Drag Prediction on NASA CRM Using Automatic Hexahedra Grid Generation Method

Atsushi Hashimoto[1], Keiichi Murakami[2], Takashi Aoyama[3], Kazuomi Yamamoto[4], Mitsuhiro Murayama[5]
Japan Aerospace Exploration Agency (JAXA), Chofu, Tokyo, 182-8522, JAPAN

and

Paulus R. Lahur[6]
Research Center of Computational Mechanics, Inc. (RCCM), Shinagawa, Tokyo, 142-004, JAPAN

To shorten the time to simulate fluid flows, we develop a fast automatic grid generation tool, HexaGrid, that produces Cartesian grids with body-fitted layers. The objective of the present study is to apply automatic grid generation to the drag prediction of the NASA Common Research Model (CRM). First, we generate a grid following the DPW gridding guidelines as far as possible, and discuss the capabilities and limitations of the automatic method. Then, we validate the CFD results computed with the grid by comparison with other solvers and grid generators. The HexaGrid results agree well with the other results, the differences in predicted drag being less than 5 counts except at the stall angle. Additionally, we compare separated flows at the stall angle. The separation lines and C_p distribution are found to be greatly affected by grid topology.

[1] Researcher, Institute of Aeronautical Technology, AIAA Senior Member
[2] Senior Researcher, Institute of Aeronautical Technology, AIAA Senior Member
[3] Senior Researcher, Institute of Aeronautical Technology, AIAA Senior Member
[4] Senior Researcher, Institute of Aeronautical Technology, AIAA Senior Member
[5] Associate Senior Researcher, Institute of Aeronautical Technology, AIAA Senior Member
[6] Engineer, Technical Development Department, AIAA Senior Member

I. Introduction

Grid generation tends to be the most time consuming task in CFD procedures. Recently, unstructured grids have become popular in practical CFD applications, especially when model geometries are complex. Although unstructured grids can reduce the grid generation time compared to using structured grids, workload still remains a big issue and specialist skill is required for grid generation. The problem is that grid generation is not fully automatic. This is a barrier to an efficient CFD implementation, as a result shape optimization by CFD remains difficult since it is hard to use a batch process for grid generation.

Grid generation has become a process that connects CAD tools with CFD solvers. When CAD tools are used to generate CFD models, the surface geometry, typically stored as IGES or STL format data, must be clean. For example, surfaces should be watertight and gaps and overlaps should be removed before grid generation. However, it is not easy to clean up manufacturing CAD data, so the model is not always clean in practical CFD applications.

We are therefore researching a fast, fully automatic grid generation method that can be used with deficient geometry data. The final goal is to develop a grid generation program that can be used in a batch process by specifying a few control parameters and inputting a CAD data set. In this study, we use a method based on a Cartesian grid[1] in which most of the computational domain is filled with a Cartesian grid and a body-fitted layered grid is used near the model's surface. Using a Cartesian grid allows us to generate a high-quality grid in a short time, while the body-fitted layer grid is suitable for computing boundary layers.

We employ a method close to that proposed by Tchon et al.[2] and Wang et al.[3] A Cartesian grid is first created around a solid object, then a layer grid is generated from the boundary of Cartesian grid towards the object's surface by snapping. The snapping process makes the grid robust against low-quality geometry data. The surface data are not used directly in this method. Grid generation proceeds from outside the model towards its surface, and the grid wraps the surface. The result is an unstructured grid without oversets or cut cells which can be used by most unstructured grid CFD solvers.

Unfortunately, this automatic method is still immature and is rarely applied to drag prediction, which require high grid quality. Usually, grid generation is carried out by experienced technicians since specialist know-how such as knowledge of gridding guidelines is necessary, but the work is time consuming even for trained people. To the authors' knowledge, fully automatic grid generation has not yet been used on a wing-body type aircraft model. The objective of this study is to evaluate this automatic method for drag prediction. The benchmark problem of the 4th

Drag prediction workshop (DPW4)[4] is chosen for validation. At the DPW4 held in June 2009, a new model, the NASA common research model (NASA CRM)[5], was provided to give a benchmark for discussing CFD applications to modern civil aircraft.

Recently, we have improved the method's grid generation capability for viscous flow simulation and have used it in preliminary drag prediction studies on simple geometries such as the ONERA M6 (Ref. 6). In this paper, we apply the fully automatic grid generation method to a more practical model, the NASA CRM, and validate the results through comparison with other solvers and grid generators.

II. Automatic Grid Generation Method (HexaGrid)

This study uses the HexaGrid[6,7] software which implements our automatic grid generation method. HexaGrid automatically generates hexahedral grids around solid surfaces encoded in STL format that have been discretized into triangles. Virtually all CAD software can output surface geometry in STL format. The grid generation procedures are outlined below.

1. Cartesian Grid Generation

A Cartesian grid is generated by successive local refinement. This step starts with a single cell of a user-defined size that covers the whole computational domain. In three-dimensional space, each refinement step divides cells isotropically into eight child cells of equal size and shape. Initially, the grid is locally refined until the size of the cells intersecting the model's surface is smaller than a maximum allowable value set by user (Fig. 1(1)). The grid is then further refined until the sizes of cells intersecting surfaces with high curvature reach either a satisfactory level or a user-defined minimum allowable size (Fig. 1(2)). In addition, the grid size may also be controlled locally using the "Refinement Box" function of the HexaGrid graphical user interface (GUI) (Fig. 1(3)).

In this and later steps, searches for triangles that form the model's surface are made very frequently. To perform these searches efficiently, the surface triangles are stored in a data structure known as an Alternating Digital Tree (ADT)[8].

2. Removal of Cells near Solid Surface and Construction of "Quad Surface"

The purposes of this step are to create space for a layer grid around the model's surface and to form a base for layer grid generation. Cartesian grid cells that intersect the model and those around the model's surface are first removed. Note that the size of Cartesian grid cells removed is almost the same as the total thickness of layer grid. The exposed faces of the remaining Cartesian grid cells then form a "quad surface[7]" which becomes the base for layer grid generation at a later stage. The quad surface is then smoothed to ensure high quality of layer grid in the direction tangential to the model surface.

Before smoothing, hanging nodes are removed from the Cartesian grid by splitting appropriate cells into pyramids and tetrahedra. This is performed so that the final grid can be used by a wide range of solvers, many of which cannot handle hanging nodes. Despite this process, most of the grid elements (approximately 80%) are hexahedral.

To accelerate searches during this process, a data structure known as the Winged-Edge data structure[9] is used to store the faces, edges and nodes of the quad surface.

3. Snapping to Solid Surface

The quad surface is snapped to the model's solid surface by moving each node of the quad surface to the closest location on the solid surface. A grid surface snapped to a solid surface is shown in Fig. 2(a). Note that snapping always finds a unique location closest to the present position. This is a very important property, because it means that the method works even when there is a gap between the triangles that form the surface and when the triangles overlap or intersect each other.

4. Layer Grid Generation

A number of layer grid layers are constructed on the snapped surface, each layer having the same number of cells, as shown in Fig. 2(b). The user defines the thickness of the first grid layer and the expansion factor of thickness. Since the total thickness of the layer grid is already determined in step 2, the number of grid layers can thus be computed. The stacks of layers follow straight vectors emanating from grid nodes on the solid surface, where the direction of the vector is taken as the average of the normal vectors of the grid faces on the solid surface surrounding a grid node.

5. Quality Improvement

The grid quality, which is measured mainly in terms of face flatness and convexity, is improved by smoothing. This ensures that the Cartesian grid blends with the layer grid around their interface. In this region, cells from both grids gradually change shape and size to match neighboring cells (Fig. 2(c)).

