An Efficient Mesh Deformation Approach Based on Radial Basis Functions in an Unstructured Flow Solver

Xingyuan Su\textsuperscript{1} and Chunhua Sheng\textsuperscript{2}

\textit{University of Toledo, Toledo, OH, 43606}

Christian Allen\textsuperscript{3}

\textit{University of Bristol, Bristol, U.K.}

An efficient parallelized mesh deformation algorithm has been implemented in an unstructured CFD solver based on the Radial Basis Functions (RBFs) method. This method requires no grid connectivity information, which allows the method to be easily implemented in unstructured CFD solvers in parallel fashion. To reduce the computational cost of the RBF method, a Greedy Algorithm is employed to vastly reduce the number of surface basis points. The test cases show that this method can effectively handle vary large mesh deformations of three-dimensional complex geometries due to their translations, rotations and deformations while maintaining a good mesh quality.

I. Introduction

An accurate prediction of aeroelastic problems generally requires the coupling of Computational Fluid Dynamics (CFD) tools with Computational Structural Dynamics (CSD) tools. This has recently become popular and practical as the computing power has consistently advanced. In the rotorcraft industry, helicopter rotor blades are usually long which inherently endure structural deformations. This will cause a moving boundary problem in the computational domain. In addition, the motion of helicopter rotor blades can be complicated, which in large eliminates the possibility of utilizing traditional mesh moving methods where all mesh points are moved like a rigid body around the blade. Therefore, an efficient and robust mesh deformation method is essential for accurate aeroelastic analysis, particularly for unsteady rotor computations. Several mesh deformation methods have been reported in the literature, which can be broadly classified into two categories: algebraic method and partial differential equation (PDE) method. Examples of algebraic method are that of transfinite interpolation (TFI)\textsuperscript{1}\textsuperscript{-3}, tension spring analogy\textsuperscript{4}, and torsion spring analogy\textsuperscript{5}. A good example of PDE method is linear elasticity analogy\textsuperscript{6}\textsuperscript{-7}. However, the performances of these methods in terms of robustness and efficiency are quite different, depending on the mesh type and the amplitude of deformation.

The TFI\textsuperscript{1}\textsuperscript{-3} method, generally used in the structured mesh, interpolates the displacements of mesh points on boundary along mesh lines to points in the interior computational domain. This method is efficient but not appropriate for unstructured meshes. For tension spring analogy\textsuperscript{5}, mesh points are viewed as connected with their neighbors by springs, whose stiffness is proportional to the reciprocal of the length. Clearly, even though this method is mathematically simple, it in nature requires mesh connectivity information. Thus, the mesh deformation involves solving a system of equations including all mesh points, which makes it too expensive for large-scale problems. This method also tends to provide a bad mesh quality for large deformations. To improve this vital disadvantage, torsion spring analogy\textsuperscript{5} attaches torsion springs to each vertex to prevent appearance of negative volumes, but at the cost of computational times. For the PDE method, mesh points are considered to follow the linear elasticity equation of solid mechanics, and the deformation domain is thus viewed as an elastic body. In theory, this method is robust, as it links the stiffness of a region to its volume, which sets the boundary layer as a solid body. However in practice, like tension spring analogy, this method requires the mesh connectivity and involves solving a system of equations of all mesh points, making the method expensive for large-scale problems as well\textsuperscript{8}.

\textsuperscript{1}Ph.D. Student, Department of Mechanical Engineering, University of Toledo, AIAA Student Member
\textsuperscript{2}Associate Professor, Department of Mechanical Engineering, University of Toledo, AIAA Associate Fellow
\textsuperscript{3}Professor, Department of Aerospace Engineering, University of Bristol, AIAA Senior Member

\textcopyright{} 2011 by the American Institute of Aeronautics and Astronautics
Recently, several researchers have paid their attentions to a novel mesh deformation method based on Radial Basis Functions (RBFs)\textsuperscript{8-11}. The RBF methods are well established for interpolating scattered data. The main advantages of the RBF interpolation are its mathematical elegance and no requirements of connectivity between scattered points. That is to say, the value of a point is obtained only based on its space position not depending on mesh connectivity. In the context of mesh deformations, a mesh point can be individually moved with the advantage of RBF interpolation, which avoids solving a system of equations involving all mesh points and high computational costs. Therefore, the RBF method can be an ideal candidate for mesh deformations, particularly for unstructured meshes.

