A Simplified Mesh Deformation Method Using Commercial Structural Analysis Software

Su-Yuen Hsu*, Chau-Lyan Chang†, Jamshid Samareh‡
NASA Langley Research Center
Hampton, VA 23681

Submitted to
10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference
Albany, New York, Aug. 30-Sep. 1, 2004

* Research Engineer, Multidisciplinary Optimization Branch. AIAA Member

† Research Scientist, Computational Modeling and Simulation Branch. AIAA Senior Member

‡ Senior Research Scientist, Multidisciplinary Optimization Branch. AIAA Associate Fellow

Abstract

Mesh deformation in response to redefined or moving aerodynamic surface geometries is a frequently encountered task in many applications. Most existing methods are either mathematically too complex or computationally too expensive for usage in practical design and optimization. We propose a simplified mesh deformation method based on linear elastic finite element analyses that can be easily implemented by using commercially available structural analysis software. Using a prescribed displacement at the mesh boundaries, a simple structural analysis is constructed based on a spatially varying Young's modulus to move the entire mesh in accordance with the surface geometry redefinitions. A variety of surface movements, such as translation, rotation, or incremental surface reshaping that often takes place in an optimization procedure, may be handled by the present method. We describe the numerical formulation and implementation using the NASTRAN software in this paper. The use of commercial software bypasses tedious reimplementation and takes advantage of the computational efficiency offered by the vendor. A two-dimensional airfoil mesh and a three-dimensional aircraft mesh were used as test cases to demonstrate the effectiveness of the proposed method. Euler and Navier-Stokes calculations were performed for the deformed two-dimensional meshes.

I. Introduction

Most existing deformation methods first deform the boundary mesh, and then try to repair the interior mesh by moving, reconnecting, adding, or deleting mesh points. For unstructured meshes with large geometry changes, a new mesh may need to be regenerated. Botkin[1] introduces a local mesh regeneration procedure that operates only on the specific edges and faces associated with the design variable changes. This technique is suitable for computational structural mechanics (CSM) mesh generation. Similarly, Kodiyalam, Kumar, and Finnigan[2] use a mesh regeneration technique based on the assumption that the model topology stays fixed for small perturbations. Model topology comprises the number of mesh-points, edges, faces, and the connectivity table. Any change in the mesh topology will cause the model regeneration to fail.

Several techniques can be used to move the interior mesh points. Gnoffo[3] presents an algorithm for moving mesh points to resolve various flow features. The algorithm is based on a spring analogy, in which mesh points are connected to each other by a set of springs. The stiffness for these springs is a function of the flow gradients. Nakahashi and Deiwert[4] extend Gnoffo's algorithm by including torsional springs that provide a control for the mesh angles. Batina[5] gives a similar mesh deformation algorithm for which the spring stiffness for a given element edge is taken to be inversely proportional to the edge length. Then, the mesh movement is computed through the static equilibrium equation. The spring stiffness is relatively large for small cells. Therefore, these small cells, which are usually near the boundary of the body, tend to undergo rigid body movement. This rigid body movement avoids rapid variations in deformation, thus eliminating the possibility of small cells having very large changes in volume. These large changes could lead to negative cell volumes. Blom[6] provides a detailed analysis for the spring method and draws an analogy between the spring method and an elliptic differential equation approach for structured mesh generation.

The spring analogy is simple to implement, but it is not robust. Several attempts have been made to revise the spring formulation to improve the robustness. Zhang and Belegundu[7] present an algorithm similar to the spring analogy that can handle large mesh deformation. They use the ratio of the cell Jacobian to the cell volume for the spring stiffness. Samareh[8] augments the spring analogy with quaternions to preserve boundary orientation for structured and unstructured viscous-mesh deformations. The quaternion approach preserves the boundary orientation. Because the boundary quaternions are based on changes in the boundary mesh-point positions and orientations, the algorithm guarantees that the boundary orientations of the deformed mesh have the same quality as those of the undeformed mesh. Preliminary two-dimensional (2-D) results indicate the traditional algorithms such as transfinite interpolation (TFI) and the spring analogy can be easily augmented with quaternions to preserve mesh quality near the viscous boundary.

Farhat et al.[9] propose a modification to the spring analogy algorithm to include an additional torsional spring to control mesh skewness and folding for 2-D meshes. They demonstrate that the modified algorithm has advantages in terms of robustness, quality, and performance. The 2-D torsional spring is similar to the approach of Nakahashi and Deiwert.[4] Murayama, Nakahashi, and Matsushima[10] present an extension of the spring analogy with a torsional spring for three dimensions. They conclude that the method is simple and robust.

Another class of deformation algorithm is based on modeling the interior mesh deformation δ by partial differential equations. The Laplace equation, $\nabla^2 \delta = 0$, is the simplest and most widely used form. This algorithm is effective for small deformations. Crumpton and Giles[11] propose a formulation based on the heat conduction equation (Laplace) with the coefficient of thermal conductivity *k* inversely proportional to the cell volume; that is, $\nabla \cdot (k \nabla \delta) = 0$. Crumpton and Giles attribute their success to their choice of cell volume used in the criteria for a valid mesh. One disadvantage of solving a linear Laplace equation is that the components of the mesh deformation are solved independently of each other. For example, if a boundary is moved only along coordinate *x*, the interior mesh points will be moved only along coordinate *x*. This behavior limits the use of Laplace deformation techniques.

The formulation of Ref. 9 has some similarities to linear elasticity formulation. Tezduyar and Behr[12] propose an algorithm based on linear elasticity, which includes a full stress tensor. Cavallo et al.[13] apply this method to mesh deformation for aero and propulsive flow field calculations, and use a generalized minimal residual algorithm to solve the linear elasticity equations. They note that the method preserves the mesh quality, and it produces a better mesh than the spring analogy method. The linear elasticity approach requires solving the complete stress tensor. In contrast, the spring analogy represents only the diagonal elements of the stress tensor. Cavallo et al. conclude that the inclusion of additional stress terms in the linear elasticity can tolerate higher degrees of boundary motion, but it is considerably more expensive. Nielsen and Anderson[14] use a linear elasticity approach, and their conclusions are similar to those of Cavallo et al.

In this paper, we propose a method along the line of a linear elasticity approach. However, in contrast to other approaches, we use two important aspects of the structural analysis. Firstly, we use the spatially varying elastic property as a simple and efficient means to control the mesh movement. Secondly, we take advantage of the commercially available finite-element based structural analysis tools. These well-developed tools make it unnecessary to reimplement the finite-element structural analysis. In the meantime, the computational efficiency offered by the software makes the mesh deformation process much more viable for routine usage in a design cycle. We present the numerical formulation, details of implementation, and test cases in the following sections.

II. Mesh Deformation Technique

Without loss of generality, we consider a meshed domain V as shown in Fig. 1. The boundary surface S consists of an inner boundary S_i and an outer boundary S_o. The mesh is expected to deform to a new one that has good mesh quality and meets certain displacement requirements over the entire boundary (S). The required boundary displacement may arise during an optimization procedure where part of the boundary surface (usually the inner one in aerodynamic applications) has to be redefined. In unsteady aerodynamic applications such as simulation of pitching or yawing wings, the inner boundary often moves according to a prescribed unsteady motion. To obtain a deformed mesh, the original mesh is viewed as a finite-element mesh for a domain of an isotropic linear elastic material, and the given displacement at each boundary node is used as the boundary condition for the elasticity problem. The desired deformed mesh is thus defined by a numerical solution for the nodal displacement. Because no "realistic" structural response is desired, a linear finite-element analysis suffices.

For the linear elasticity problem of interest, the volume average of the strain tensor depends on the displacement specified over the entire boundary, but it is independent of the elastic properties. This fact is obvious from the following derivation:

$$\frac{1}{V_0}\int_V \varepsilon_{ij} dv = \frac{1}{V_0}\int_V \frac{1}{2}\left(u_{i,j} + u_{j,i}\right) dv = \frac{1}{2V_0}\int_S (u_i n_j + u_j n_i) ds \tag{1}$$

where i = 1, 2, or 3, j = 1, 2, or 3, u_i and ε_{ij} are the Cartesian tensors of displacement and strain, respectively, V represents the domain occupied by the elastic continuum, S is the boundary of V, n_i is the unit vector normal to S, and V_0 is the volume of V. Because the average strain for a specific mesh deformation problem is fixed, a significant reduction in strain concentration is usually accompanied by a strain increase in major low strain regions.

The mesh deformation is accomplished by two consecutive linear finite-element analyses using the original mesh. The first one is performed with uniform elastic properties to produce a baseline solution followed by the second one with non-uniform elastic properties. The nodal displacement resulting from the second analysis defines the desired new location of each node. To better control the mesh movement, the Young's modulus, E, and Poisson's ratio, ν, of the elastic continuum are assumed as follows:

$$E = 1 + f(T) \tag{2}$$

$$\nu = 0 \tag{3}$$

where $f(T)$ is a monotonically increasing function of T with $f(0) = 0$, and T is a spatially varying isotropic scalar parameter that satisfies $0 \le T \le 1$. The vanishing Poisson's ratio is intended to allow more deformation in low strain regions.

For the first round of analysis, $T = 0$ (consequently, $E = 1$) is assumed throughout the domain. The homogeneity of the material property used for the precursor analysis usually leads to severe straining of some elements; the deformed mesh is thus unacceptable. Nevertheless, the deformation serves as a guideline for the determination of the non-uniform T function to be used for the second round of analysis. It appears that T should assume a larger value in a more severely strained element so that according to Eqs. 2 and 3, greater stiffness is used for that element. We use the positive definiteness and isotropy of the strain energy density to define T for the second analysis as follows:

$$T = \frac{\sqrt{U} - \sqrt{U_{min}}}{\sqrt{U_{max}} - \sqrt{U_{min}}} \tag{4}$$

where U is the strain energy density obtained from the first finite-element solution, and U_{max} and U_{min} are the maximum and minimum of U over the entire domain, respectively.

Note that there is freedom in choosing $f(T)$. The simplest functional form is a linear function,

$$f(T) = cT \tag{5}$$

with the constant 1□ c. This linear function has been used exclusively and has produced quality results in a number of numerical experiments. Our experience suggests that use of $c = 10^3$ should yield satisfactory results for most cases. Once T is evaluated by using the solution from the first analysis, and a suitable value of c is selected, the second round of analysis can be performed to obtain the deformed mesh.

III. Implementation

It is evident that the proposed mesh deformation method may be implemented with a basic linear elasticity finite-element code. In the present investigation, we use the commercially available NASTRAN code of the MSC Software Corporation. The parameter T, defined by Eq. 4, is an element data since the strain energy density U is essentially an element quantity (as opposed to nodal quantity such as a displacement) output from the first analysis. NASTRAN allows use of a temperature-dependent linear elastic material whose properties are calculated internally using element temperature data. For the 2-D case, we may take T for temperature because NASTRAN accepts element temperature input. For the three-dimensional (3-D) case, NASTRAN accepts only nodal temperature input, computes the average of nodal temperature for each element, and uses the average as the element temperature required for the material property calculation. To avoid the error introduced by the conversion from an element quantity to a nodal quantity and then back to an element quantity, we do not use the NASTRAN temperature-dependent material for the 3-D case. Instead, the input file for the second analysis is obtained by modifying the input file for the first analysis so that in the second input file, each 3-D element refers to an element-specific material, whose Young's modulus is calculated by using Eqs. 2, 4, and 5 externally, and is written together with the zero Poisson's ratio (Eq. 3) into the input file.