III. Flow Computational Method (TAS-code)

As an unstructured flow solver, the TAS (Tohoku university Aerodynamic Simulation) code[10] is used in this study. This is a well-validated code which is used in a variety of aerospace applications[11]. In TAS, full Navier-Stokes equations are solved by a cell-vertex finite volume method. The HLLEW (Harten-Lax-van Leer-Einfeldt-Wada) method[12] is used for the numerical flux computations, and the implicit LU-SGS (Lower/Upper Symmetric Gauss-Seidel) method is used for time integration. Second-order spatial accuracy is realized by a linear reconstruction of the primitive variables with Venkatakrishnan's limiter[13] and the Unstructured MUSCL-scheme[14] (U-MUSCL). For the turbulence model, the Spalart-Allmaras model[15] is used, and the boundary layer is fully turbulent. The equations of the turbulence model are also solved using the second-order scheme.

IV. Computational Grids

A) Model

The model is the NASA Common Research Model (NASA CRM)[5], shown in Fig. 3. This is a simple wing/body/tail configuration with a wing-body fairing. A supercritical wing is employed as the main wing. The fuselage is representative of a wide/body commercial transport aircraft. Four configurations are provided by DPW4; three have the horizontal tail wings set at different angles (i_H=-2°, 0°, 2°), and the other is a tailless model. In this paper, we use the model with a horizontal tail wing angle of i_H=0°. Reference values of this model are shown in Table 1. CAD data are provided in IGES files that can be downloaded from the DPW4 website[4].

B) Grids generated with HexaGrid

Three different grids (coarse, medium, and fine) are generated with HexaGrid. The medium grid is shown in Fig. 4. In addition, close-up views of the wing root, wing tip, and horizontal tail are shown in Figs. 5–7, with coarse, medium, and fine grids shown for each part. The grids were generated with the input parameters shown in Table 2.

As mentioned in Section II, we control the Cartesian grid using two parameters, maximum and minimum grid size on the model surface. For the coarse grid, we set the same value of 5 inches as the max. and min. sizes, and a uniform-size grid was generated (Figs. 5(a), 6(a), 7(a)). For the medium grid, we set the max. size to 5 in and the min. size to 1.25 in (=$5/2^2$). The grid was refined until the grid size in areas of large curvature reached the minimum grid size. In this case, the grid near the leading and trailing edges was refined two more times (Figs. 5(b), 6(b), 7(b)), becoming first a half then a quarter of the initial size. For the fine grid, the minimum grid size was set to be half that of the medium one. The grid near the leading and trailing edges was refined once more (Figs. 5(c), 6(c), 7(c)). For the layer grids, we employed the same expansion ratio and first layer thickness for all three grids (Table 2).

Figure 8 shows the grids of the main wing at the cross-section at η=0.50, where η is a ratio of distance between the body's centerline and the cross-section to the span. The grid size around the airfoil is uniform in the case of the coarse grid, whereas the grid is refined more around the leading and trailing edges in the medium and fine grids. The total thickness of the layer grid depends on the neighboring Cartesian grid size. Hence, the layer grid is relatively thin around the trailing edge in the medium and fine grids (Fig. 9(b), (c)).

Table 3 shows the grid generation results. The numbers of generated grid cells are approximately 3.2 million, 11 million, and 37 million for the coarse, medium, and fine grids, respectively. The numbers of nodes and cells are almost the same since hexahedral elements are predominant. The grids were generated by a Fujitsu SPARC Enterprise M9000 of the JAXA Supercomputer System (JSS). As an example, the generation time of the medium grid (serial execution) was 120 minutes, which is substantially faster than using manual methods as we will see later.

Gridding guidelines are provided by DPW4[9]. Although we followed as many of the guidelines as possible, some of them had to be neglected due to restrictions of HexaGrid. Details are as follows.

- We used the values of thickness of first layer grid and growth rate in the layer defined in the guidelines. These can be controlled in HexaGrid by input parameters for layer grid generation.
- Regarding the outer boundary, HexaGrid can locate the boundary anywhere defined by the user. However, the domain shape has to be cubic, otherwise anisotropic grids will be generated.
- We could not maintain the same stretching factors and topologies for all three grids since HexaGrid generates grids by successive local refinement.

- The guidelines specify the chord-wise spacing at the leading and trailing edges for the main and tail wings as less than 0.1% local chord for the medium grid. Hence, the grid size must vary according to the local chord length. However, HexaGrid cannot control grid size in this way.
- The grid is rather coarse near the leading edge to keep the number of grid cells reasonable. For example, 0.1% of local chord is 0.11–0.47 in for the main wing and 0.09–0.22 in for the tail wing. However, the actual grid size is uniformly 1.25 in for the medium grid. The grid size near the wing tip is approximately ten times coarser than that required by the gridding guidelines.
- A minimum of 12 cells are required across the base of the trailing edge for the medium grid. We neglected this requirement due to a HexaGrid limitation. As shown in Fig. 9, 5–6 nodes are located across the base at the cross-section η=0.50 for the three grids. However, it is difficult to allocate more grid cells across the base since the total thickness of layer grid becomes thin when the fine Cartesian grid is generated near the trailing edge.

Additionally, HexaGrid has a feature capturing function[7] which was employed for the three grids. As shown in Figs. 5 and 7, the feature line of wing-body juncture is clearly captured except in the case of the coarse grid. Feature capture becomes difficult when the grid is coarse because the distances between feature lines and grid nodes to be moved become relatively large. However, this problem can be overcome by using a sufficiently fine grid.

HexaGrid can automatically generate grids using a small number of parameters, shown in Table 2. Therefore, even CFD beginners can generate grids easily and the training time needed is short. Moreover, since the generation process is fully automatic, using the same parameter values results in the same grid being generated, so grid generation is less dependent on user experience.

C) Grid generated with MEGG3D

We also generated a tetrahedron-based grid for comparison. A hybrid unstructured grid was generated using an unstructured surface/volume grid generator, MEGG3D[16,17]. First, ridges (feature lines) of the STL data are set manually using the MEGG3D GUI. Then, surface grids are triangulated by the direct advancing front method based on the ridges and STL data[16]. After that, isotropic tetrahedral volume grids are generated using the method of

Delaunay tetrahedral meshing. Finally, prismatic layers are added on the non-slip walls[17]. This grid generator has been well-validated and is used for many applications such as drag prediction[11].

A medium grid generated by MEGG3D is shown in Fig. 11. The grid is well clustered near the leading and trailing edges compared with that generated by HexaGrid. However, it is difficult to generate a fine grid across the trailing edge base since the grid is basically isotropic. The MEGG3D grid in the middle of wing is relatively coarse, whereas the HexaGrid grid is uniformly fine (Figs. 4(b), 10(b)). The grid information is shown in Table 4. We spent about six hours setting the ridge lines and about six hours generating the spatial grid, which include the time of trial and error taken to satisfy the gridding guidelines. The surface grids were generated in a few minutes. The flow solver of the TAS code was used for this grid.

D) Grid generated with Gridgen®

A multi-block structured grid was also generated for comparison using a commercial program, Gridgen®. A medium grid generated with Gridgen is shown in Fig. 11. The grid is well clustered near the leading and trailing edges compared with those generated with HexaGrid (Fig. 4) and MEGG3D (Fig. 10). The grid is fine in the chord-wise direction but coarse in the span-wise direction. The grid information is shown in Table 4. We spent about five days generating the grid.

UPACS[18] was used as the flow solver for this grid. This flow solver is based on a cell-centered finite volume method. In this study, the second-order scheme of the Roe’s flux difference splitting for convection terms is used with MUSCL extrapolation and van Albada’s limiter. The viscous terms are discretized using a scheme based on 2nd-order central differences. Time integration is carried out using the MFGS implicit method. The Spalart-Allmaras model[15] is used as the turbulence model.