The goal of this study is to develop an efficient mesh deformation module suitable for an unstructured mesh CFD solver U\textsuperscript{2}NCLE\textsuperscript{12}. This module is a key component of an aeroelastic analytical tool being developed for rotorcraft aeromechanical problems. Therefore, it remains interesting to see how RBF interpolation handles the large motion and deformation of a realistic and complex helicopter rotor blade, and to evaluate its efficiency and robustness for large-scale viscous unstructured meshes at such a level of complexity. In the following, the surface deformation approach in a rotating coordinate system will be introduced first, followed by the volume deformation using the RBF interpolation. Then, a greedy algorithm will be detailed to reduce the usage of surface mesh points, and followed by discussions on the parallelization of the RBF method. Finally, several test cases are presented to validate the performance of the method for unstructured mesh motion and deformation.

II. Methodology

A. Surface Mesh Deformation

![Fig 1. A diagram of an aerodynamic model of CFD/CSD coupling](image)

Before performing the volume mesh deformation in the computational domain, a CFD solver should know mesh deformation information, for example from a CSD solver, about the surface (boundary) mesh movement. The ultimate goal of this study is to develop an aeroelastics analytical tool by coupling the current unsteady unstructured Reynolds-averaged Navier-Stokes solver, U\textsuperscript{2}NCLE\textsuperscript{12}, with a finite element based tool for nonlinear flexible multibody systems, DYMORE\textsuperscript{13}, to predict the rotorcraft aeromechanical performance. Therefore, the current surface mesh deformation is developed based on the output data format of the DYMORE code. DYMORE assumes each rotor blade as a beam, as shown in Fig 1, which is typical for rotorcraft industries. With this simplification, the surface deformation is greatly simplified, and can be obtained by three translational deformations (\(\Delta x, \Delta y, \Delta z\)) and three rotational deformations (\(\Delta \phi, \Delta \theta, \Delta \psi\)) on air station points consisting of lifting lines. These deformations are written as a function of azimuthal angles and radial stations. The blade motions are interpolated using a bilinear interpolation scheme from the CSD lifting line to the CFD radial station. The rotational deformation is applied to the initial undeformed mesh following the x-y-z sequence of the Euler angle rotation around the reference point as follows\textsuperscript{14}:
\[
\begin{bmatrix}
x \\
y \\
z \\
\end{bmatrix}_r = L_f \begin{bmatrix}
x \\
y \\
z \\
\end{bmatrix}_f - \begin{bmatrix}
x \\
y \\
z \\
\end{bmatrix}_\text{ref}
\]

(1)

where

\[
L_f = L_3(\Delta \psi)L_2(\Delta \theta) L_1(\Delta \phi)
\]

\[
= \begin{bmatrix}
CS \cdot CT & -SS \cdot CF + ST \cdot CS \cdot SF & SS \cdot SF + ST \cdot CS \cdot CF \\
SS \cdot CT & CS \cdot CF + SS \cdot ST \cdot SF & -CS \cdot SF + SS \cdot ST \cdot CF \\
-ST & CT \cdot SF & CT \cdot CF
\end{bmatrix}
\]

and

\[
CS = \cos(\Delta \psi), \quad CT = \cos(\Delta \theta), \quad CF = \cos(\Delta \phi)
\]

\[
SS = \sin(\Delta \psi), \quad ST = \sin(\Delta \theta), \quad SF = \sin(\Delta \phi)
\]

The coordinates \((x,y,z)\) with index \(in\) refers to the initial mesh coordinates placed at zero degrees azimuth without pre-cone, elastic deformation, flapping and pitch control input, but with built-in twist angle. The point with index \(ref\) is the reference point, where the deformations are obtained after the interpolation. The final mesh at the desired azimuthal angle is then obtained after applying the linear transformation, coning angle and rotation to the azimuthal location as follows:

\[
\begin{bmatrix}
x \\
y \\
z \\
\end{bmatrix}_\text{ref} = L_{th} \begin{bmatrix}
x \\
y \\
z \\
\end{bmatrix}_f + \begin{bmatrix}
\Delta x \\
\Delta y \\
\Delta z \\
\end{bmatrix}
\]

