Computer Graphics and Applications 11 (01) (1991) 55–71. doi:10.1109/38.67702.

[30] H. Li, L. Song, Y. Li, Z. Feng, 2D viscous aerodynamic shape design optimization for turbine blades based on adjoint method, Journal of Turbomachinery 133 (3). doi:10.1115/1.4001234.

[31] S. N. R. Abadi, A. Ahmadpour, S. Abadi, J. P. Meyer, CFD-based shape optimization of steam turbine blade cascade in transonic two phase flows, Applied Thermal Engineering 112 (2017) 1575–1589. doi:https://doi.org/10.1016/j.applthermaleng.2016.10.058.
[32] V. Marciniak, E. Kügeler, M. Franke, Predicting transition on low-pressure turbine profiles, in: V European Conference on Computational Fluid Dynamics ECCOMAS CFD, Vol. 2010, 2010.

[33] V. Michelassi, J. Wissink, W. Rodi, Analysis of DNS and LES of flow in a low pressure turbine cascade with incoming wakes and comparison with experiments, Flow, Turbulence and Combustion 69 (3) (2002) 295–329. doi:https://doi.org/10.1023/A:10273343033200

[34] H. Schlichting, J. Kestin, Boundary layer theory, Vol. 121, Springer, 1961.

[35] V. R. Konda, J. N. Tsitsiklis, Actor-critic algorithms, in: Advances in Neural Information Processing Systems, 2000, pp. 1008–1014.

[36] S. Fujimoto, H. van Hoof, D. Meger, Addressing function approximation error in actor-critic methods, in: Proceedings of the 35th International Conference on Machine Learning, Vol. 80, PMLR, 2018, pp. 1587–1596.

[37] T. Arts, M. Lambertderouvroit, A. Rutherford, Aero-thermal investigation of a highly loaded transonic linear turbine guide vane cascade. A test case for inviscid and viscous flow computations, NASA STI/Recon Technical Report N 91 (1990) 23437.

[38] P. Stadtmüller, L. Fottner, A test case for the numerical investigation of wake passing effects on a highly loaded LP turbine cascade blade, in: Turbo Expo: Power for Land, Sea, and Air, Vol. 78507, American Society of Mechanical Engineers, 2001, p. V001T03A015. doi:https://doi.org/10.1115/2001-GT-0311
[39] N. Anand, A. Rubino, P. Colonna, M. Pini, Adjoint-based aeroelastic design optimization using a harmonic balance method, in: Turbo Expo: Power for Land, Sea, and Air, Vol. 84089, American Society of Mechanical Engineers, 2020, p. V02CT35A054. doi:https://doi.org/10.1115/GT2020-16208

[40] T. Fransson, J. Verdon, Panel discussion on standard configurations for unsteady flow through vibrating axial-flow turbomachine-cascades, in: Unsteady Aerodynamics, Aeroacoustics, and Aeroelasticity of Turbomachines and Propellers, Springer, 1993, pp. 859–889. doi:https://doi.org/10.1007/978-1-4613-9341-2_44.
List of Algorithms

1  DRL-based mesh generation method  

32
Algorithm 1: DRL-based mesh generation method

initialize actor network $\pi_\phi$ and critic network $Q_\theta$ with random parameters $\phi, \theta$;

initialize episode and replay buffer $B$;

repeat

episode ← episode+1;

randomly sample $c = (BSP, pitch, x_{in}, x_{out}, N_0, \Delta n_1)$ from $\Phi$;

receive state $s$ by scaling $c$;

select action with exploration noise: $a \leftarrow \text{clip}(\pi_\phi(s) + \epsilon, -1, 1), \epsilon \sim N(0, \sigma^2)$;

determine $x = (y_{in}, y_{out}, \alpha_{camber}, \beta_{in}^o, \beta_{out}^o, N_t, \gamma_{le}, \gamma_{te})$ by rescaling $a$;

generate mesh and evaluate $Q$;

receive reward: $r \leftarrow Q$;

store data of $(s, a, r)$ in $B$;

sample mini-batch of $N_b$ data from $B$;

update $\theta$ with the loss $N_b^{-1} \sum (r - Q_\theta(s, a))^2$;

if episode mod 2 then

update $\phi$ by the deterministic policy gradient

$N_b^{-1} \sum \nabla_a Q_\theta(s, a)|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$;

end

until convergence;
List of Tables

1 Parameters required for the base mesh generation algorithm and their descriptions. ........................................ 34
Table 1: Parameters required for the base mesh generation algorithm and their descriptions.