V. Results

In this section, we compare the following three cases:

- HexaGrid+TAS: Hexahedral unstructured grid generated by HexaGrid and flow solved with TAS-code.
- MEGG3D+TAS: Tetrahedral unstructured grid generated by MEGG3D and flow solved with TAS-code.
- Gridgen+UPACS: Multi-block structured grid generated by Gridgen and flow solved with UPACS.

A) Grid convergence

Drag coefficients are computed at C_L=0.500 for the coarse, medium, and fine grids. The free stream Mach number is 0.85 and the Reynolds number based on the reference length, C_{ref}, is 5×10^6.

Contours of C_p computed with HexaGrid and the TAS code are shown in Fig. 12. The grid is medium and the angle of attack is 2.388°. The pressure at the wing-body juncture is smooth since the feature line is successfully captured.

Drag coefficients for the three grids are shown in Fig. 13, where HexaGrid+TAS is compared with MEGG3D+TAS and Gridgen+UPACS. Total drag is the summation of pressure drag and friction drag. The horizontal axis is a grid index, 1/(number of grid cells)$^{(2/3)}$. The HexaGrid result shows good grid convergence and the predicted drag coefficients are between the results of MEGG3D+TAS and Gridgen+UPACS (Fig. 13(a)). In other words, drag values predicted using automatic grid generation has comparable accuracy to those given by manual grid generation. The pressure and friction drags of MEGG3D+TAS and Gridgen+UPACS have good convergence tendencies, whereas those of HexaGrid show different trends with finer grid (Fig. 13(b)(c)). One reason is the differences between the three grids used in this study. The MEGG3D and UPACS grids have the same stretching factors and topology at all three grid densities. However, HexaGrid cannot generate such a family of grids, the medium and fine grids being locally refined near the leading and trailing edges (Figs. 5–7).

B) Comparison of C_p and C_f distributions

Here we compare the medium grid results at C_L=0.500, which are the same results as used in the grid convergence study. The C_p and C_f distributions at η=0.20, 0.50, and 0.95 of the main wing and at η=0.30 of the tail wing are shown in Figs. 14–17, and the cross-sections are illustrated in Fig. 18. These cross-sections are a part of the data required by DPW4, and their detailed definitions are described in Ref. 4. Generally, the HexaGrid+TAS results agree well with the other results except for the cross-section near wing tip, although the grid generated by HexaGrid is coarser near the leading edge than those generated by MEGG3D and Gridgen. The shock wave is well captured in the case of HexaGrid (Fig. 15(a)), the generated grid being isotropic and relatively fine compared with the others (Fig. 4(b), 10(b), 11(b)). The friction coefficient of HexaGrid becomes less than the others behind the shock wave location, since the friction is decreased by the large adverse pressure gradient at the shock wave location (Fig. 15(b)).

The difference of shock strength and the related friction cause differences in the friction drag results. In fact, the friction drag obtained with HexaGrid is less than the others, as shown in Fig. 13.

Major differences are observed in the middle of the wing section η=0.95 (Fig. 16). In order to see the effect of grid topology, C_p contours and C_p contours with grid are shown in Figs. 19 and 20, respectively. The grid generated by HexaGrid is isotropic and uniformly fine, and its C_p distribution seems to be the best among the three results (Figs. 19(a), 20(a)). The MEGG3D grid is fine near the leading and trailing edges, but coarse in the middle of wing. As a result, the C_p distribution is smeared due to the coarse grid (Figs. 19(b), 20(b)). The Gridgen grid is also fine near the leading and trailing edges but is anisotropic and relatively coarse in the span-wise direction. Therefore, the pressure distribution is smeared in the span-wise direction (Fig. 19(c), 20(c)). This comparison demonstrates the merit of using a uniformly fine grid.

The C_p distributions of the three grids are almost the same at the tail wing (Fig. 17(a)). However, the friction of HexaGrid is less than the others, probably due to the coarse grid at the leading edge (Fig. 17(b)). This deficiency may be covered by local refinement using a refinement box (Fig. 1).

C) C_L-α, C_L-C_D polar curve

Figures 21 and 22 show the C_L-α graph and C_L-C_D polar curve, respectively, where the three results using medium grids are compared. The result of HexaGrid+TAS agrees well with those of MEGG3D+TAS and Gridgen+UPACS except at the attack angle of 4°. Figure 22 shows the C_L-C_D polar curve. The result of HexaGrid+TAS is close to that of Gridgen+UPACS except for the attack angle of 4°, where the difference is less than 5 counts.

The flow is separated at the attack angle of 4°. The separation line differs greatly between the three grids as shown in Figs. 23 and 24. The HexaGrid+TAS result shows the largest separation and its flow pattern is close to the Gridgen+UPACS result. This difference will be revisited after planned wind-tunnel measurements have been accomplished.

VI. Conclusions

In this paper, we applied an automatic hexahedral grid generation program, HexaGrid, to a more practical model than in other automatic grid generation studies, NASA CRM, and validated the results by comparison with other solvers and grid generators.

We compared the generated grid with the DPW gridding guidelines and discussed the capabilities and limitations of HexaGrid. HexaGrid can generate layer grids with the desired first-layer thickness and growth rate. Furthermore, it can capture the feature lines of wing-body junctures clearly. However, it is difficult to generate a family of grids for grid convergence studies, where grids of different densities are required to have the same stretching factors and topology. Additionally, the grid is rather coarse near the leading edge to keep the number of grid cells reasonable.

Despite such deficiencies, the lift and drag predicted using this automatic grid generator agree well with those using manual methods, the difference of predicted drag being less than 5 counts except at the wing stall angle. The pressure and friction distributions obtained with HexaGrid are a little less accurate near the leading edge due to the grid coarseness. However, the shock wave in the middle of wing is well captured with HexaGrid. The friction coefficient of HexaGrid becomes less than the others behind the shock wave location due to the difference of shock strength. This is one of the reasons for the lower computed friction drag. Additionally, we compared the separated flows at an attack angle of 4°. The separation lines and C_p distributions are greatly affected by the grid topologies. These differences will be re-examined after experimental data has been gathered.

HexaGrid has been validated through comparison with other results. This automatic method can quickly generate grids that are comparable to grids produced by manual methods, with great time savings. HexaGrid becomes a powerful tool to accelerate CFD processes.

Acknowledgement

The authors would like to thank Mr. Hideki Kitajima and Mr. Masafumi Yamamoto of Research Center of Computational Mechanics, Inc., Mr. Taisuke Nanbu of Waseda University, Mr. Kentaro Tanaka and Mr. Tohru Hirai of Ryoyu Systems Co., Ltd. for their support of the sequence of CFD processing.

References

[1]Aftosmis, M.J., "Solution Adaptive Cartesian Grid Methods for Aerodynamic Flows with Complex Geometries," *VKI Lecture Series*, 1997-02, 1997.

[2]Tchon, K.F., Hirsch, C., and Schneiders, R., “Octree-based Hexahedral Mesh Generation for Viscous Flow Simulations,” AIAA paper 97-1980, 1997.

[3]Wang, Z.J. and Chen, R.F., “Anisotropic Solution-Adaptive Viscous Cartesian Grid Method for Turbulent Flow Simulations,” AIAA Journal, Vol. 40, No. 10, 2002, pp. 1969-1978.

[4]http://aaac.larc.nasa.gov/tsab/cfdlarc/aiaa-dpw/

[5]Vassberg, J.C., DeHaan, M.A., Rivers, S.M., Wahls, R.A., “Development of a Common Research Model for Applied CFD Validation Studies,” AIAA paper 2008-6919, 2008.

[6]Hashimoto, A., Murakami, K., Aoyama, T., Lahur P., “Lift and Drag Prediction Using Automatic Hexahedra Grid Generation Method,” AIAA paper 2009-1365, 2009.