(2)

where

\[
L_{th} = L_3(-\psi)L_2(\theta)
\]

\[
= \begin{bmatrix}
\cos(\psi)\cos(\theta) & -\sin(\psi) & -\cos(\psi)\sin(\theta) \\
\sin(\psi)\cos(\theta) & \cos(\psi) & \sin(\psi)\sin(\theta) \\
\sin(\theta) & 0 & \cos(\theta)
\end{bmatrix}
\]

and \(\psi\) is the azimuthal angle (positive, counter-clockwise from top view), and \(\theta\) is the flap angle (positive, flap up).

It is worth noting that the above deformations are applied to the initial undeformed mesh. Since DYMORE only outputs the blade deformation at a finite number of air station points in the initial position, a linear interpolation technique is employed to obtain the deformed coordinates of all CFD blade surface points.

**B. Volume Deformation**

After deforming the surface mesh, the volume mesh deformation is then performed. Since the RBF based interpolation is very effective to approximate scattered data, which has been utilized in the computer graphics\(^{15}\) and the numerical solution of PDE\(^{16}\), the volume mesh deformation will take this advantage. The basic form of RBFs approximation can be expressed as:

\[
f(x) = \sum_{i=1}^{n} \lambda_i \phi(\|x - x_i\|)
\]

(3)

where \(f(x)\) is the function to be interpolated at point \(x\), \(x_i\) is the base point whose function value is known, \(n\) is the total number of base points, \(\lambda_i\) is the weighting coefficient for the base point \(i\), \(\|\|\) denotes the Euclidean distance between two points, and \(\phi()\) is the radial basis function. In the case of mesh deformation, \(f(x)\) is the mesh displacement. The goal here is to interpolate the mesh displacement for every volume point based on the motion or deformation of the surface points, which are calculated using the relations (1) and (2).

Consider a three-dimensional mesh problem, where three interpolation functions are needed, with one for each coordinate. Generally, the interpolated value of function \(f(x)\) at base point \(i\) should be equal to the known value, since fluid-structure problems typically require no crossing on the fluid-structure interface:

\[
f(x_i) = d(x_i)
\]

(4)

where \(d\) is the known mesh displacement.

If the above equation (4) is written for every surface mesh point, the following system may be obtained:
\[ A \lambda_x = d_x, \]
\[ A \lambda_y = d_y, \]
\[ A \lambda_z = d_z, \] (5)

where

\[ d_x = [\Delta x_i, \Delta x_j, \ldots, \Delta x_n]^T, \]
\[ d_y = [\Delta y_i, \Delta y_j, \ldots, \Delta y_n]^T, \]
\[ d_z = [\Delta z_i, \Delta z_j, \ldots, \Delta z_n]^T, \]
\[ A = \begin{bmatrix} \phi_{i_1} & \phi_{i_2} & \cdots & \phi_{i_n} \\ \phi_{j_1} & \phi_{j_2} & \cdots & \phi_{j_n} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{n_1} & \phi_{n_2} & \cdots & \phi_{n_n} \end{bmatrix}, \]

with \( \phi_{ij} = \phi(\text{dis}_{ij} / r) \), where \( \text{dis}_{ij} \) is the distance between surface point \( i \) and \( j \). In equation (5), \( A \) is a distance matrix and \( d_x, d_y, d_z \) are mesh displacement vectors of surface mesh points in \( x, y, z \) directions, respectively.

There are many choices of the radial basis equations. The popular Wendland’s radial basis functions with the compact support are listed in Table 1. A very elegant property of compact support RBF functions is that they automatically ensure that \( A \) is strictly positive definite. However, it should be noted that some choices of the radial basis equations may cause a very ill-conditioned matrix. For the current study, the \( C_2 \) function is used. The reason is that if the supporting radius is appropriately chosen, the chance of an ill-conditioned matrix \( A \) will be greatly reduced even for large-scale problems. Two numerical methods, direct method and iteration method, can both be applied to solve the above system.