| Parameters | Descriptions |
|------------|--------------|
| **Geometric parameters** | |
| Blade shape | Set of scattered point coordinates of the blade |
| pitch | Blade spacing |
| $x_{in}$ | Inlet position in the horizontal direction |
| $x_{out}$ | Outlet position in the horizontal direction |
| **Meshing parameters** | |
| $y_{in}$ | Inlet position in the vertical direction |
| $y_{out}$ | Outlet position in the vertical direction |
| $\alpha_{camber}$ | Degree of the curvature of the lower boundary following the camber line |
| $\beta^o_{in}$ | HOH interface position at the inlet normalized by $x_{in}$ |
| $\beta^o_{out}$ | HOH interface position at the outlet normalized by $x_{out}$ |
| $N_t$ | Number of nodes of the O-type mesh in the tangential direction to the blade surface |
| $N_n$ | Number of nodes of the O-type mesh in the normal direction to the blade surface |
| $\gamma_{le}$ | Degree of clustering of nodes at the leading edge |
| $\gamma_{te}$ | Degree of clustering of nodes at the trailing edge |
| $\Delta n_1$ | First cell height in the normal direction to the blade surface |
List of Figures

1  Schematic diagram of the base mesh generation algorithm that generates a structured mesh of a two-dimensional turbomachinery blade. ........................................... 38

2  Node clustering using the hyperbolic tangent function. Equally distributed points on the x-axis are transformed using the hyperbolic tangent function and scaled linearly to the red line. (a) Node clustering at the leading and the trailing edges using $\gamma_{le}$ and $\gamma_{te}$. (b) Node clustering to the blade surface using $\Delta n_1$. 39

3  Mesh quality metrics. (a) Determinant ratio of the Jacobian matrix $Q_J$. (b) Skewness $Q_S$. ................................. 40
Convergence and optimality of the mesh generation algorithm based on deep reinforcement learning (DRL). Mesh quality as a function of the number of episodes is depicted for each test condition ($c_{LS89}$, $c_{T106A}$, $c_{SIRT}$, and $c_{STD10}$). The light red lines and the red lines indicate the qualities of meshes generated by the algorithm during multi-condition (MC) optimization. The light red lines indicate the mesh qualities evaluated every 100th episode, and the red lines represent the average mesh qualities of 1000 episodes centered on the corresponding episode. Likewise, the light blue lines and the blue lines indicate the mesh qualities evaluated during single-condition (SC) optimization. The light blue lines indicate the mesh qualities evaluated every 100th episode, and the blue lines represent the average mesh qualities of 1000 episodes centered on the corresponding episode. Note that the light blue lines and the blue lines overlap. The black lines indicate the maximum value within the accumulated data during SC optimization.

Meshes generated by the DRL-based mesh generation algorithm for test conditions ($c_{LS89}$, $c_{T106A}$, $c_{SIRT}$, and $c_{STD10}$). Each section is composed of a full view of the mesh and magnified views around the leading and the trailing edges. In each full view, the meshes are repeatedly attached to the periodic boundaries.