[7]Lahur, P. R., “Automatic Hexahedra Grid Generation Method for Component-based Surface Geometry,” AIAA paper 2005-5242, 2005.

[8]Bonet, J. and Peraire, J., “An Alternating Digital Tree (ADT) Algorithm for 3D Geometric Searching and Intersection Problems,” Int. J. Numer. Meth. Eng., Vol 31, 1991, pp. 1-17.

[9]Baumgart, B.G., “Winged-edge Polyhedron Representation for Computer Vision,” National Computer Conference, 1975. Also available from: http://www.baumgart.org/winged-edge/winged-edge.html

[10]Nakahashi, K., Ito, Y., and Togashi, F, “Some challenges of realistic flow simulations by unstructured grid CFD”, *International Journal for Numerical Methods in Fluids*, Vol.43, 2003, pp.769-783.

[11]Murayama, M. and Yamamoto, K., "Comparison Study of Drag Prediction by Structured and Unstructured Mesh Method," *Journal of Aircraft*, Vol. 45, No. 3 (2008), pp. 799-822.

[12]Obayashi, S. and Guruswamy, G. P., “Convergence Acceleration of an Aeroelastic Navier-Stokes Solver,” *AIAA Journal*, Vol. 33, No. 6, 1995, pp. 1134-1141.

[13]Venkatakrishnan, V., “Convergence to Steady State Solutions of the Euler Equations on Unstructured Grids with Limiters,” *Journal of Computational Physics*, Vol.118, 1995, pp.120-130.

[14]Burg, C., “Higher Order Variable Extrapolation For Unstructured Finite Volume RANS Flow Solvers,” AIAA Paper 2005-4999, 2005.

[15]Spalart, P. R. and Allmaras, S. R., “A One-Equation Turbulence Model for Aerodynamic Flows,” AIAA Paper 92-0439, 1992.

[16]Ito, Y. and Nakahashi, K., "Direct Surface Triangulation Using Stereolithography Data," AIAA Journal, Vol. 40, No. 3, March 2002, pp. 490-496.

[17]Ito, Y., Shih, A. M., Soni, B. K. and Nakahashi, K., "Multiple Marching Direction Approach to Generate High Quality Hybrid Meshes," AIAA Journal, Vol. 45, No. 1, January 2007, pp. 162-167.

[18]Takaki, R., Yamamoto, K., Yamane, T., Enomoto, S. and Mukai, J., "The Development of the UPACS CFD Environment," High Performance Computing, Proc. of ISHPC 2003, Springer, pp. 307-319, 2003.

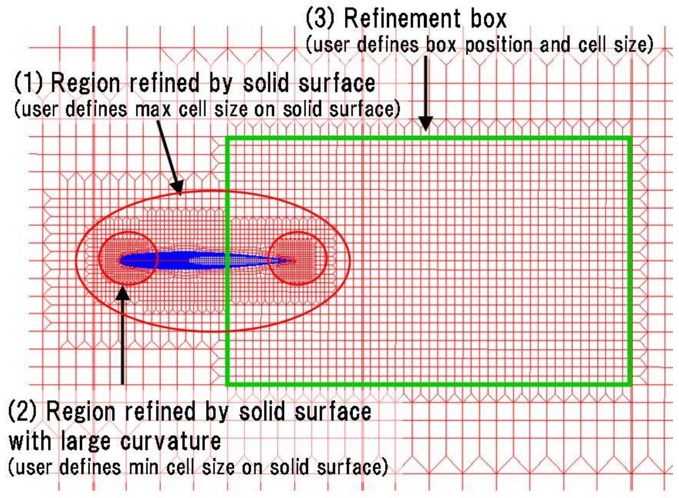
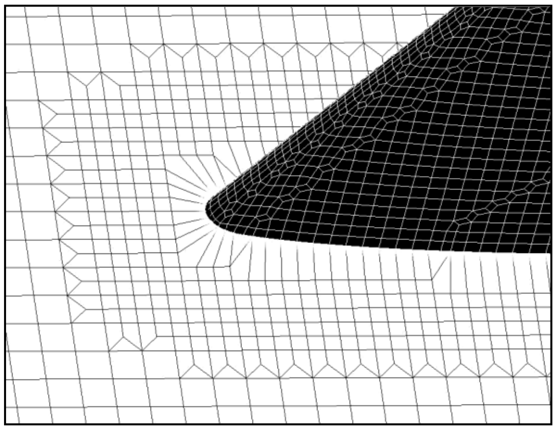
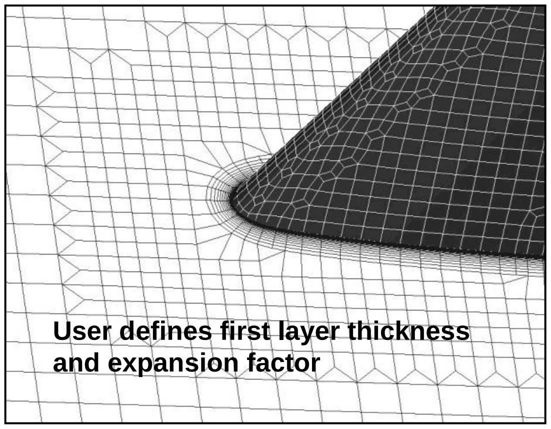

(3) Refinement box
(user defines box position and cell size)
(1) Region refined by solid surface
(user defines max cell size on solid surface)
(2) Region refined by solid surface
with large curvature
(user defines min cell size on solid surface)

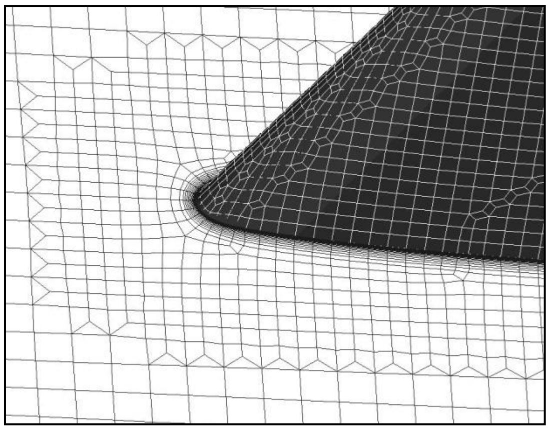
Figure 1 Refinement process of Cartesian grid



(a) Snapping to solid surface



(b) Layer layer generation



(c) Grid smoothing

Figure 2 Grid generation algorism

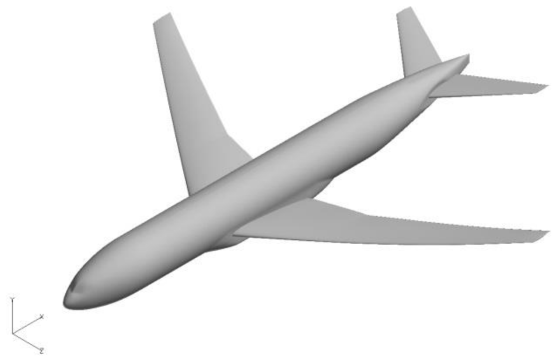
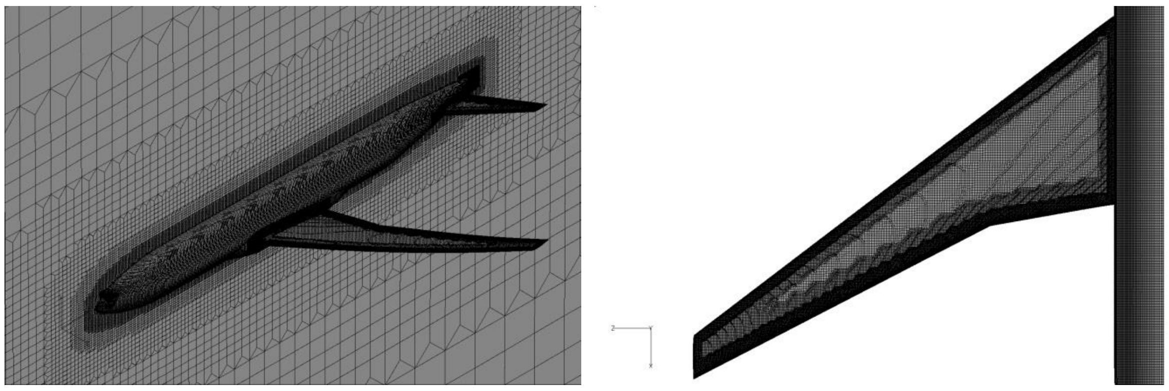