<table>
<thead>
<tr>
<th>( \xi )</th>
<th>( \phi(\xi) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0</td>
<td>( (1-\xi)^2 )</td>
</tr>
<tr>
<td>C2</td>
<td>( (1-\xi)^4(4\xi + 1) )</td>
</tr>
<tr>
<td>C4</td>
<td>( (1-\xi)^6(35\xi^2 + 18\xi + 3) )</td>
</tr>
<tr>
<td>C6</td>
<td>( (1-\xi)^8(32\xi^3 + 25\xi^2 + 8\xi + 1) )</td>
</tr>
</tbody>
</table>

\( \xi = |x|/r \) and \( r \) is the supporting radius.

**Table 1. Popular Wendland’s radial basis functions with compact support**

After computing the weighing coefficients for all base points, the displacement of the volume points can be calculated by equation (3). Again, if Eq. (3) is written for every volume point, the following system of equations can be obtained.

\[ \Delta x = B \lambda_x, \]
\[ \Delta y = B \lambda_y, \]
\[ \Delta z = B \lambda_z, \] (6)

where,

\[ \Delta x = [\Delta x_i, \Delta x_j, \ldots, \Delta x_n]^T, \]
\[ \Delta y = [\Delta y_i, \Delta y_j, \ldots, \Delta y_n]^T, \]
\[ \Delta z = [\Delta z_i, \Delta z_j, \ldots, \Delta z_n]^T, \]

and
\[ B = \begin{bmatrix} \phi_{i_1} & \phi_{i_2} & \cdots & \phi_{i_n} \\ \phi_{i_1} & \phi_{i_2} & \cdots & \phi_{i_n} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{i_1} & \phi_{i_2} & \cdots & \phi_{i_n} \end{bmatrix} \]

where \( \Delta x, \Delta y, \Delta z \) are the mesh displacement vectors of the volume points in the \( x, y, z \) direction, respectively. From Eq. (6), it shows that the solution of the volume mesh deformation is actually a matrix multiplication, which is computationally efficient. After solving the above equations (3-6) and obtaining the deformation on all volume mesh points, the mesh coordinates are updated to form a new deformed mesh for the CFD computation.

### III. Greedy Algorithm

The conventional RBF method uses all the surface points to calculate the weighting coefficients for deforming the volume mesh points, where the computational costs can be huge or prohibitive for performing large-scale viscous flow computations. In order to reduce the computational time, a much fewer number of surface points should be selected as the basis surface points without degrading the desired accuracy. This can be accomplished by a greedy algorithm as described below:

1. choose a subset of surface points to start the greedy algorithm
2. deform the surface mesh points based on the selected surface points
3. check the displacement error of every surface mesh point
4. if the largest displacement error (greedy residual) among the whole surface mesh points is smaller than a give criteria, then stop the greedy algorithm
5. else put the node with the largest error into the subset - selected surface points
6. go to step 2

The displacement error of a mesh point in step 2 is defined as follows:

\[ Err = \sqrt{Err_x^2 + Err_y^2 + Err_z^2} \tag{7} \]

where

\[ Err_x = \Delta x_{\text{accurate}} - \Delta x_{\text{greedy}} \]
\[ Err_y = \Delta y_{\text{accurate}} - \Delta y_{\text{greedy}} \]
\[ Err_z = \Delta z_{\text{accurate}} - \Delta z_{\text{greedy}} \]

Here the subscript accurate denotes the accurate displacement obtained in the surface deformation, and greedy denotes the computed displacement using the greedy algorithm. In practice, to make the greedy algorithm more meaningful and easier to control the mesh quality, the greedy algorithm stop criteria may be modified as the ratio of greedy residual to the mesh characteristic length reduced to a certain value. With the greedy algorithm, the computational cost of original RBF method can be greatly reduced, making the RBF method practical for large-scale viscous problems.

### IV. Parallelization

It is clear that the above RBF approach should be parallelized to save the computational time for the mesh deformation. Fortunately, the RBF method requires no mesh connectivity, and thus can be implemented in the parallel fashion in a straightforward way. To illustrate the approach, consider the following two-dimensional mesh where an inner box will be moved inside a larger fixed box. The computational domain has been decomposed into four blocks. It is convenient to define a data structure associated with the RBF method and attach to each block. Through this data arrangement, the information about the boundary points, the volume points, and their indices for the whole deformed domain is attached in each of the four blocks, respectively.
The computational cost of the deforming large volume meshes can be implemented in a parallel fashion, significantly reducing the computational time for deforming large-sized volume meshes.