Implementation of the proposed method for the 2-D case is depicted in Fig. 2. The two rounds of analysis may be performed by using a simple NASTRAN input file with minor modifications. As shown, the material model (Eqs. 2, 3 and 5) is defined by merely specifying a few lines in the input file. For the first analysis, the "TEMPERATURE (MATERIAL)" and "INCLUDE" commands are commented out. As a result, E = 1 and $\nu = 0$ are used in the analysis. The element strain energy density (the output generated by the ESE command of the first analysis) is then post-processed by using Eq. 4 to create the element temperature data file (named temperature.bdf in Fig. 2) according to the NASTRAN input format for 2-D element temperature. By uncommenting the TEMPERATURE (MATERIAL) and INCLUDE commands in the input file shown in Fig. 2, the calculated temperature field is employed to compute the Young's modulus element by element during the second analysis.

Implementation of the proposed method for the 3-D case is depicted in Fig. 3. As seen in Fig. 3, homogeneous material properties are used for the first analysis, and each element refers to a specific material in the second analysis. A trivial code, which reads in the element definitions from the input file for the first analysis and the element strain energy density from the first analysis results, can generate all the necessary element and material definitions for the second analysis.

In a typical computational fluid dynamics (CFD) mesh, the outer boundary is usually an artificially created free-stream boundary. During the mesh deformation process, this boundary is often held fixed. Therefore, for all our results presented herein, we assume a zero displacement at the outer boundary. On the other hand, a displacement is prescribed at each node on the inner boundary (e.g., airfoil surface) in order to produce a specific deformation of the mesh.

As mentioned before, we need to determine the parameter c. Judging from the high strain concentration that is expected to be captured by the fine mesh around the theoretical singularity, we started with $c = 10^3$ for all the numerical examples, and obtained consistently good results. Subsequently, $c = 10^4$ was used, and even better deformed meshes were achieved. Finally, $c = 10^5$ was tested; however, we observed very little improvement from using this value. Therefore, all the results presented herein were generated with $c = 10^4$.

IV. 2-D Test Cases

The chosen 2-D mesh is intended for a transonic flow over an airfoil. The baseline mesh is an unstructured mesh for a 2-D airfoil with triangular elements as shown in Fig. 4. The mesh has a total of 18812 elements and 9598 nodes with many of them clustering near the airfoil surface. The chord of the airfoil is parallel with the x-axis and has a length of 1 unit. To demonstrate the effectiveness of the present method, we attempt to deform the mesh in four different ways: 1) by a translation of the airfoil by 0.5 unit in the flow (x) direction, 2) by a 0.5-unit upward (y) translation, 3) by a 45° rotation, and 4) by cambering the airfoil shape via imposing an arc-shaped distribution of upward (y) displacement (zero at both leading and trailing edges, and a peak displacement of 0.125 unit at the mid-chord position). The camber movement is often encountered in a design optimization process, while the rigid-body motion cases may appear in both design optimization and unsteady aerodynamic analysis. Each NASTRAN run took about

13 seconds on a computer with an Intel Pentium III 930 MHz processor and the Linux operating system.

Figures 5-8 show the mesh patterns resulting from the first stage finite-element analysis for the above four target deformation cases, respectively. The uniform elastic property is employed at this stage, and the resulting mesh is of very poor quality. It appears that the near-wall mesh points have been dragged along with the airfoil motion. In particular, “overspill” (non-unique mapping) occurs at the trailing edge for all the cases. Furthermore, the near-wall elements lose the quasi-orthogonality of the baseline mesh. These poor-quality meshes are a natural consequence of the imposed large displacement within the framework of linear elasticity theory.

The final meshes resulting from the second stage finite-element analysis for the four deformation cases are presented in Figs. 9-12, respectively. Evidently, the artificial temperature field created in the present analyses along with the varying elastic properties has made the problem well posed. The leading and trailing edges appear to be problem-free, and no overspill is observed in the results. For the two translation cases, the orthogonality of the near-wall mesh is almost completely preserved. For the other two cases, the near-wall mesh, although less orthogonal than for the translation cases, still has good quality. The variable elastic property plays a crucial role in achieving the desirable mesh deformation. The impact of the property variation can be visualized by comparing Fig. 7a (ill-formed mesh) with Fig. 11a (final mesh). It is seen from the comparison that the deformation is successfully propagated toward the low strain regions. A consequence of the propagation is the dramatic reduction in strain concentration, as indicated by Figs. 7b and 11b.

V. 2-D Euler & Navier-Stokes Computations Using the Deformed Meshes

To validate the proposed mesh deformation technique, we solved Euler and Navier-Stokes equations over the original as well as the deformed meshes. All calculations presented here were performed by using a space-time CE/SE method.[15] Figure 13 shows the resulting non-dimensional pressure contours over the horizontally translated mesh (Fig. 9) with a free-stream

Mach number of 0.8. A similar calculation was performed for the rotated mesh (Fig. 11) with the same free-stream Mach number but at a 45° free-stream incident angle to facilitate comparisons with the translated case. The resulting pressure contours shown in Fig. 14 resemble those of the translated case in Fig. 13. Figure 15 shows the airfoil surface pressure distribution for the above two cases along with that generated with the baseline mesh (Fig. 4). Except for the small difference after the shock, the three solutions fall almost on top of one another. It should be noted that the shock solution is usually more sensitive to mesh resolution due to the limiters applied to resolve the discontinuities. Also note that we use the mesh index instead of the x coordinate as the abscissa in the figure because all the three cases share an identical mesh index. In fact, connectivity and other mesh-related information is preserved in the present mesh deformation method. This property has positive implications in many CFD applications. For example, the calculation of sensitivity derivatives in an optimization procedure can be simplified. For the above calculations, the same mesh identity makes it possible to import the converged solution from the original mesh as the initial condition of the deformed mesh directly without any index manipulation. A good initial condition on the deformed mesh eliminates the nonlinear transient effect and substantially speeds up the convergence to steady state.

A Navier-Stokes calculation was also performed by using the cambered mesh with a free-stream Mach number of 0.8 and a chord Reynolds number of 10^4. The resulting Mach number contours are shown in Fig. 16. Due to strong separation associated with the cambered geometry, a steady-state solution is not possible at this Reynolds number. The solution indicates that vortex shedding is about to form, and a moving shock at the bottom surface is also evident. However, the relatively coarse mesh in the wake region prevents these dynamic flow features from being resolved. This camber case also shows that the present mesh deformation approach may be used for mesh generation for a derived airfoil shape (such as a camber) based on an existing, easily generated mesh.

VI. 3-D Test Case

The chosen 3-D mesh is an unstructured mesh created by Frink[16] for a transonic flow passing a transport aircraft. As shown in Fig. 17, the discretized domain for the CFD computation is an orthogonal hexahedron. For explanatory purposes, the high-density surface mesh of the aircraft and a local rectangular coordinate system embedded in the left wing are exposed to the reader through a truncation of the hexahedron. In the 3-D test, the mesh is deformed by rotating the outboard portion of the wing upward about the z-axis of the local coordinate system. The resulting wing tip rise (Fig. 18) is about half the wing tip chord, similar to the 2-D test case reported with Figs. 6 and 10. As seen in Fig. 17, a cutout of the CFD mesh is selected to accommodate the deformation. Therefore, the proposed mesh deformation method is applied to the cutout, which has 855,727 tetrahedral elements and 150,028 nodes. Each of the two NASTRAN runs took about two hours and fifty minutes on a computer with an Intel Xeon 3.06 GHz CPU, 3 GB RAM, and the Linux operating system.

The first layer of elements attached to the wing surface is examined in Fig. 19. A comparison of Fig. 19a and Fig. 19b indicates the presence of overspill at the tip and edges of the wing from the first analysis. In fact, the penetration propagates over about half the rotated portion of the trailing edge. In contrast, no penetration results from the second analysis (Fig. 19c). The deformed geometry of the fifth layer of elements outside the wing surface is compared with the undeformed in Fig. 20. The penetration is still evident in Fig. 20a, although the extent is reduced. A further examination indicates that the penetration propagates through more than ten layers of elements near the trailing edge point of the wing tip.

The quality of the deformed meshes must be examined by inspecting some critical cross sections. The two cross sections shown in Figs. 21a and 22a are inspected. Caution should be taken when inspecting the cross sections because the two cross sections, each consisting of many element faces, are not flat. Although only the close-ups of the trailing edge (Fig. 21f) and wing tip (Fig. 22f) are given herein, the quality of the mesh deformed by the second analysis is consistently satisfactory over the entire cross section, and the quality is comparable with that observed in Fig. 10 for the 2-D case. In contrast, the mesh from the first analysis only reflects the information guiding the material property correction.

VII. Concluding Remarks

A simplified mesh deformation method based on a commercial structural analysis tool is proposed. The present approach requires two consecutive linear elastic finite-element analyses, the first one with homogeneous material properties and the second one with inhomogeneous material properties. The element strain energy density output from the first analysis is used to compute the element-specific Young's modulus in a very simple manner for the second analysis. The proposed two-stage structural analysis may be accomplished via simple NASTRAN input files. Several surface movements such as translation, rotation, and cambering have been investigated using the proposed approach. Consistently good quality of the deformed meshes is verified by visual inspection in details. The quality of the 2-D deformed mesh is further validated by Euler and Navier-Stokes calculations. One distinct feature of the present approach is that the mesh connectivity and indices are completely preserved during the deformation process. This property can greatly simplify the sensitivity calculation required in an optimization process.

The present method also could be used as a dynamic mesh generation tool for a series of unsteady moving-boundary CFD calculations. The main advantage is that the present method allows a fixed outer boundary with a preserved mesh connectivity that in many cases may simplify the numerical solution procedure.

References

1. Botkin, M. E., *Three-Dimensional Shape Optimization Using Fully Automatic Mesh Generation*, AIAA J., Vol. 30, No. 5, pp. 1932–1934, 1992.
2. Kodiyalam, S., Kumar, V., and Finnigan, P., *Constructive Solid Geometry Approach to Three-Dimensional Structural Shape Optimization*, AIAA J., Vol. 30, No. 5, pp. 1408–1415, 1992.
3. Gnoffo, P. A., *A Finite-Volume Adaptive Grid Algorithm Applied to Planetary Entry Flowfields*, AIAA J., Vol. 21, No. 9, pp. 1249–1254, 1983.
4. Nakahashi, K. and Deiwert, G. S., *Self-Adaptive-Grid Method with Application to Airfoil Flow*, AIAA J., Vol. 25, No. 4, pp. 513–520, 1987.
5. Batina, J. T., *Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes*, AIAA J., Vol. 28, No. 8, pp. 1381–1388, 1990.
6. Blom, F. J., *Considerations on the Spring Analogy*, Int. J. Numer. Methods Fluids, Vol. 32, No. 6, pp. 647–668, 2000.
7. Zhang, S. and Belegundu, A. D., *A Systematic Approach for Generating Velocity Fields in Shape Optimization*, Struct. Opt., Vol. 5, No. 1–2, pp. 84–94, 1993.
8. Samareh, J. A., *Application of Quaternions for Mesh Deformation*, NASA TM 2002-211646, 2002.
9. Farhat, C., Degand, C., Koobus, B., and Lesoinne, M., *Torsional Springs for Two-Dimensional Dynamic Unstructured Fluid Meshes*, Comput. Methods Appl. Mech. Engrg., Vol. 163, pp. 231–245, 1998.
10. Murayama, M., Nakahashi, K., and Matsushima, K., *Unstructured Dynamic Mesh for Large Grid Movement and Deformation*, AIAA Paper 2002-0122, 2002.
11. Crumpton, P. I. and Giles, M. B., *Implicit Time-Accurate Solutions on Unstructured Dynamic Grids*, Int. J. Numer. Methods Fluids, Vol. 25, No. 11, pp. 1285–1300, 1997.
12. Tezduyar, T. E. and Behr, M., *A New Strategy for Finite-Element Computations Involving Moving Boundaries and Interfaces- the Deforming-Spatial-Domain/Space-Time Procedure: I. the Concept and the Preliminary Numerical Tests*, Comput. Methods Appl. Mech. Engrg., Vol. 94, pp. 339–351, 1992.
13. Cavallo, P. A., Hosangadi, A., Lee, T. A., and Dash, S. M., *Dynamic Unstructured Grid Methodology with Application to Aero/Propulsive Flowfields*, AIAA Paper 97-2310, 1997.