Figure 3 NASA Common Research Model (wing/body/tail (i_H=0°))

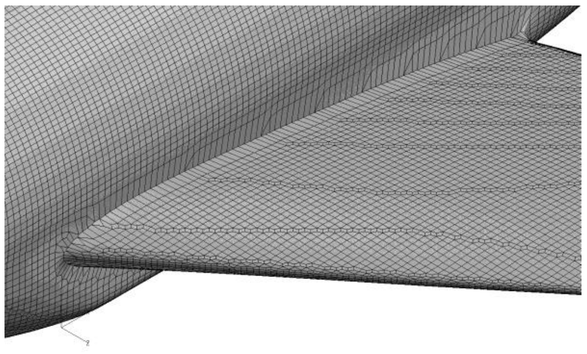
Table 1 Reference geometry

S_{ref} (reference area)	594,720 in^2
C_{ref} (reference chord length)	275.80 in
X_{ref}, Y_{ref}, Z_{ref} (moment reference center)	1325.90 in, 0 in, 177.95 in

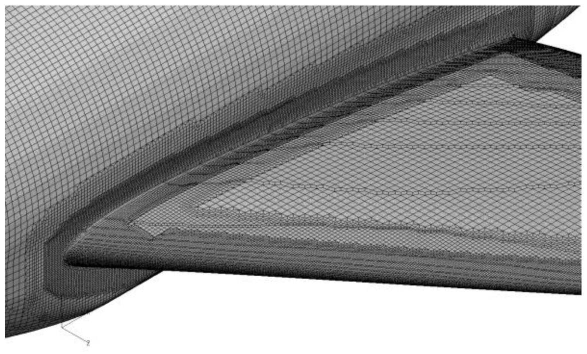


(a) Overall view **(b) Upper surface of main wing**

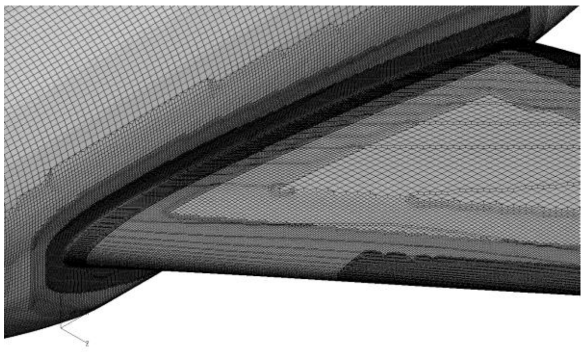
Figure 4 Medium grid generated with HexaGrid



(a) Coarse grid

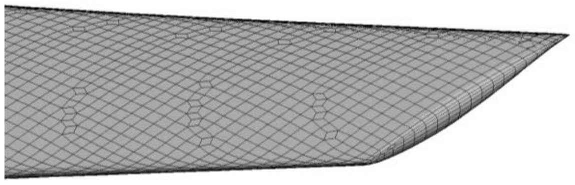


(b) Medium grid

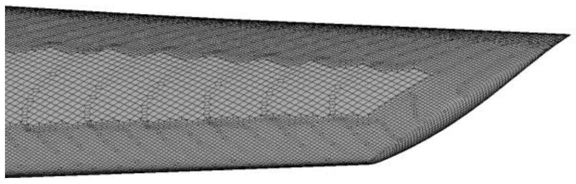


(c) Fine grid

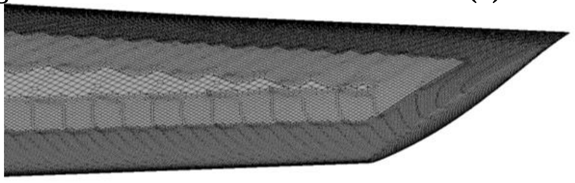
Figure 5 Grid around wing root



(a) Coarse grid

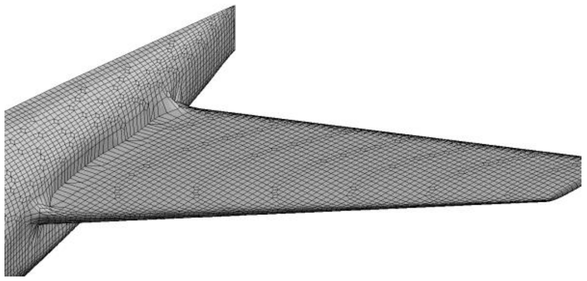


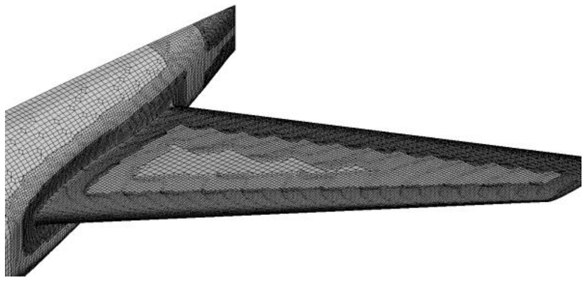
(b) Medium grid



(c) Fine grid

Figure 6 Grid around wing tip


(a) Coarse grid


(b) Medium grid

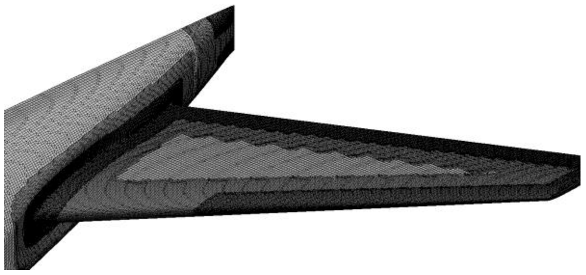

(c) Fine grid

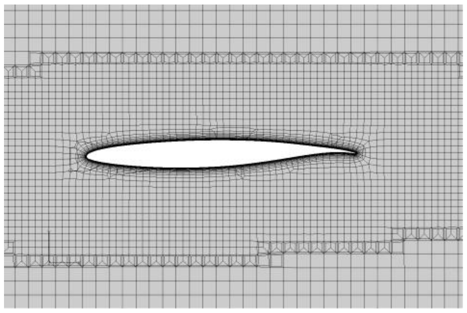
Figure 7 Grid around horizontal tail

Table 2 Grid generation settings

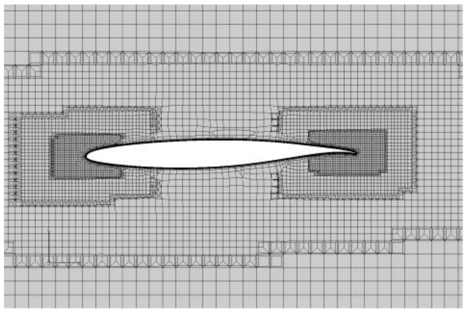
Setting parameter	Coarse	Medium	Fine
Max cell size on solid surface	5 in	5 in	5 in
Min cell size on solid surface	5 in	1.25 in (=5/2^2)	0.625 in (=5/2^3)
Expansion ratio of layers	1.25	1.25	1.25
Thickness of first layer	9.85E-4 in	9.85E-4 in	9.85E-4 in
Cubic domain size	100 C_{ref}	100 C_{ref}	100 C_{ref}