V. Results and Discussions

A. NACA0012: Simple Mesh Movement

To validate the performance of the mesh deformation approach, a simple two-dimensional mesh movement of a NACA0012 rotor is first considered. Figure 3 shows the unstructured viscous computational mesh generated using mixed-element cells, with prisms in the boundary layers and tetrahedral cells in the inviscid region off the solid body. The blade is constructed by a NACA0012 airfoil at center of the domain, and surrounded by an interior cylinder surface to define the outer fixed boundary. The interior cylinder surface is embedded by another half cylinder as the far field boundary.

The diameter of the cylinder is about 5 times the chord length of the airfoil. The total number of mesh nodes in the deformed domain is 379,170, with 25,121 nodes on blade surface. The tests included mesh movements due to rigid body rotation and translation of the airfoil. To demonstrate the efficiency and robustness of the mesh deformation approach and its applicability to realistic unsteady aerelastic problems, all mesh movements performed here are completed in just one step without any intermediate and consecutive mesh movements. Since the value of the support of RBF method can influence the quality and robustness of the approach, the support radius in this case is chosen to be 4 times the diameter of the interior cylinder. The greedy stop criterion is set equal to $5 \times 10^{-4}$ for all cases.

Figure 4 shows the deformed mesh after one-step translational movement of the airfoil in 0.1 and 0.2 times the radius of the cylinder, respectively, where the cylinder surface is fixed. It is seen that comparing with the original mesh, the deformed meshes preserve a valid connectivity, and meshes in the boundary layer remain smooth even for such a large deformation (see zoomed views). Figure 5 illustrates the deformed meshes after a one-step rotational movement of the airfoil by $20^\circ$ and $40^\circ$ in a counter-clockwise direction around the center of the domain, respectively. Like the translation movements, the cylinder surface is fixed. Again, smoothness and valid connectivity are maintained in both cases. For the larger rotating movement, the mesh in the boundary layer preserves good orthogonality without any skewed cells even after a $40^\circ$ blade rotation. This test demonstrates a very robust feature of the current mesh deformation approach, as it eliminates the singular metrics and negative volumes of the deformed meshes. More validations combined with the U$^2$NCLE flow solver will be presented in the following Section B and C.

Figure 2. Schematic of parallel computing for RBF method
Figure 3. NACA 0012 airfoil unstructured mesh
Figure 4. One-step translational movements in 0.1 (upper) and 0.2 (lower) times the radius of the cylinder.
It has been reported that mesh deformations based on the tension spring analogy and linear elasticity analogy tend to suffer from high computational time and costs for large scale problems. In the current approach, however, the computational time to deform a large scale mesh problem is significantly reduced due to the use of the greedy algorithm together with the parallelization technique. The key to the efficiency in the current method is the huge reduction of the surface points being used for the computing the weighting coefficients, which only account to roughly 1% of the total surface points. Figures 6 and 7 show the variation of greedy residuals with a continuously surface point selection. The greedy algorithm tends to use more surface points to approximate the surface deformation for larger deformation; however, it is interesting to note that the greedy selection procedure is a little bit different for the two mesh movements above. For the translation movement, the greedy residual drops down quickly and remains stable after 100 points have been selected, see Figure 6. For the rotating motion, there exist stronger oscillations during the greedy selection procedure, as shown in Figure 7. Fortunately, this oscillation does not prevent the quick reduction of the greedy residual, which is vital to the efficiency of the RBF approach.