Meshes generated by the DRL-based mesh generation algorithm according to pitch.
7  Meshes generated by the DRL-based mesh generation according to $x_{in}$ and $x_{out}$. .......................................................... 44
8  Meshes generated by the DRL-based mesh generation according to $N_o$. .......................................................... 45
9  Meshes generated by the DRL-based mesh generation for various blades. .......................................................... 46
Figure 1: Schematic diagram of the base mesh generation algorithm that generates a structured mesh of a two-dimensional turbomachinery blade.
Figure 2: Node clustering using the hyperbolic tangent function. Equally distributed points on the x-axis are transformed using the hyperbolic tangent function and scaled linearly to the red line. (a) Node clustering at the leading and the trailing edges using $\gamma_{le}$ and $\gamma_{te}$. (b) Node clustering to the blade surface using $\Delta n_1$. 

(a) $y = \tanh(x)$

(b) $y = \tanh(x)$
\[ Q_{i,j} = \frac{\frac{\partial x}{\partial i} \frac{\partial y}{\partial j} - \frac{\partial x}{\partial j} \frac{\partial y}{\partial i}}{2} = \frac{x_{i+1,j} - x_{i-1,j} + x_{i,j+1} - x_{i,j-1}}{2} \]

Area of the blue quadrangle

\[ Q_J = \min(Q_{i,j}, Q_{i+1,j}, Q_{i+1,j-1}, Q_{i,j-1}) / \max(Q_{i,j}, Q_{i+1,j}, Q_{i+1,j-1}, Q_{i,j-1}) \]

\[ Q_S = 1 - \max\left(\frac{90^\circ - \min(\theta_1, \theta_2, \theta_3, \theta_4)}{90^\circ}, \frac{\max(\theta_1, \theta_2, \theta_3, \theta_4) - 90^\circ}{90^\circ}\right) \]

(a) Determinant ratio of the Jacobian matrix $Q_J$. (b) Skewness $Q_S$.

Figure 3: Mesh quality metrics. (a) Determinant ratio of the Jacobian matrix $Q_J$. (b) Skewness $Q_S$. 
Figure 4: Convergence and optimality of the mesh generation algorithm based on deep reinforcement learning (DRL). Mesh quality as a function of the number of episodes is depicted for each test condition ($c_{\text{LS89}}$, $c_{\text{T106A}}$, $c_{\text{SIRT}}$, and $c_{\text{STD10}}$). The light red lines and the red lines indicate the qualities of meshes generated by the algorithm during multi-condition (MC) optimization. The light red lines indicate the mesh qualities evaluated every 100th episode, and the red lines represent the average mesh qualities of 1000 episodes centered on the corresponding episode. Likewise, the light blue lines and the blue lines indicate the mesh qualities evaluated during single-condition (SC) optimization. The light blue lines indicate the mesh qualities evaluated every 100th episode, and the blue lines represent the average mesh qualities of 1000 episodes centered on the corresponding episode. Note that the light blue lines and the blue lines overlap. The black lines indicate the maximum value within the accumulated data during SC optimization.
Figure 5: Meshes generated by the DRL-based mesh generation algorithm for test conditions \( (c_{\text{LS89}}, c_{\text{T106A}}, c_{\text{SIRT}}, \text{and } c_{\text{STD10}}) \). Each section is composed of a full view of the mesh and magnified views around the leading and the trailing edges. In each full view, the meshes are repeatedly attached to the periodic boundaries.
Figure 6: Meshes generated by the DRL-based mesh generation algorithm according to pitch.

\begin{align*}
\text{pitch} &= 0.3C \\
\text{pitch} &= 0.5C \\
\text{pitch} &= 0.7C \\
\text{pitch} &= 0.9C
\end{align*}
Figure 7: Meshes generated by the DRL-based mesh generation according to $x_{in}$ and $x_{out}$. 

$x_{in} = 0.2C$
$x_{out} = 0.2C$

$x_{in} = 0.4C$
$x_{out} = 0.4C$

$x_{in} = 0.6C$
$x_{out} = 0.6C$

$x_{in} = 0.8C$
$x_{out} = 0.8C$
Figure 8: Meshes generated by the DRL-based mesh generation according to $N_o$. 

$N_o = 10000$  
$N_o = 23000$  
$N_o = 36000$  
$N_o = 50000$
Figure 9: Meshes generated by the DRL-based mesh generation for various blades.