Table 3 Grid generation results

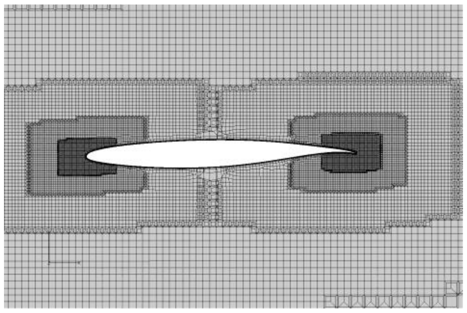
	Coarse	Medium	Fine
Cartesian grid finest level	13	15	16
Number of layers	35	29	26
Node count	3,213,783	11,055,602	36,601,899
Cell count	3,644,942	12,654,764	41,630,191
Boundary node count	105,686	295,394	757,593
Boundary face count	106,272	297,697	762,131
Layer cell count	1,932,525	7,145,542	17,752,826
Generation Time (JSS)	20 min	120 min	380 min



(a) Coarse grid

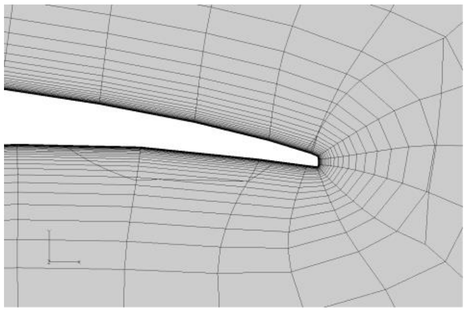


(b) Medium grid

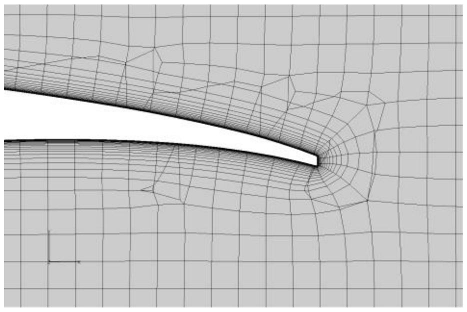


(c) Fine grid

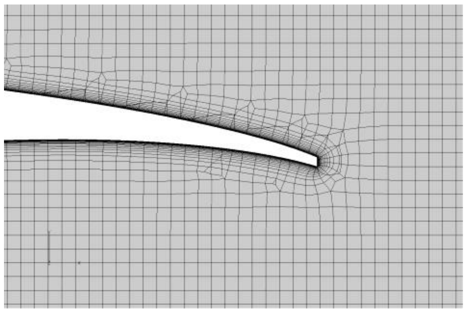
Figure 8 Main wing grid at the cross-section of η=0.50



(a) Coarse grid

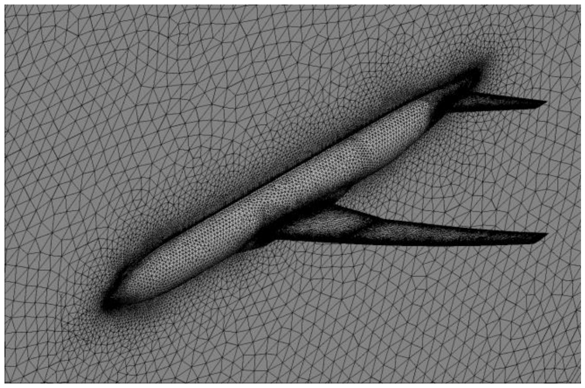


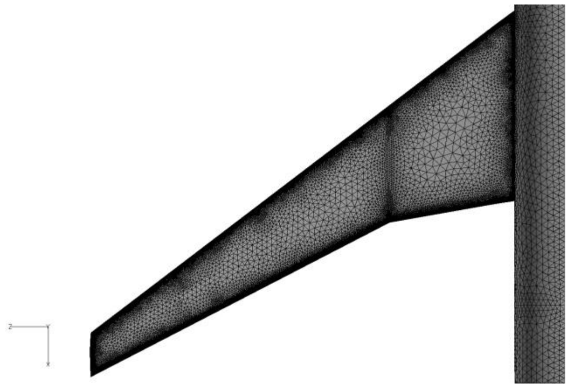
(b) Medium grid



(c) Fine grid

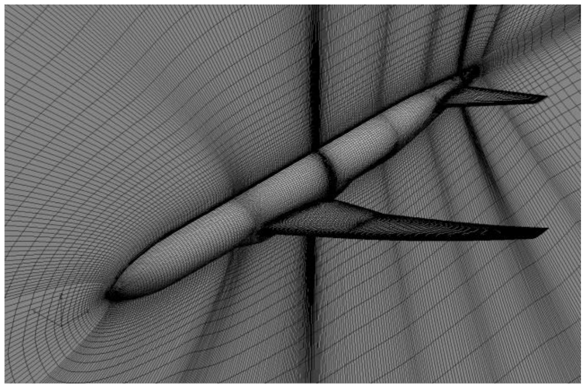
Figure 9 Main wing grid near trailing edge at the cross-section of η=0.50

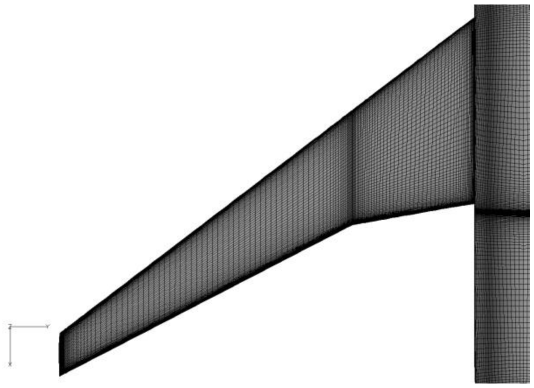




(a) Overall view (b) Upper surface of main wing

Figure 10 Medium grid generated with MEGG3D



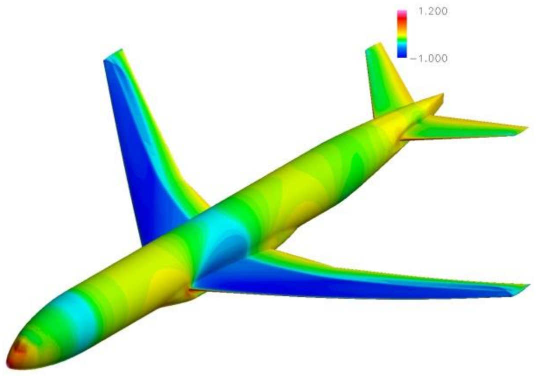


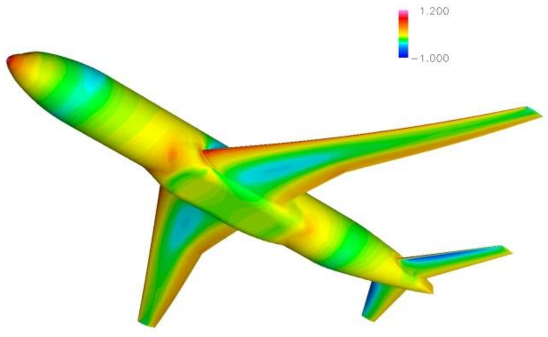
(a) Overall view (b) Upper surface of main wing

Figure 11 Medium grid generated with Gridgen

Table 4 Grid information (Medium grid)