Table 2 shows that the statistic data and CPU times of the current method applied to the above computations. All the cases utilize 16 processors on a Linux High Performance Computing cluster. The cluster is comprised of 256 nodes using 2 Dell Powerconnect 6248 48Gbps Ethernet switches for inter-node communication. In general, a larger mesh deformation or motion will generate more greedy surface points selected than a smaller mesh deformation or motion, which increases the computational time proportionally. Although more greedy surface points may increase the interpolation accuracy of the mesh deformation, they should never be over 200-300 points at the maximum. Therefore, the key to the current method is to achieve an optimal balance between the computational costs and the deformed mesh quality.
<table>
<thead>
<tr>
<th>Case</th>
<th>CPU time (s)</th>
<th>Wall time (s)</th>
<th>Greedy points selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Translational move 0.1 cylinder radius</td>
<td>1.98</td>
<td>2.49</td>
<td>174</td>
</tr>
<tr>
<td>Translational move 0.2 cylinder radius</td>
<td>2.95</td>
<td>3.58</td>
<td>219</td>
</tr>
<tr>
<td>Counter-clockwise rotate 20°</td>
<td>4.92</td>
<td>7.03</td>
<td>175</td>
</tr>
<tr>
<td>Counter-clockwise rotate 40°</td>
<td>6.12</td>
<td>8.85</td>
<td>237</td>
</tr>
</tbody>
</table>

Table 2. Computing time and greedy points selected for simple mesh movements in one time step

Figure 6. Variation of greedy residual with greedy point selection (translation movement)

Figure 7. Variation of greedy residual with greedy point selection (rotating movement)
B. NACA0012: Rotating Pitching Movement

The goal of the present investigation is to develop an efficient mesh deformation algorithm and module in an unstructured CFD solver, which will be used for the rotorcraft aeroelastic study in the future. To this end, the mesh deformation approach is validated in the previous NACA 0012 rotor geometry\(^1\) in hover motion including a rigid cyclic pitching. The purpose here is to verify that the mesh deformation procedure presented in this study will repeat the same rotor blade motion using the sliding mesh approach\(^1\)\(^2\). If the rotor blade rotates around Z axis, the blade cyclic motion can be expressed as\(^2\)

\[
\begin{align*}
\varphi(t) &= \varphi_0 + \alpha_1 \sin(2\pi ft + \phi) + \alpha_2 \cos(2\pi ft + \phi) \\
\theta(t) &= \theta_0 + \alpha_3 \sin(2\pi ft + \phi) + \alpha_4 \cos(2\pi ft + \phi) \\
\psi(t) &= \Omega t
\end{align*}
\]

(8)

Figure 8. Deformed mesh at 90^\circ azimuthal angle position using mesh deformation method (upper) and sliding interface method (lower)
where $\phi$, $\theta$ and $\psi$ are the pitching, flapping and azimuthal angles, respectively; $\Omega$ is the angular rotating velocity; $f$ is frequency; $\phi_0$ is phase angle; $\phi_0$, $\theta_0$, $\alpha_i$ ($1 \leq i \leq 4$) are the blade motion control angles.

In the present study, the blade control angles are set as follows: $\phi = 90^0; \theta_0 = 8^0; \alpha_1 = -7.65^0$ and $\alpha_2 = -0.96^0$. The flapping motion is not considered here. $\Omega$ is chosen to rotate the rotor at a rate of one degree azimuth angle per time step. The mesh used in this case is identical to the cases shown in the previous Section, where the pre-cone angle is $0^0$ and the collective pitch angle is $10^0$. To validate the mesh deformation and its efficiency and robustness, the mesh deformation is performed together with flow solver to predict the NACA 0012 rotor in a hover flight. The blade motion and computational results are compared with those obtained by using the sliding interface method in the U*NCLE code. The inviscid flux is evaluated with a second order Roe scheme, while the time advancement is carried out with a second order temporal accuracy using the Newton's method. A series of Newton sub-iteration (3–4) are performed to ensure the temporal accuracy from time step $n$ to $n + 1$, where 6–8 Gauss-Seidel relaxations are used to converge the linear solution at each time step. The turbulence model used here is a standard Spalart-Allmaras model, and a wake model is utilized to account for the rotor wake effect.