14. Nielsen, E. J. and Anderson, W. K., *Recent Improvements in Aerodynamic Design Optimization on Unstructured Meshes*, AIAA J., Vol. 40, No. 6, pp. 1155–1163, 2002.

15. Chang, C.-L., *Compressible Navier-Stokes Computations Using the Space-Time CE/SE Method*, NASA TM in preparation, 2004.

16. Frink, N. T., example mesh used in 2nd AIAA CFD Drag Prediction Workshop, Orlando, FL, June 2003.

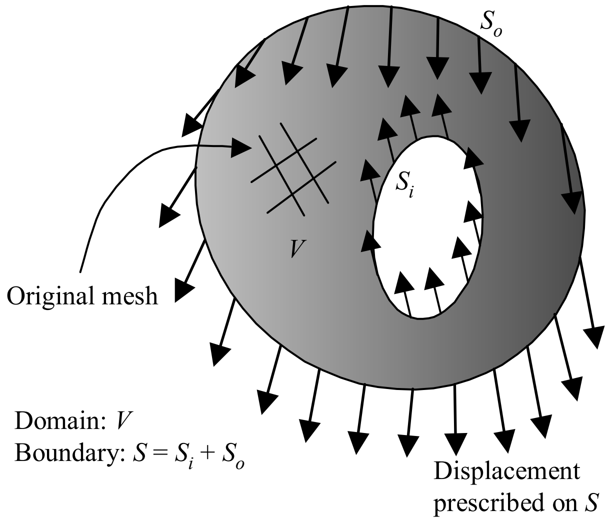
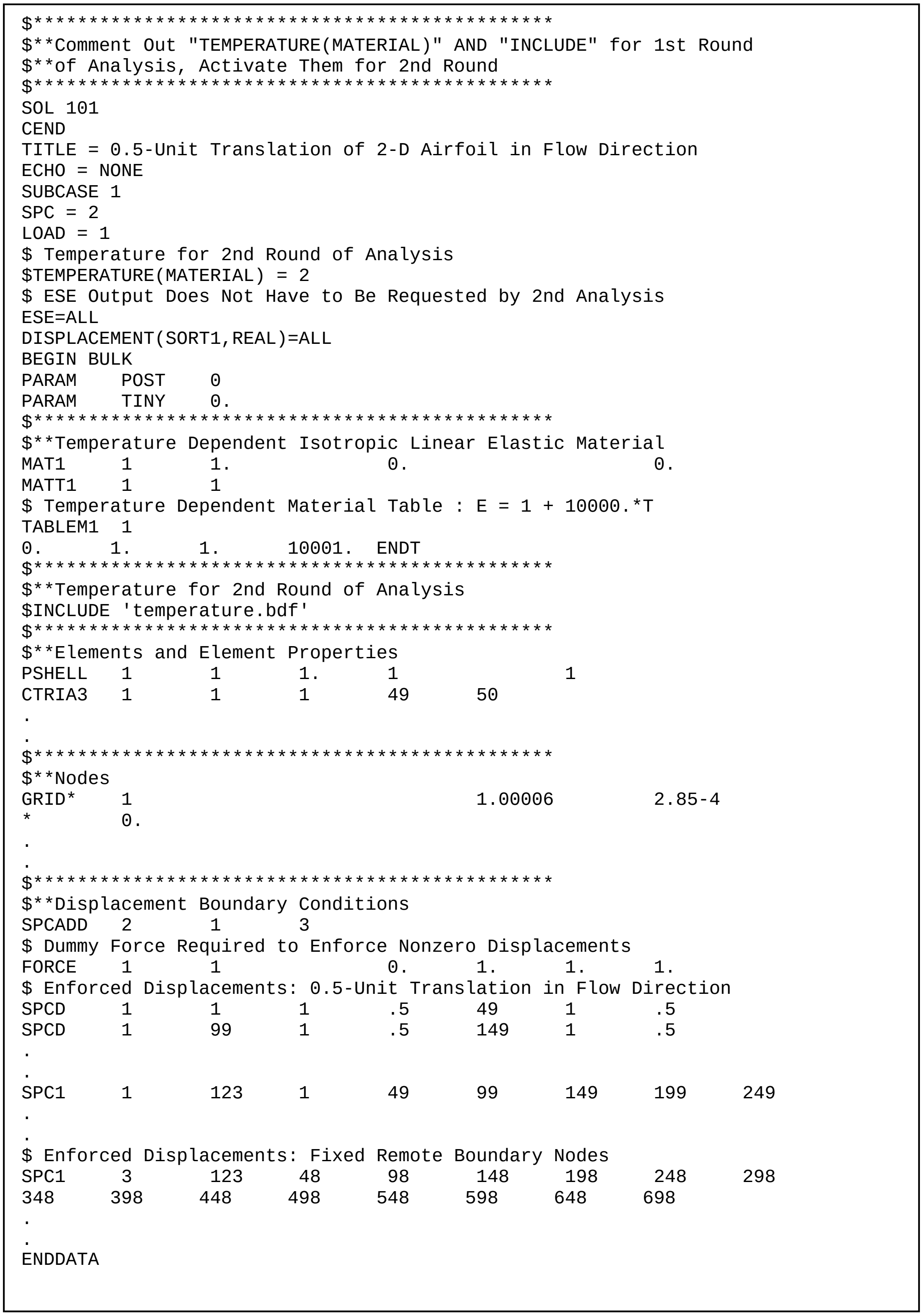


Figure 1: Schematic illustration of linear elasticity problem with displacement-type boundary conditions.



```
$***********************************************
$**Comment Out "TEMPERATURE(MATERIAL)" AND "INCLUDE" for 1st Round
$**of Analysis, Activate Them for 2nd Round
$***********************************************
SOL 101
CEND
TITLE = 0.5-Unit Translation of 2-D Airfoil in Flow Direction
ECHO = NONE
SUBCASE 1
SPC = 2
LOAD = 1
$ Temperature for 2nd Round of Analysis
$TEMPERATURE(MATERIAL) = 2
$ ESE Output Does Not Have to Be Requested by 2nd Analysis
ESE=ALL
DISPLACEMENT(SORT1,REAL)=ALL
BEGIN BULK
PARAM    POST    0
PARAM    TINY    0.
$***********************************************
$**Temperature Dependent Isotropic Linear Elastic Material
MAT1     1       1.              0.                      0.
MATT1    1       1
$ Temperature Dependent Material Table : E = 1 + 10000.*T
TABLEM1  1
0.      1.      1.      10001.  ENDT
$***********************************************
$**Temperature for 2nd Round of Analysis
$INCLUDE 'temperature.bdf'
$***********************************************
$**Elements and Element Properties
PSHELL   1       1       1.      1               1
CTRIA3   1       1       1       49      50
.
.
$***********************************************
$**Nodes
GRID*    1                               1.00006         2.85-4
*        0.
.
.
$***********************************************
$**Displacement Boundary Conditions
SPCADD   2       1       3
$ Dummy Force Required to Enforce Nonzero Displacements
FORCE    1       1               0.      1.      1.      1.
$ Enforced Displacements: 0.5-Unit Translation in Flow Direction
SPCD     1       1       1       .5      49      1       .5
SPCD     1       99      1       .5      149     1       .5
.
.
SPC1     1       123     1       49      99      149     199     249
.
.
$ Enforced Displacements: Fixed Remote Boundary Nodes
SPC1     3       123     48      98      148     198     248     298
348     398     448     498     548     598     648     698
.
.
ENDDATA
```

Figure 2: NASTRAN input file used for 2-D case.

```
SOL 101
CEND
TITLE = 3-D Example, 1st Round of Analysis
ECHO = NONE
SUBCASE 1
SPC = 2
LOAD = 1
$ ESE Output Does Not Have to Be Requested by 2nd Analysis
ESE=ALL
DISPLACEMENT(SORT1,REAL)=ALL
BEGIN BULK
PARAM   POST    0
PARAM   TINY    0.
$***********************************************
$**Elements and Element Properties
PSOLID  1       1       0
MAT1    1       1.              0.0                     0.0
CTETRA  1       1       5592    4966    5768    16709
CTETRA  2       1       5592    4966    16709   16707
CTETRA  3       1       16707   4966    16709   16708
               .
               .
$***********************************************
$**Nodes
               .
               .
$***********************************************
$**Displacement Boundary Conditions
                .
                .
$***********************************************
$**Referenced Coordinate Frames
CORD2R* 1                                181.893       -233.728
*       19.8596         -114.937        -233.728         19.8596
*       181.893         -530.557         19.8596
*
ENDDATA



----------------------------------------------------------------
SOL 101
CEND
TITLE = 3-D Example, 2nd Round of Analysis
                .
                .
$***********************************************
$**Elements and Element Properties
PSOLID  1       1       0
MAT1    1       15.0            0.0                     0.0
CTETRA  1       1       5592    4966    5768    16709
PSOLID  2       2       0
MAT1    2       14.1            0.0                     0.0
CTETRA  2       2       5592    4966    16709   16707
PSOLID  3       3       0
MAT1    3       15.7            0.0                     0.0
CTETRA  3       3       16707   4966    16709   16708
               .
               .
ENDDATA
```

Figure 3: NASTRAN input files used for 3-D case.