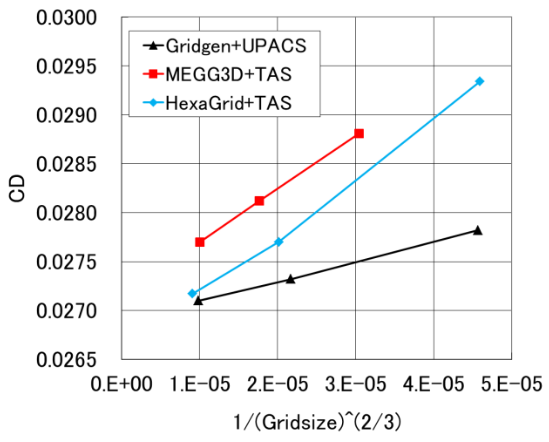
	MEGG3D	Gridgen
Zones	1	283
Number of prism layers	34	40
Node count	13,490,678	9,903,509
Cell count	36,250,881	9,006,808
Boundary node count	369,541	294,408
Boundary face count	798,336	275,608



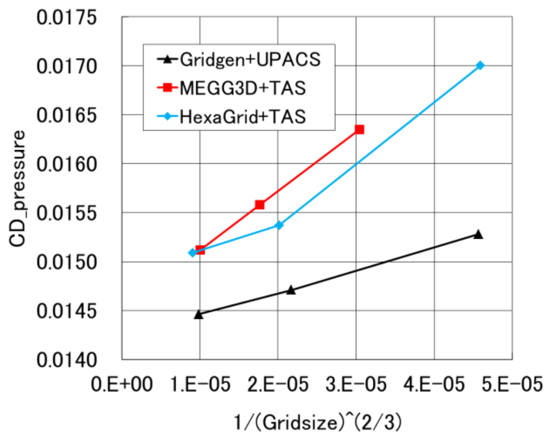


(a) Upper surface (b) Lower surface

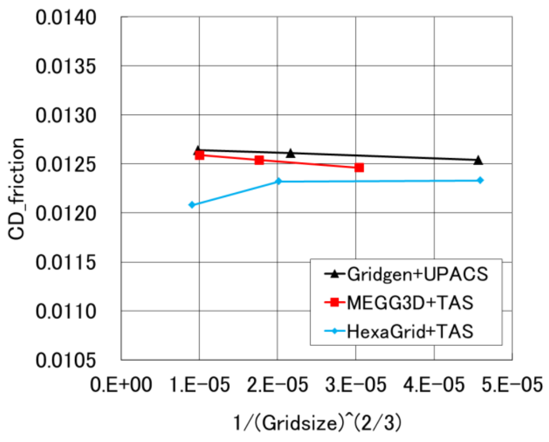
Figure 12 C_p contours (HexaGrid+TAS, medium grid, C_L=0.500)