![Graphs showing convergence histories of normalized thrust ($C_T$), hub pitching moment ($C_{MX}$), hub roll moment ($C_{MY}$) and hub torque ($C_{MZ}$) coefficients](image)

Figure 9. Comparisons of convergence histories of normalized thrust ($C_T$), hub pitching moment ($C_{MX}$), hub roll moment ($C_{MY}$) and hub torque ($C_{MZ}$) coefficients
In the present study, all unsteady computations (including the mesh deformation method and the sliding interface method) are started from a converged steady-state solution, which is obtained with a local CFL number using two Newton iterations per time step for five rotor revolutions. This solution is considered to be sufficient as an initial solution for the unsteady simulations. For the mesh deformation method, the support radius is chosen in the same way as in the previous case A. The criterion of the greedy algorithm stop used in this case, in contrast to the previous case A, is set to $5\times10^{-4}$ to guarantee the shape of the blade surface and to avoid the occurrence of negative volumes. Shown in Figure 8 is the comparison of the meshes at a $90^\circ$ azimuthal angle position using the mesh deformation method and the sliding interface method. The comparison shows the identical blade positions obtained by both methods. The mesh quality of the deformed mesh is also comparable with that obtained by the sliding interface method, particularly in the boundary layers. Clearly, the current mesh deformation approach has successfully repeated the blade motion as predicted by the sliding interface method. This conclusion can be further proved in Figure 9 by comparing the forces and moments predicted with both methods. Figure 9 shows the convergence histories of the normalized coefficients of the rotor thrust ($C_T$), hub pitching moment ($C_M$), hub roll moment ($C_{Mx}$), and hub torque coefficients ($C_{My}$) using the mesh deforming method and the sliding method. The agreements between the two approaches are excellent in general, except for a slight discrepancy on the peak thrust.

Table 3 compares the computational costs of the mesh deformation method relative to the CFD flow solver. It is certain that the number of the greedy surface points selected will have a significant impact on the performance of the mesh deformation. In the present case, the greedy surface points are 270 to 290 during each mesh deformation. The overhead of the computational cost is about 37.7% of the CFD flow solution along, which is rather economical compared to other mesh deformation methods. It is worth to mention that, even though only 270 to 290 surface mesh points (about 1.15% of the total surface points) are selected by the greedy algorithm for the current mesh deformation, the blade surface shape is well preserved, and predicted forces and moments are very comparable to that obtained by the sliding interface method.

<table>
<thead>
<tr>
<th>Method</th>
<th>CPU time (s)</th>
<th>Wall time (s)</th>
<th>Greedy points selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mesh deformation only</td>
<td>4389.08</td>
<td>6766.75</td>
<td>270–290</td>
</tr>
<tr>
<td>Mesh deformation with the solver</td>
<td>12996.26</td>
<td>17953.55</td>
<td>270–290</td>
</tr>
</tbody>
</table>

**Table 3. Computing time and greedy points selected for the NACA 0012 in five revolutions**

C. Bell 427 Main Rotor: Rotating Pitching and Flapping Movement

The previous validation has demonstrated an accurate and efficient mesh deformation method applied to a simple NACA 0012 rotor geometry in a hover condition. To further validate the applicability of the current mesh deformation method for complex rotor configurations, a realistic Bell Helicopter M427 main rotor\(^1\) in a forward flight condition is investigated. This rotor has a long blade radius of 5.625 meter, with the tip Mach number of 0.691 and an advanced ratio of 0.306 in forward flight. Figure 10 shows the complex M427 rotor blade geometry including closer views in the blade tip, middle fin, and hub regions. The helicopter rotor cyclic flapping and pitching motion is described by equation (8). The rotor blade has a collective pitch angle of $\varphi_0 = 10.04^\circ$, and coning angle of $\theta_0 = 2.619^\circ$. The cyclic pitching motion is defined as follows: $\alpha_1 = -7.65^\circ$ and $\alpha_2 = -0.96^\circ$. The blade cyclic flapping is defined as: $\alpha_3 = 3.211^\circ$ and $\alpha_4 = -0.421^\circ$. A single blade computational mesh is generated with the wake model\(^2\) to account for the rotor trailing wake effect in forward flights. An interior surface, shown in Figure 11, is generated to embrace the blade which serves as the outer deformed boundary for the deforming mesh method, or the sliding interface for the sliding mesh method. Viscous anisotropic unstructured mesh is generated using mix element cells. The mesh size for the single M427 rotor blade contains 7,937,491 volume nodes with 201,929 blade surface nodes.