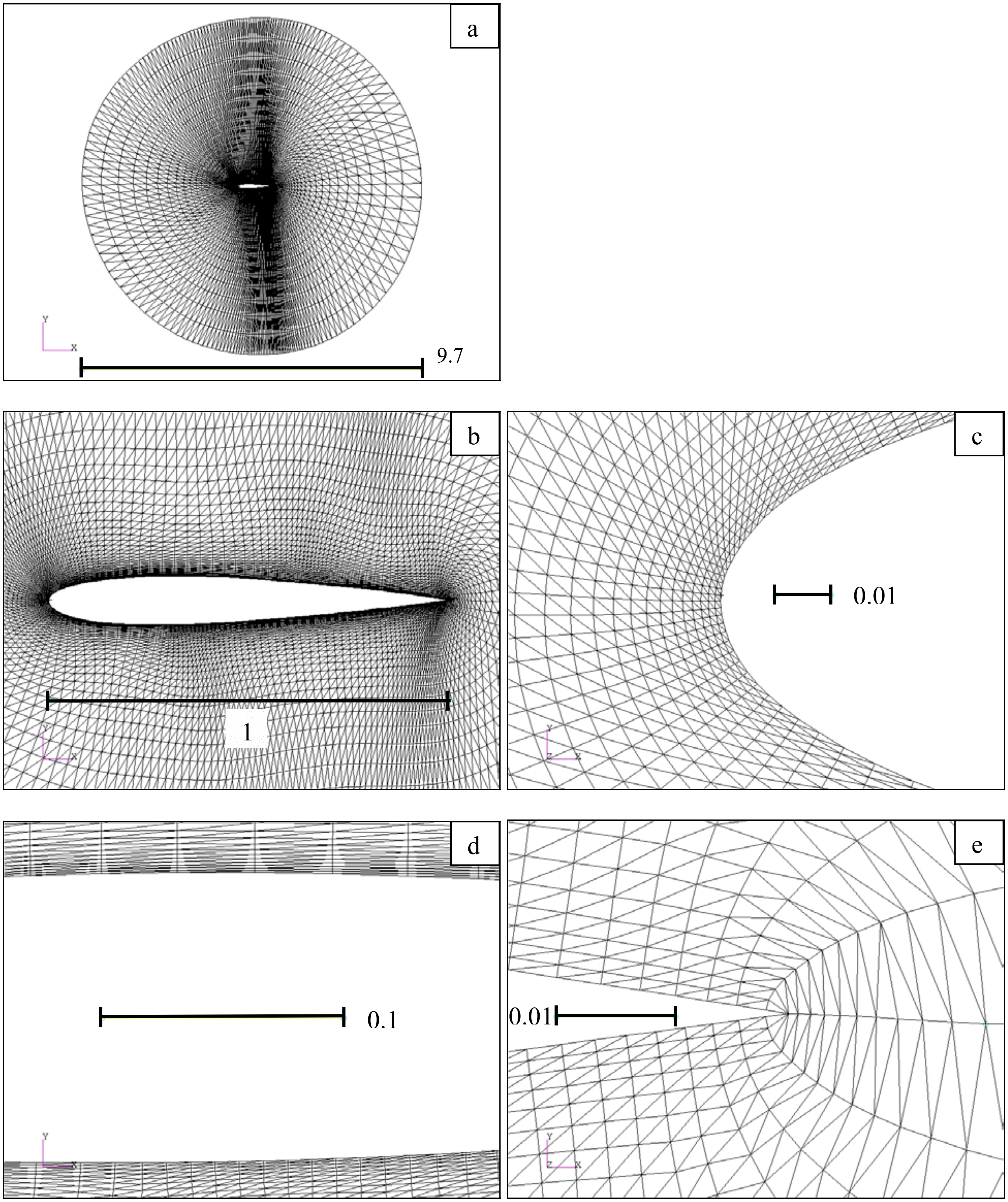


Figure 4: Two-dimensional CFD mesh prior to deformation.

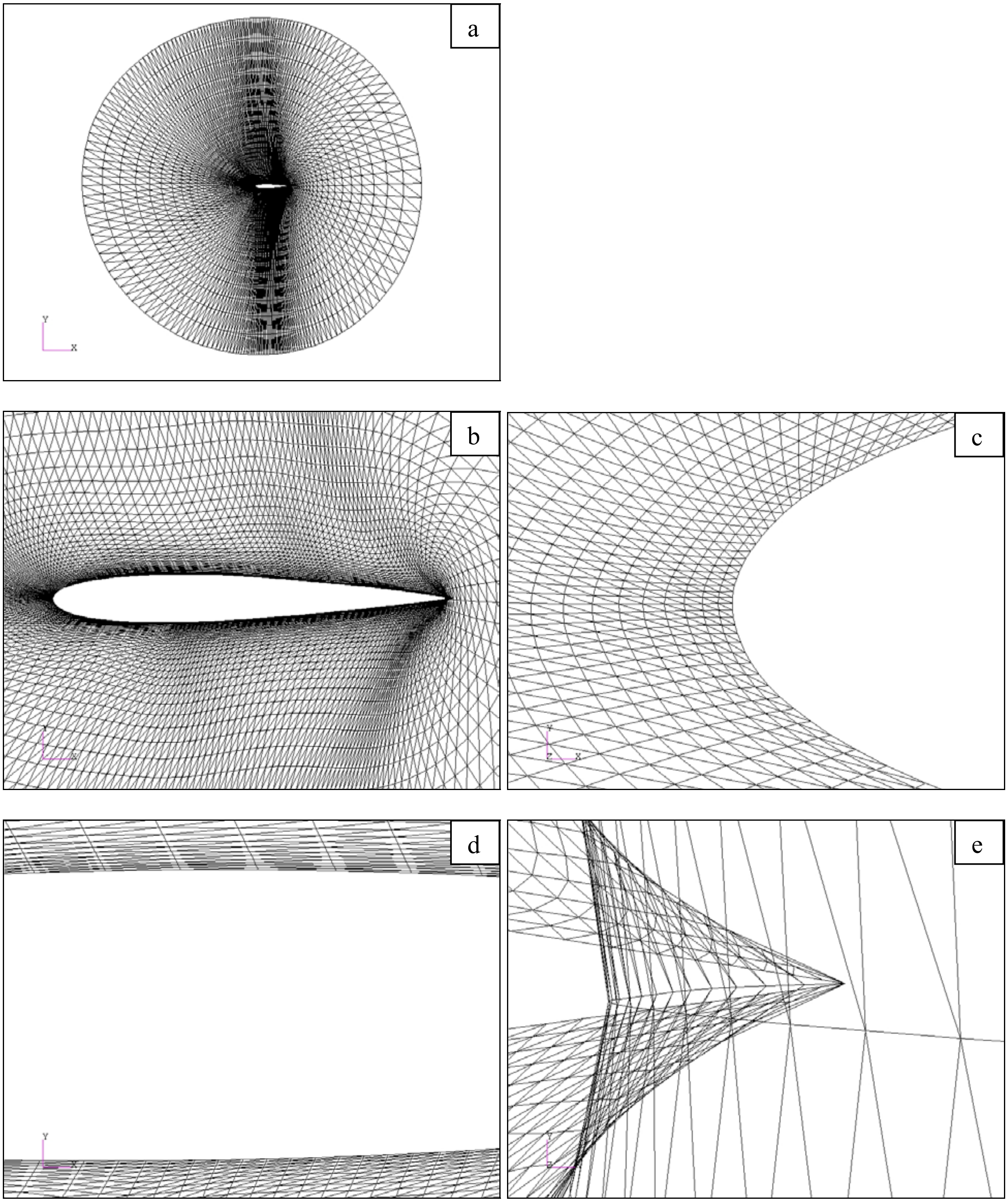


Figure 5: Mesh deformed by 0.5-unit translation of airfoil in flow (*x*) direction, resulting from first analysis.

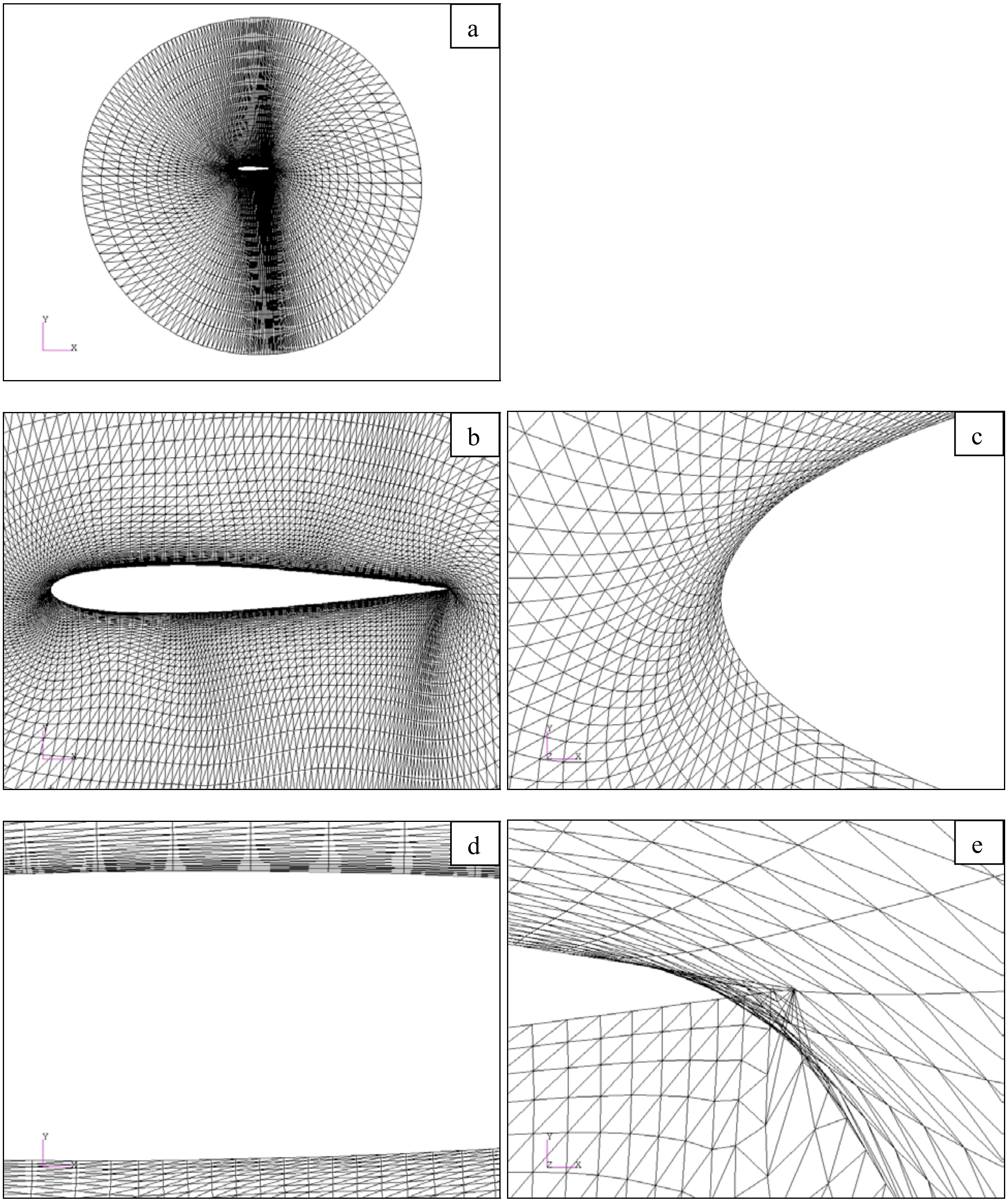


Figure 6: Mesh deformed by 0.5-unit upward (*y*) translation of airfoil, resulting from first analysis.