One of the objectives of the current study is to develop computational capabilities to model the helicopter rotor in realistic aeroelastic motions including the cyclic pitching, cyclic flapping, and structural deformation. The simulation of the M427 main rotor in forward flight is similar to the NACA 0012 rotor case. A converged steady-state solution is obtained first using a local time step and one Newton iteration at each time step. This converged steady-state solution is then used as the initial solution to start the unsteady time-accurate simulation, where three Newton iterations are used with a minimum time step of one degree of azimuthal angle per time step. Figure 12 shows the instantaneous blade pitching and flapping positions predicted by the mesh deformation method at $90^\circ$, $180^\circ$, $270^\circ$, $360^\circ$ azimuthal angles.
Figure 10. Bell 427 main rotor geometry and surface mesh resolution

Figure 11. Computational model for Bell 427 main rotor in forward flight
The normalized rotor thrust \( (C_T) \), hub pitching moment \( (C_{MX}) \), hub roll moment \( (C_{MY}) \), and hub torque coefficients \( (C_{MZ}) \) obtained predicted by the deformed mesh methods (pitching only, and pitching and flapping) are compared in Figure 13. In order to illustrate the influence of the blade flapping motion on the rotor aerodynamic loading, predicted forces and moments by the mesh deformation method and the sliding interface method are plotted in the figure. It is seen that no significant differences are found on the computed force and moment coefficients by both deforming mesh and sliding mesh methods in the blade cyclic pitching motion. However, predicted peak thrust, pitching moment, and torque coefficients are different from that obtained by the deforming mesh method including the cyclic flapping motion of the blade. In other words, to better predict the rotor aerodynamic performance, the actual rotor blade motion has to be considered in the CFD simulations.

Table 4 lists the computational costs of the unsteady simulation of the Bell M427 main rotor in the forward flight condition, including the costs for the pitching only motion in the mesh deformation method and the sliding interface method, and both pitching and flapping motion in the mesh deformation method. With the greedy surface points of 82 to 94 selected for the mesh deformations, the current approach is very comparable to the sliding method (with pitching only), which only costs 8% more than the sliding method in the CFD computations. Over all, the computational overhead is nominal, and the current mesh deformation method is very efficient and robust in handling realistic and complex configurations involving large-scale viscous unstructured meshes.

<table>
<thead>
<tr>
<th>Method</th>
<th>CPU time (s)</th>
<th>Wall time (s)</th>
<th>Greedy points selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mesh deformation (pitch only)</td>
<td>58612.21</td>
<td>58612.21</td>
<td>82</td>
</tr>
<tr>
<td>Mesh deformation (pitch+flap)</td>
<td>62433.51</td>
<td>62433.51</td>
<td>94</td>
</tr>
<tr>
<td>Sliding interface (pitch only)</td>
<td>57669.57</td>
<td>57669.57</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Table 4. Computing time and greedy points selected for the Bell M427 main rotor in five revolutions
VI. Conclusion

An efficient and robust mesh deformation method and module has been developed and implemented in the unstructured grid CFD solver U^2NCLE. The surface mesh deformation was carried out using the beam approximation, and the volume deformation has utilized radial basis functions (RBFs) for an efficient and robust mesh deformation. This mesh deformation module is designed as a part of the aeroelastic analysis tool being developed for rotorcraft aeromechanical problems. Numerical validations have been conducted on two rotor configurations, a NACA 0012 rotor in hover and a Bell M427 main rotor in a forward flight. Numerical results indicated that the current mesh deformation method is accurate, efficient, and robust in handling large-scale unstructured viscous meshes for complex rotor configurations.

Acknowledgements

This research was supported by Bell Helicopter IR&D program, with John Bridgeman and Jim Narramore as the Technical Point of Contacts. The authors are grateful to their guidance and suggestions to this research and to their long-term support to the U^2NCLE code development.

Thanks also go to Kenneth Miller of the Department of Electrical Engineering and Computer Sciences for administrating and maintaining the HPC cluster computer at the University of Toledo.

Figure 13. Comparisons of normalized rotor thrust ($C_T$), hub pitching moment ($C_{MX}$), hub roll moment ($C_{MY}$) and hub torque ($C_{MZ}$) coefficients of Bell 427 Rotor in Forward Flight
References