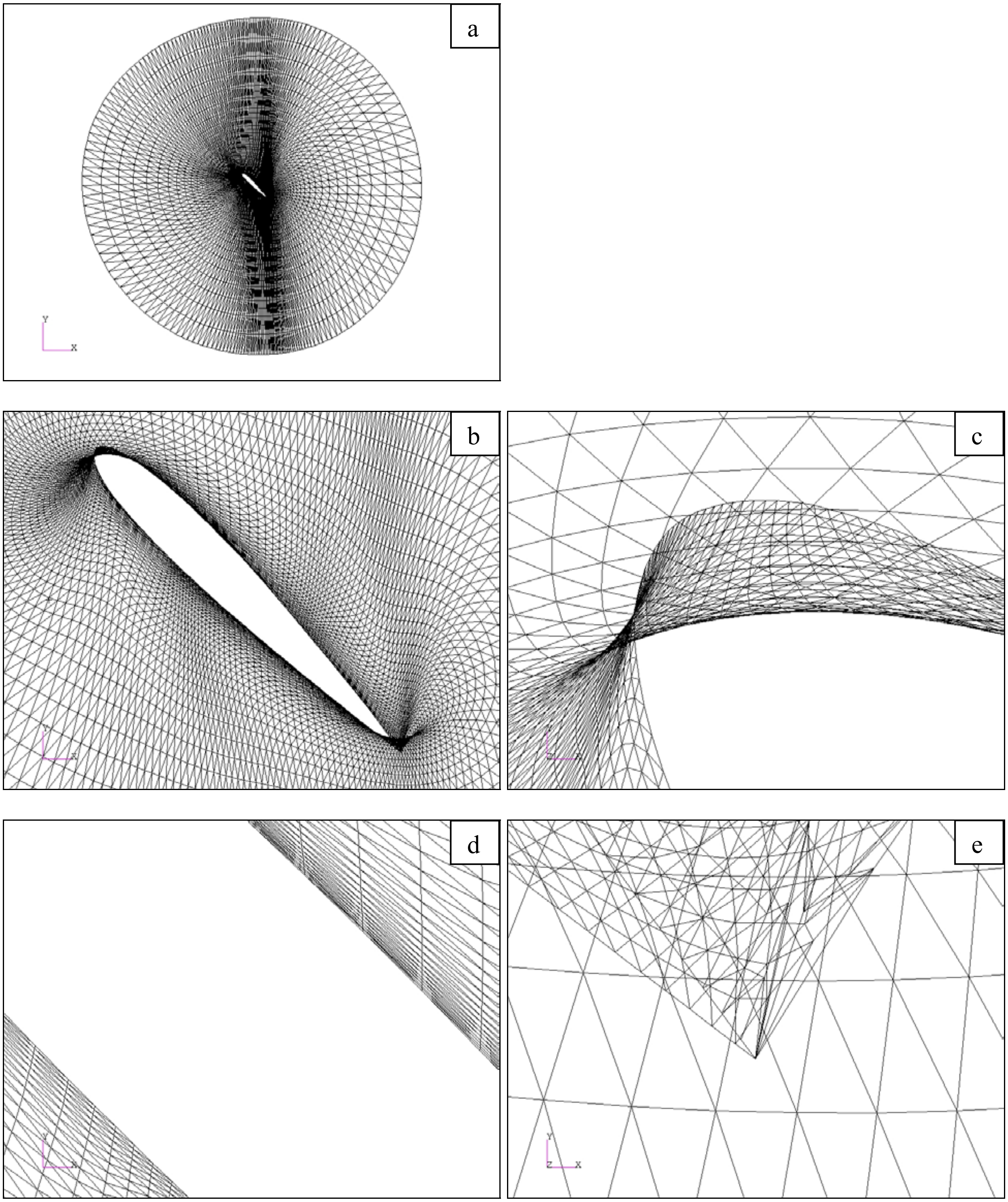


Figure 7: Mesh deformed by 45° rotation of airfoil, resulting from first analysis.

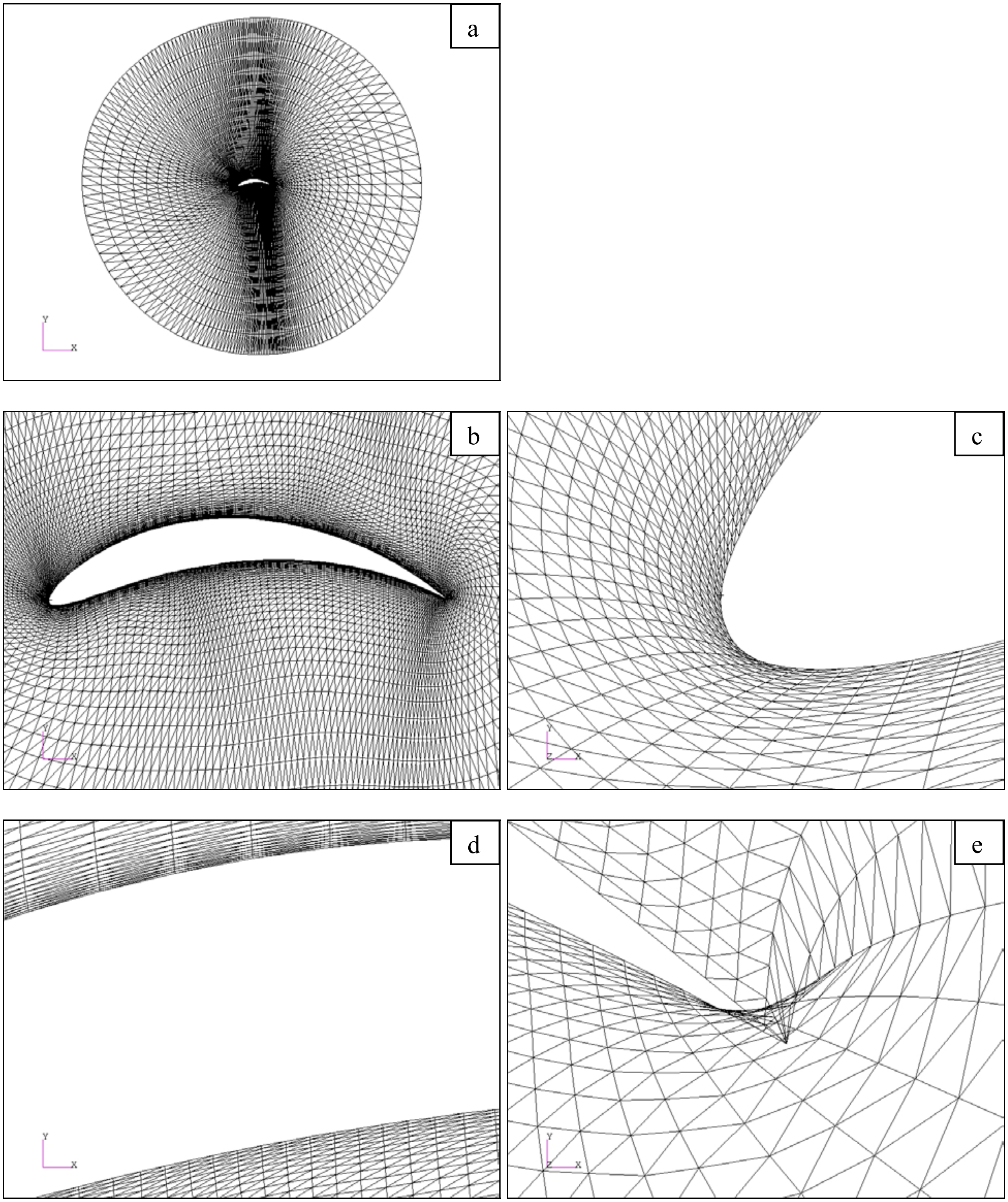


Figure 8: Mesh deformed by imposing arc-shaped distribution of upward (*y*) displacement on entire airfoil surface, resulting from first analysis.

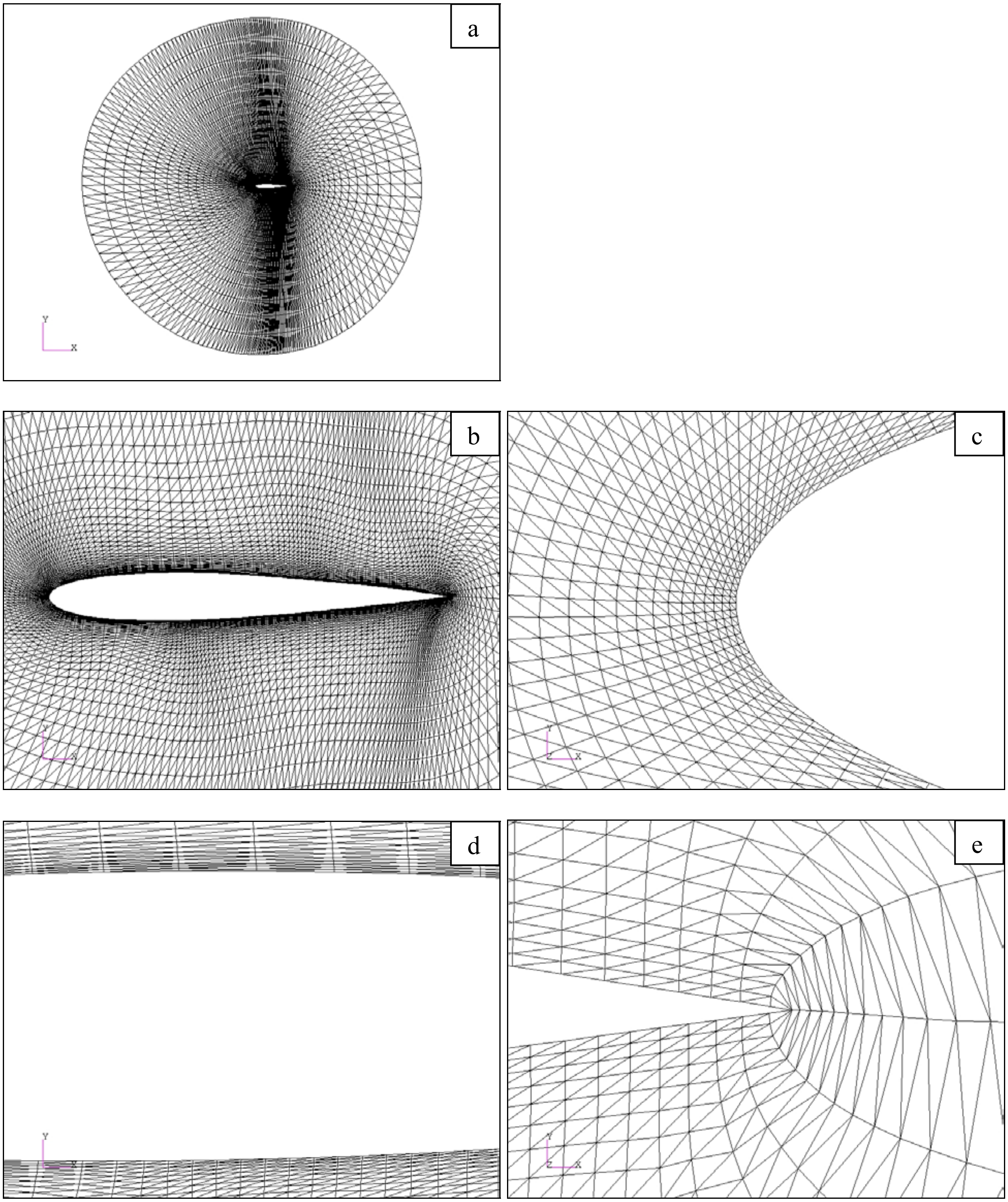


Figure 9: Mesh deformed by 0.5-unit translation of airfoil in flow (*x*) direction, resulting from second analysis.

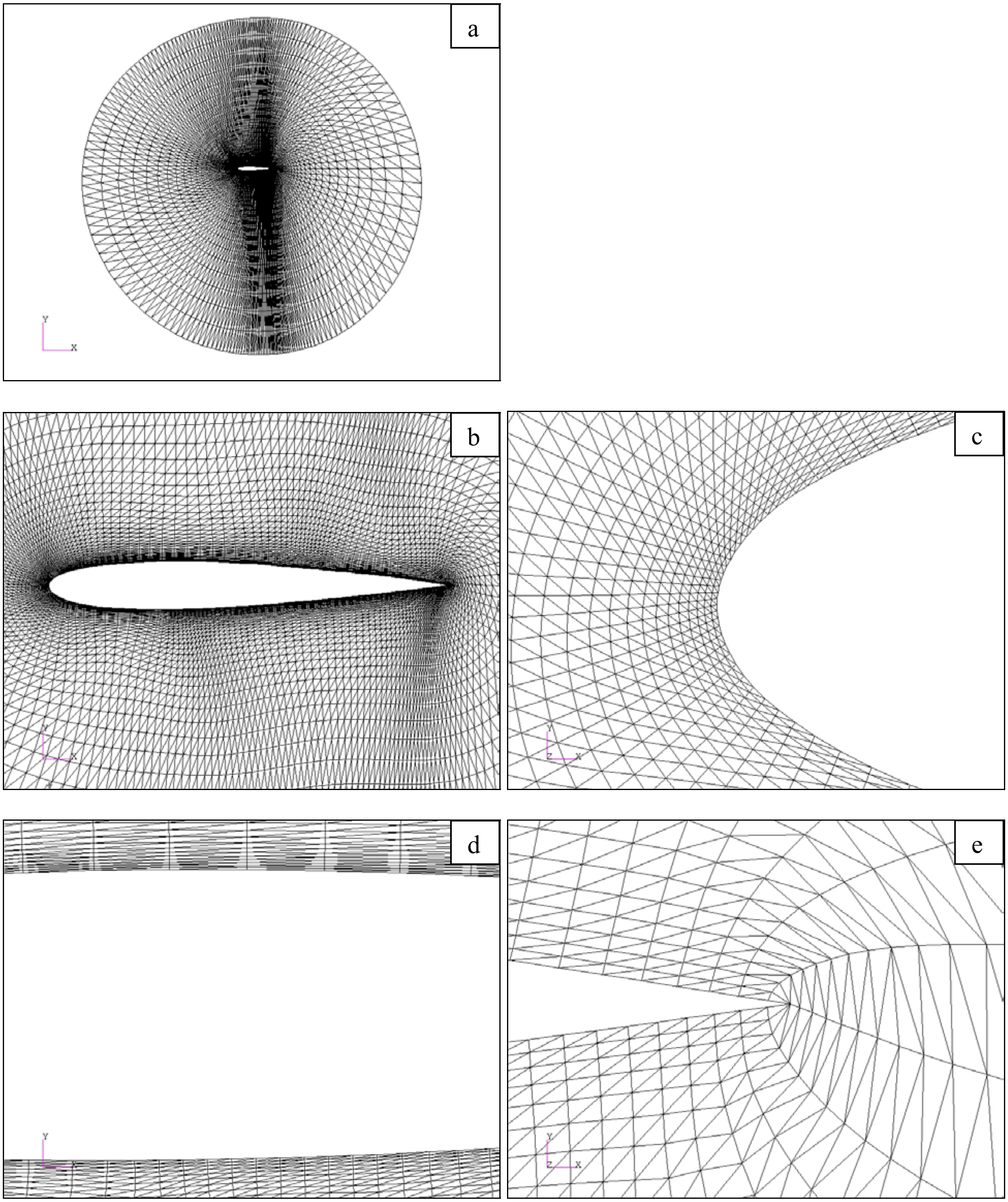


Figure 10: Mesh deformed by 0.5-unit upward (y) translation of airfoil, resulting from second analysis.

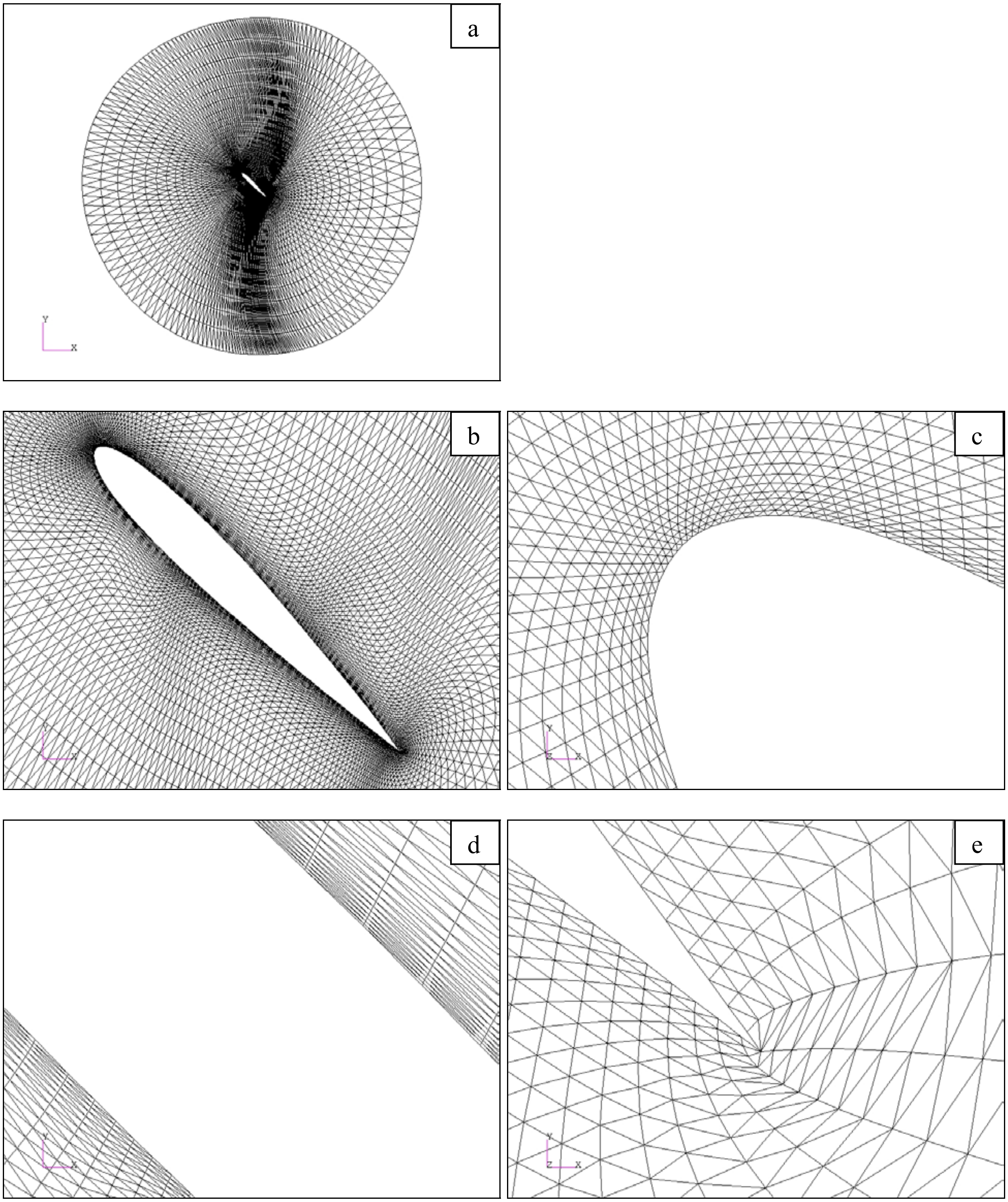


Figure 11: Mesh deformed by 45° rotation of airfoil, resulting from second analysis.

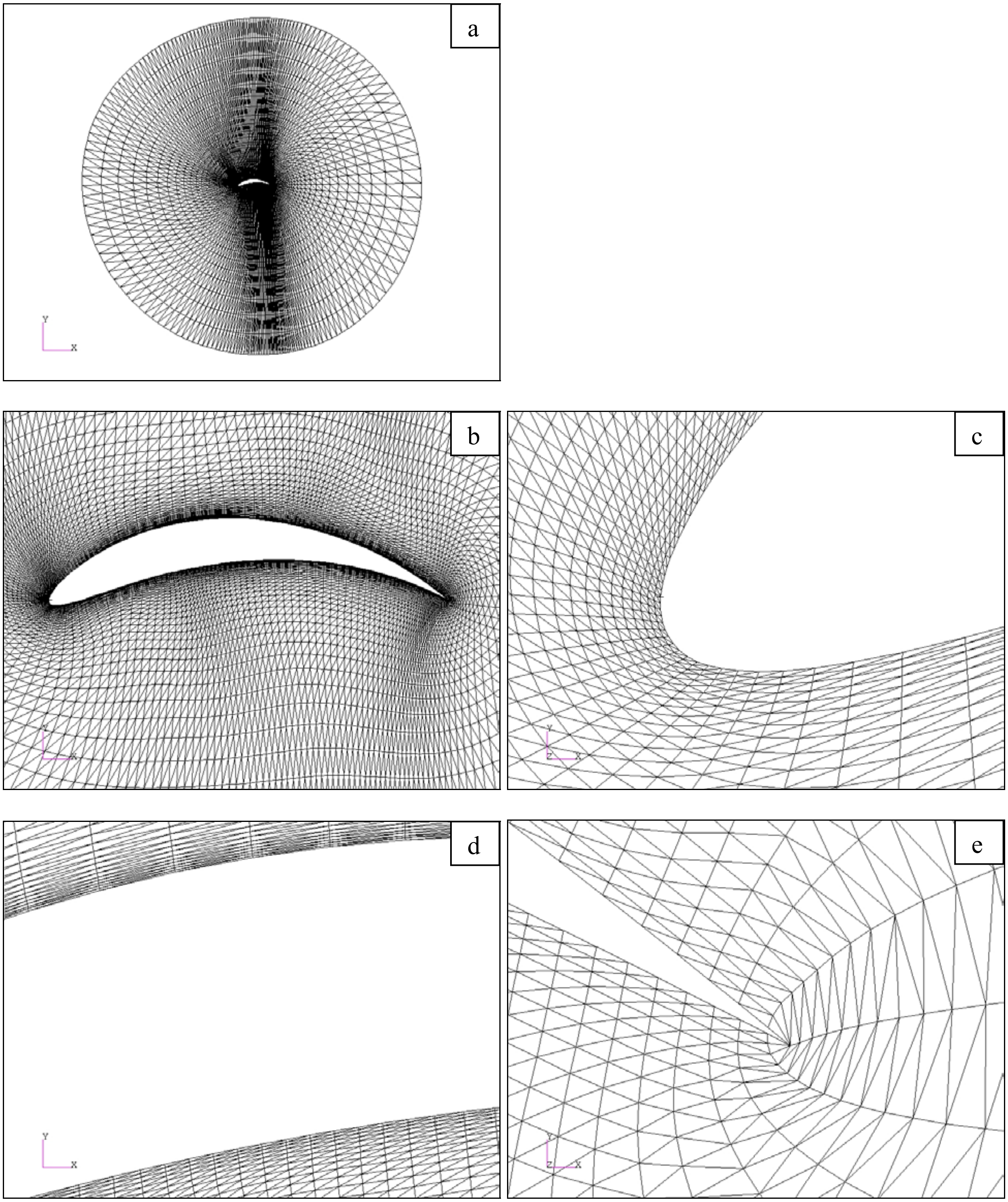


Figure 12: Mesh deformed by imposing arc-shaped distribution of upward (*y*) displacement on entire airfoil surface, resulting from second analysis.

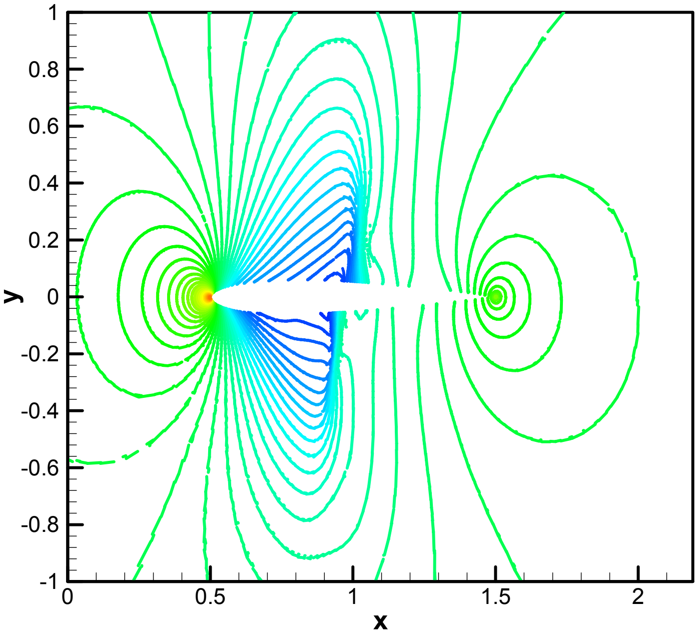


Figure 13: Euler pressure contours computed by using mesh translated in *x* direction.

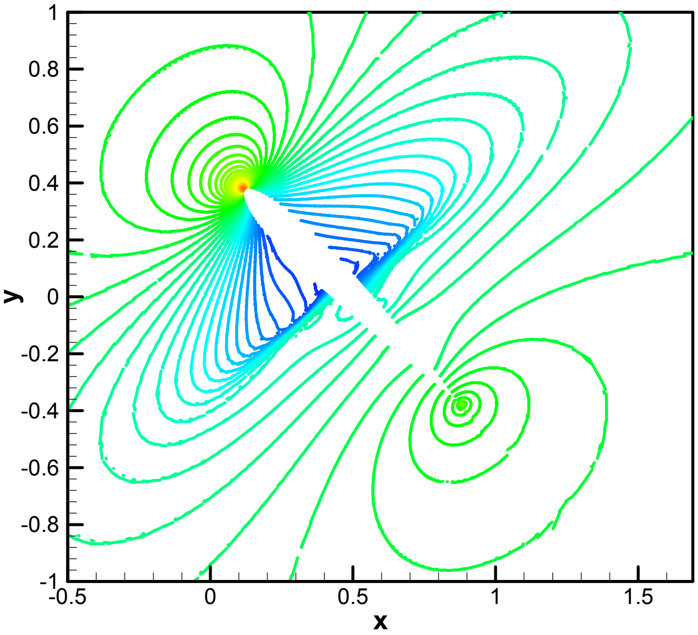


Figure 14: Euler pressure contours computed by using rotated mesh.

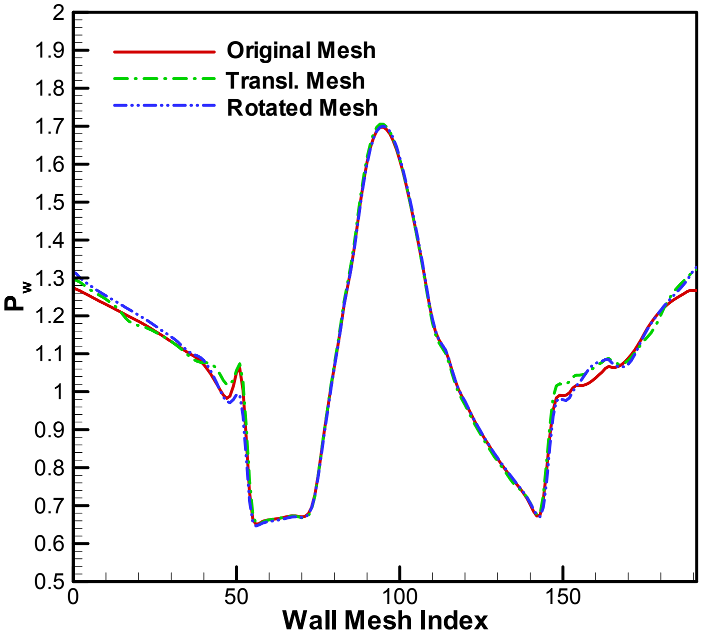


Figure 15: Comparison of non-dimensional wall pressure among three different meshes.

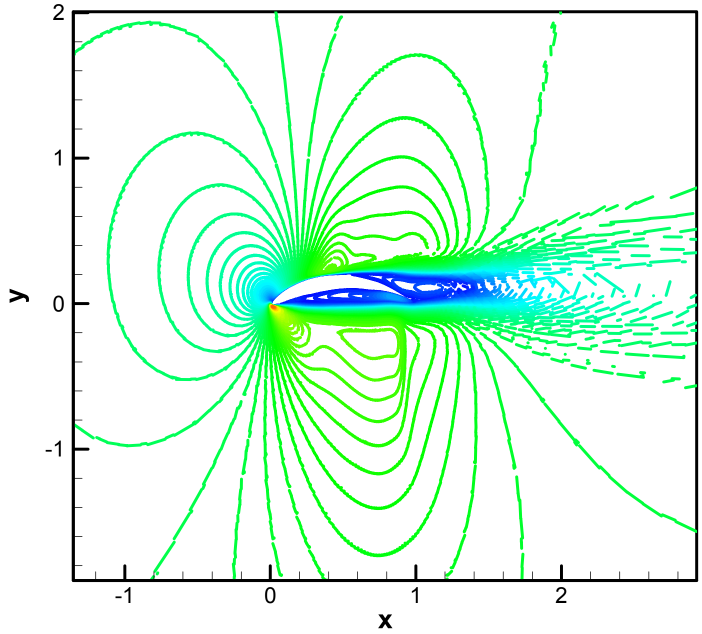


Figure 16: Navier-Stokes Mach number contours computed by using cambered mesh.

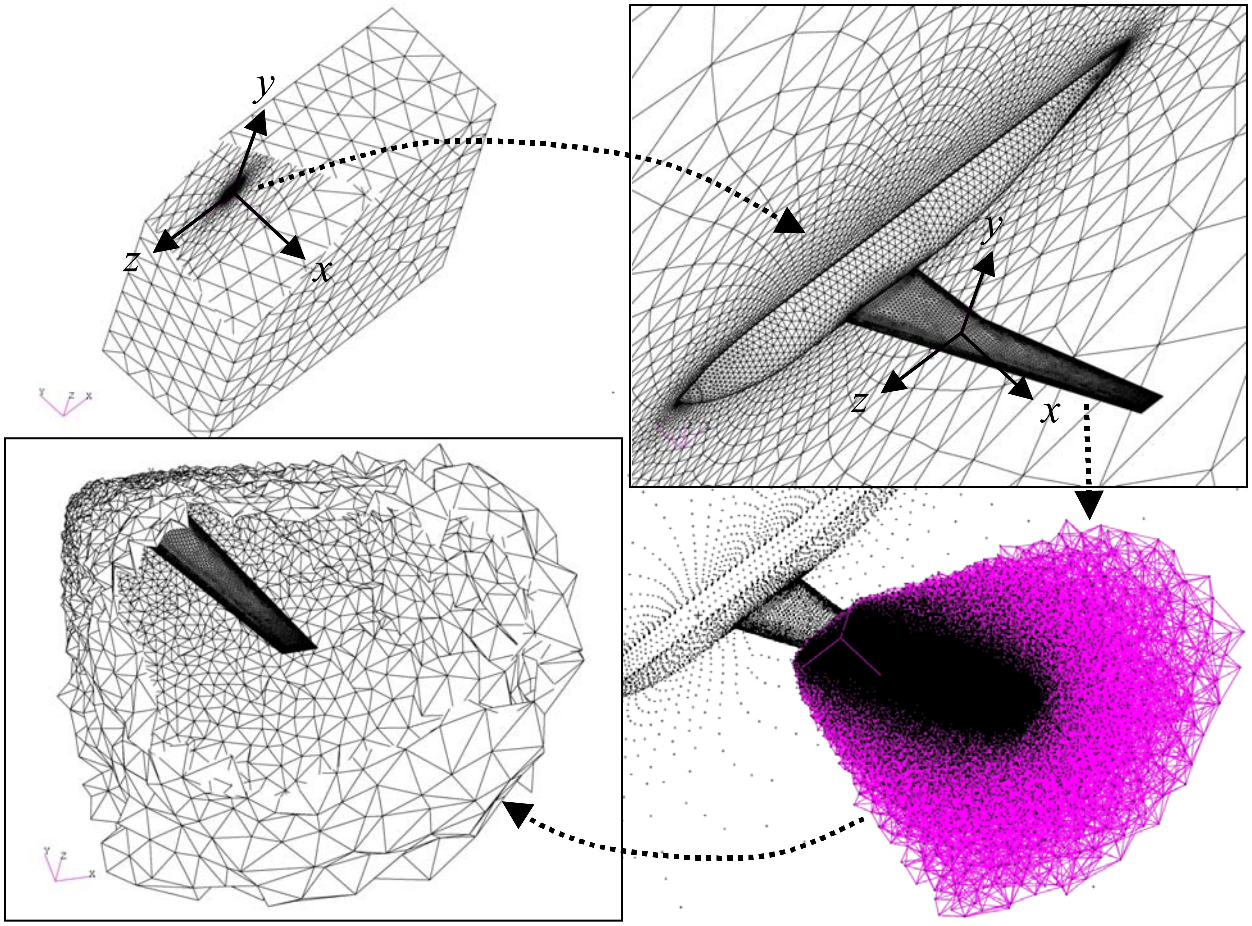


Figure 17: Complete 3-D CFD mesh and cutout portion to be deformed.

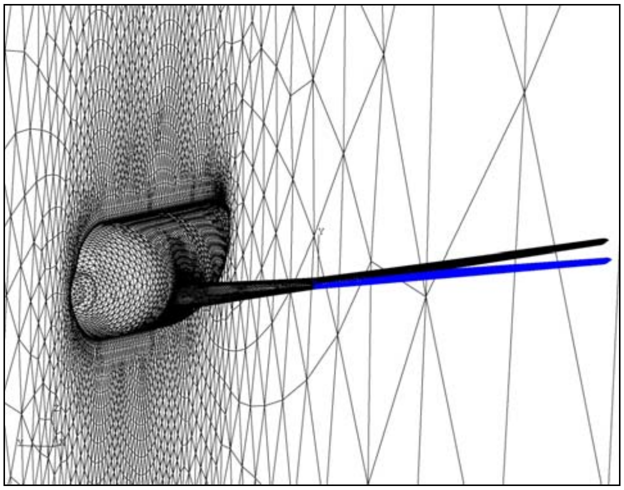


Figure 18: Outboard portion of wing rotated upward.

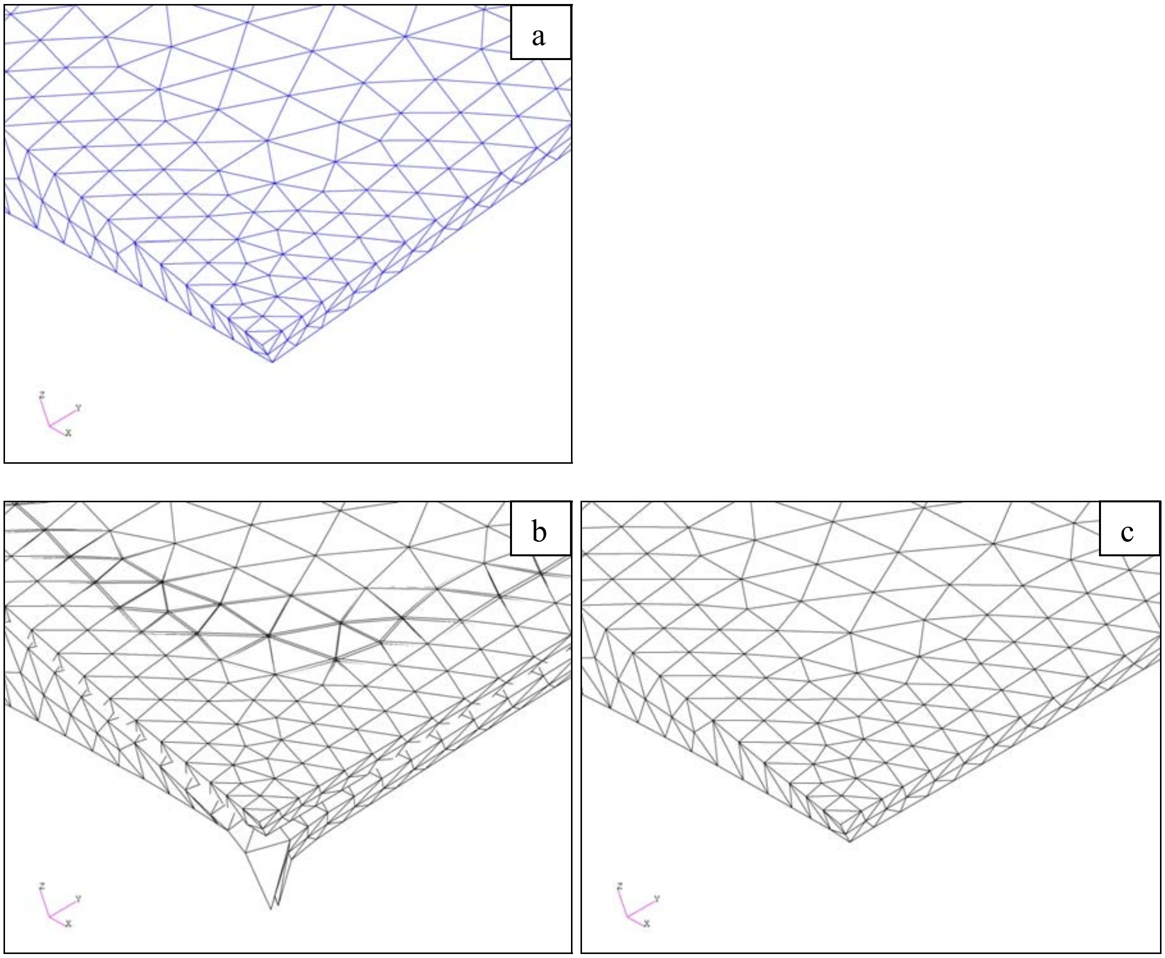


Figure 19: Examining first layer of elements. (Fig. 19a: Mesh prior to deformation. Fig. 19b: Mesh from first analysis. Fig. 19c: Mesh from second analysis.)

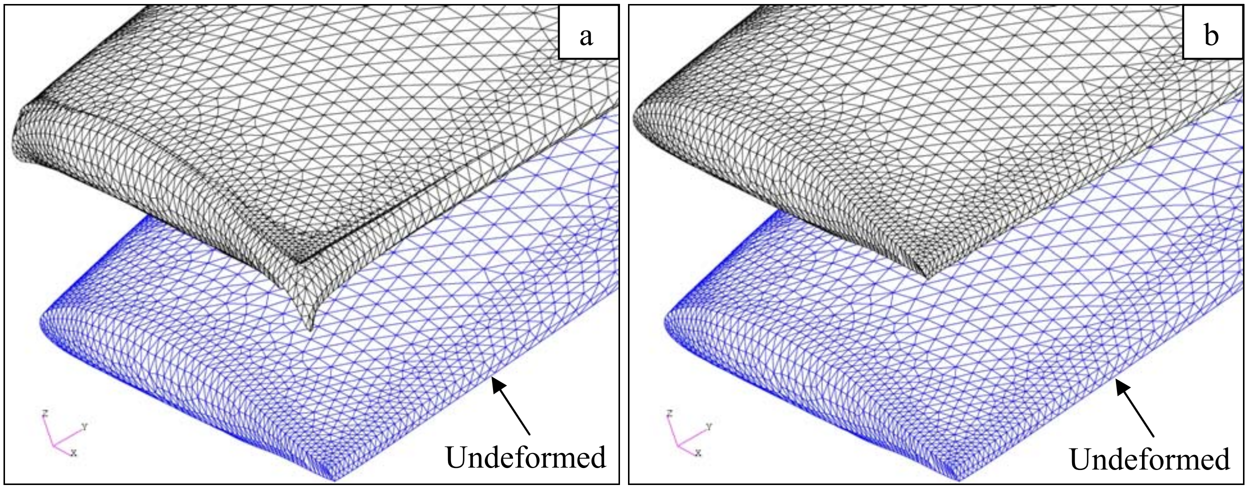


Figure 20: Examining fifth layer of elements. (Fig. 20a: Mesh from first analysis. Fig. 20b: Mesh from second analysis.)

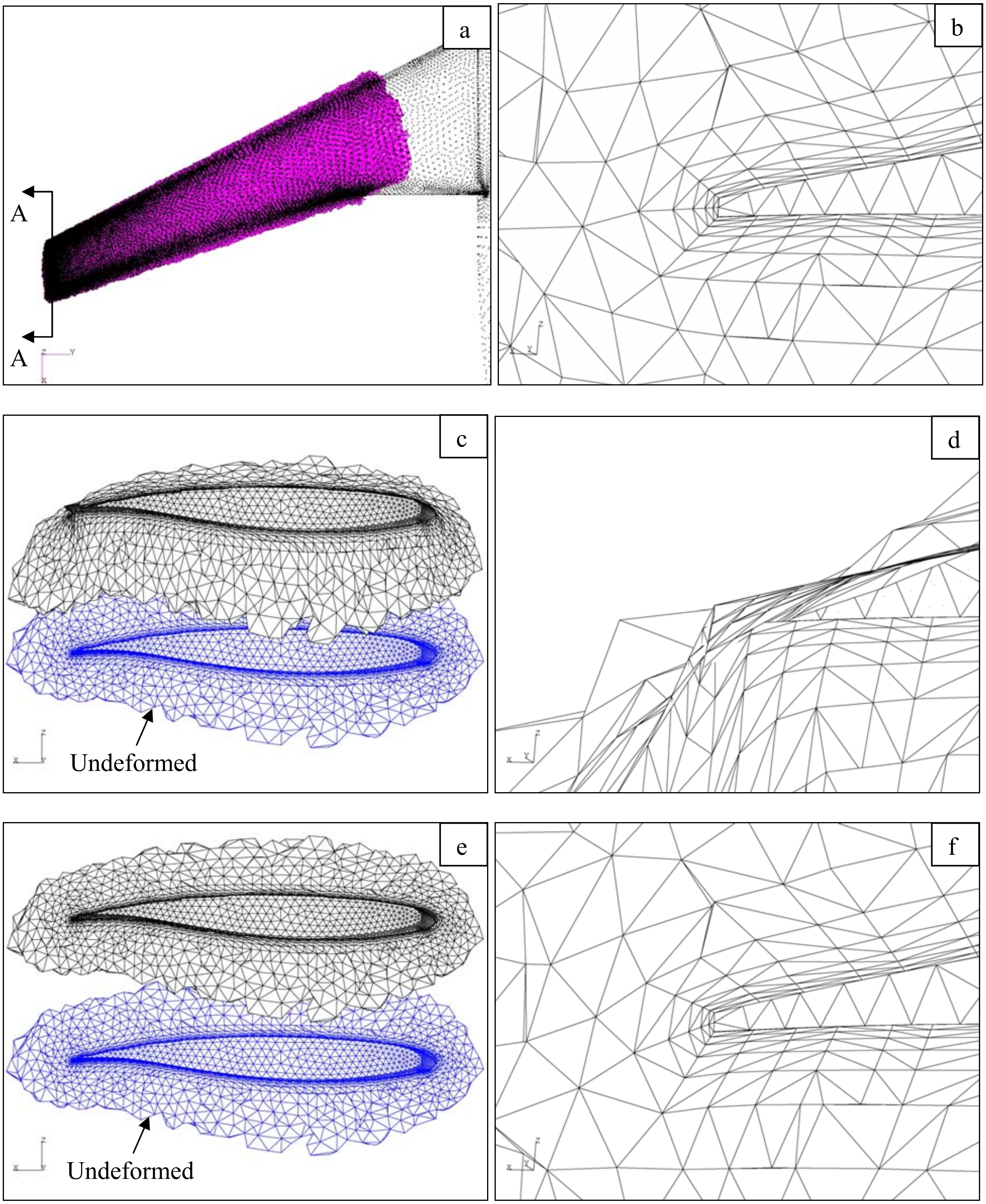


Figure 21: Examining ten layers of elements near wing tip. (Fig. 21a: Region examined defined by cross section view AA. Fig. 21b: Mesh around trailing edge prior to deformation. Fig. 21c: Mesh from first analysis. Fig. 21d: Mesh around trailing edge from first analysis. Fig. 21e: Mesh from second analysis. Fig. 21f: Mesh around trailing edge from second analysis.)

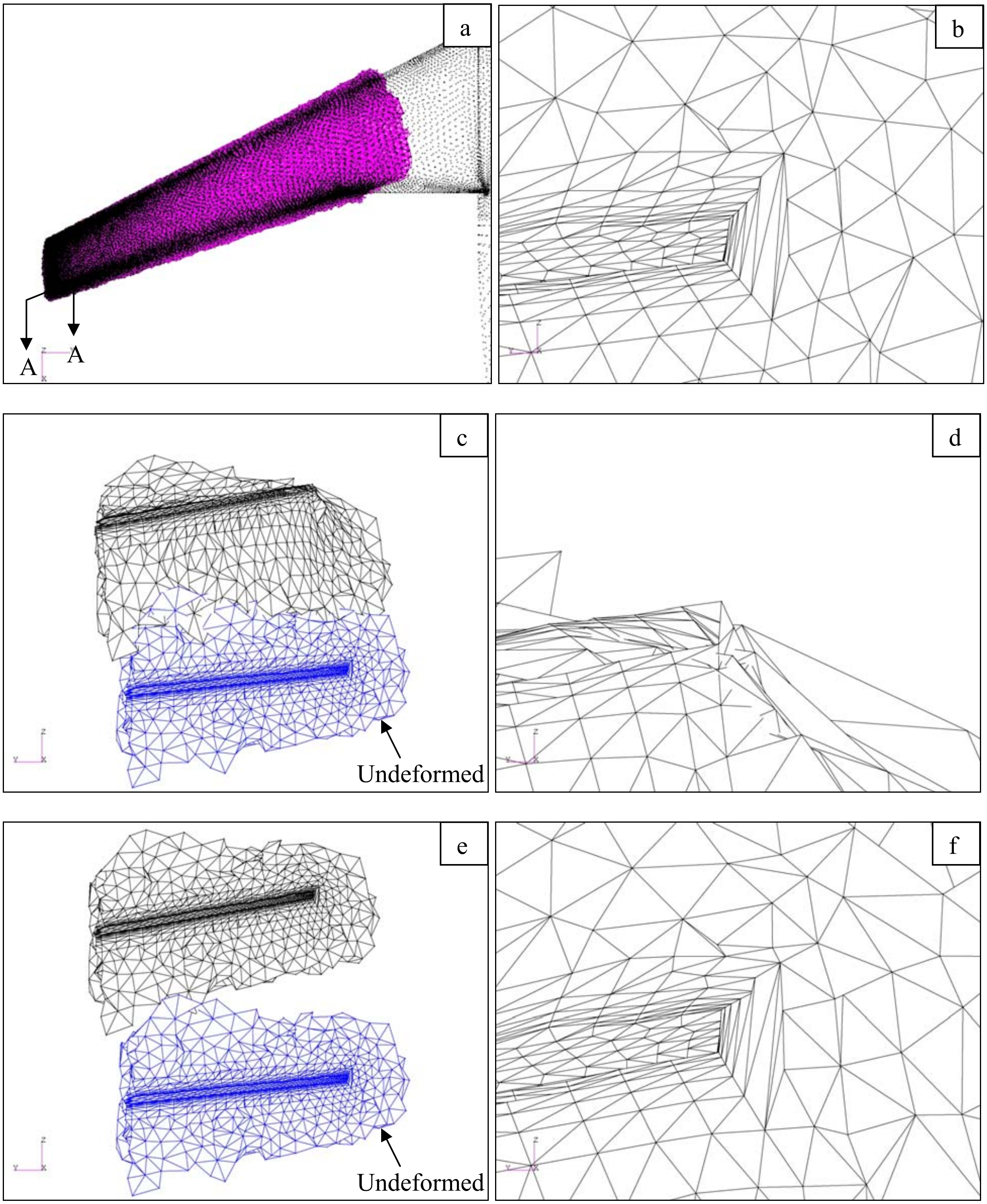


Figure 22: Examining ten layers of elements near trailing edge. (Fig. 22a: Region examined defined by cross section view AA. Fig. 22b: Mesh around wing tip prior to deformation. Fig. 22c: Mesh from first analysis. Fig. 22d: Mesh around wing tip from first analysis. Fig. 22e: Mesh from second analysis. Fig. 22f: Mesh around wing tip from second analysis.)