(a) Total drag

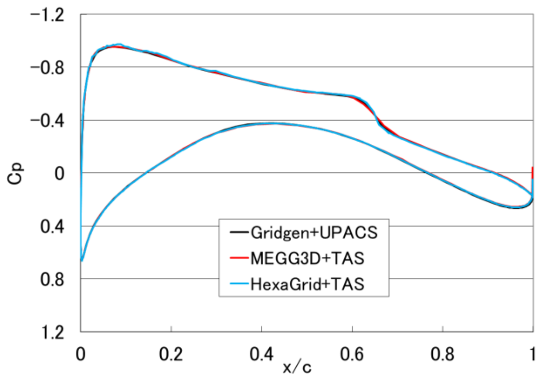


(b) Pressure drag

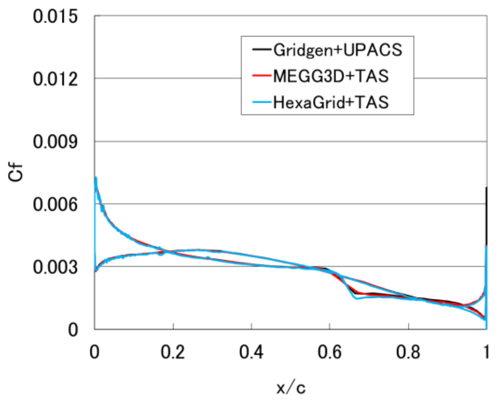


(c) Friction drag

Figure 13 Comparison of grid convergence

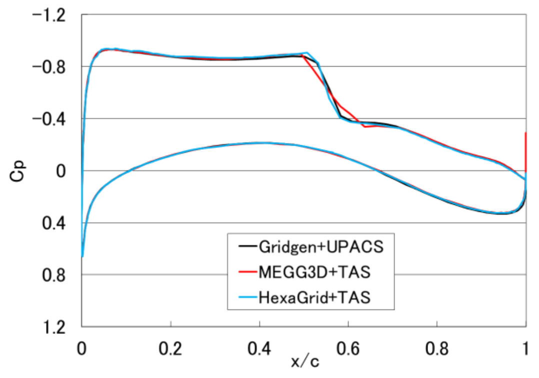


(a) C_p

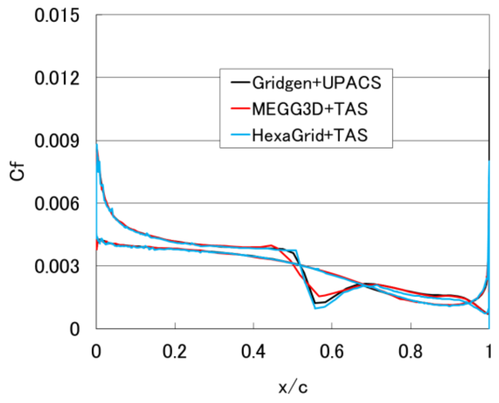


(b) Cf

Figure 14 Main wing C_p and C_f distributions at the cross-section of η=0.20

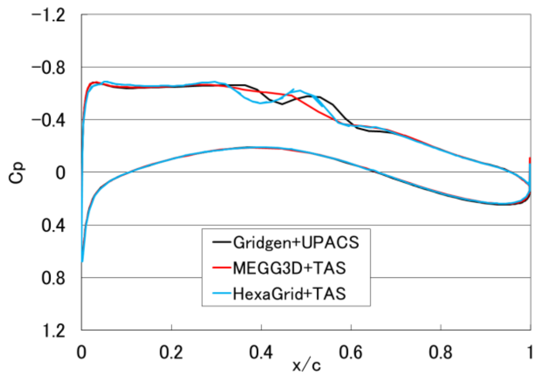


(a) C_p

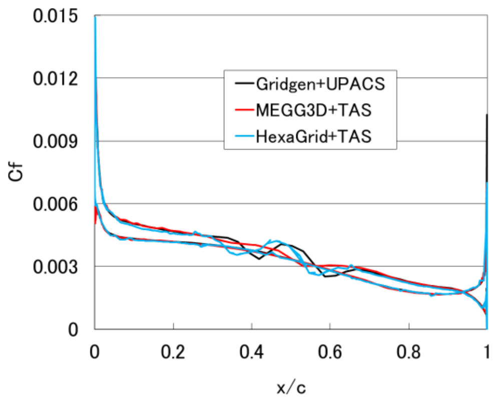


(b) Cf

Figure 15 Main wing C_p and C_f distributions at the cross-section of η=0.50

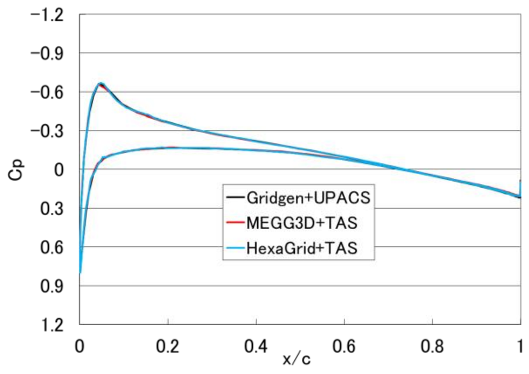


(a) C_p

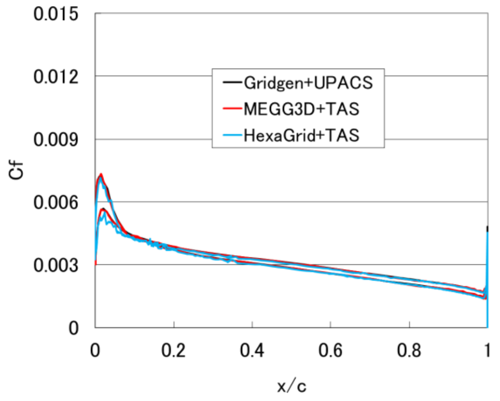


(b) Cf

Figure 16 Main wing C_p and C_f distributions at the cross-section of η=0.95



(a) C_p



(b) Cf

Figure 17 Tail wing C_p and C_f distributions at the cross-section of η=0.30

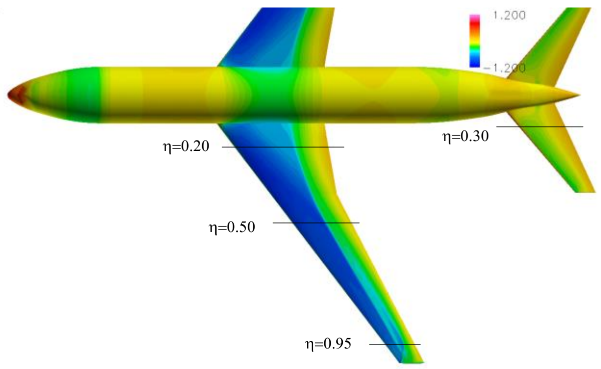
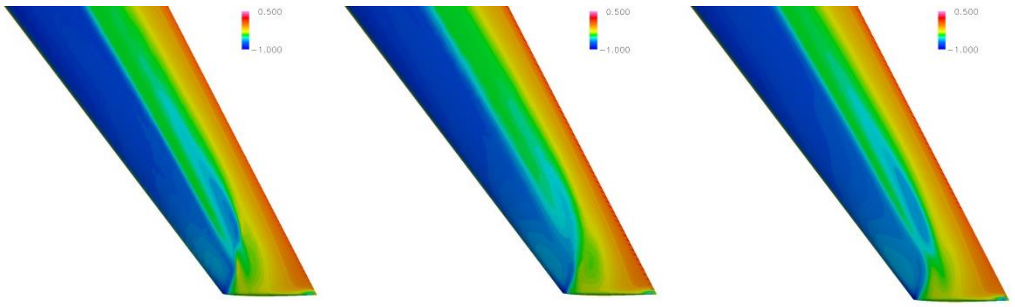
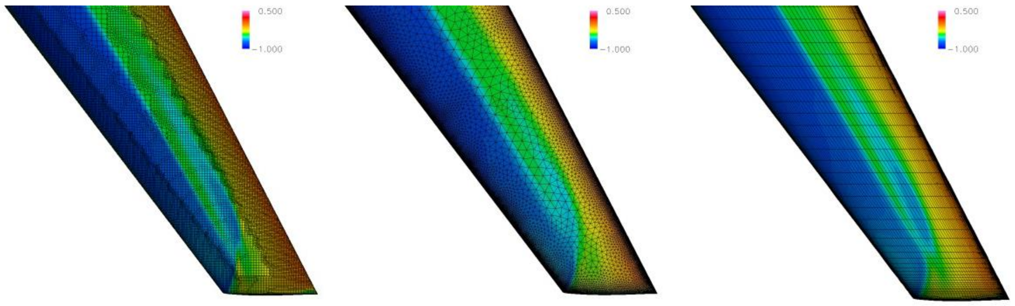


Figure 18 Cross-sections of C_p and C_f distributions



(a) HexaGrid+TAS (b) MEGG3D+TAS (c) Gridgen+UPACS

Figure 19 Wing tip C_p contours (C_L=0.5, medium grid)



(a) HexaGrid+TAS (b) MEGG3D+TAS (c) Gridgen+UPACS

Figure 20 Wing tip C_p contours with grid (C_L=0.5, medium grid)

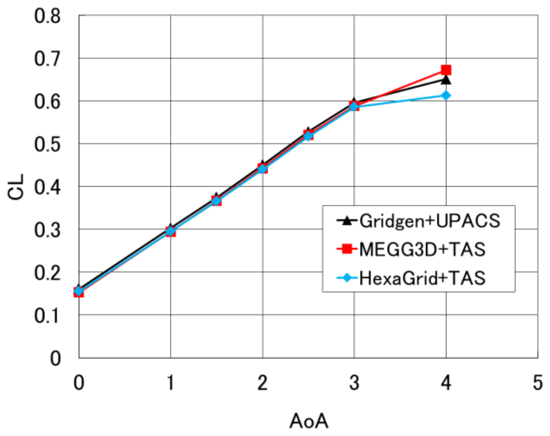


Figure 21 Comparison of C_L-α

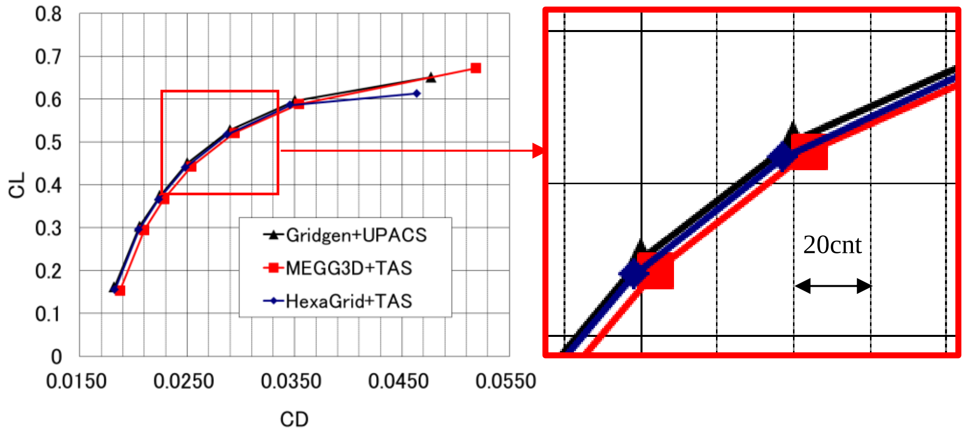
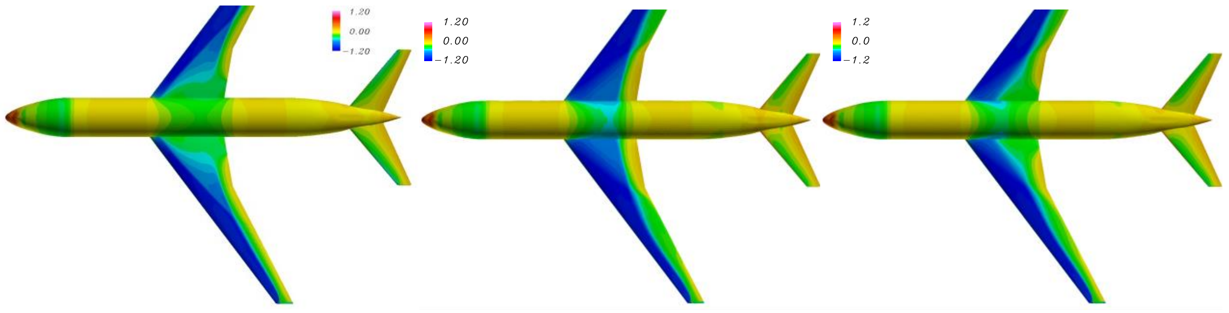
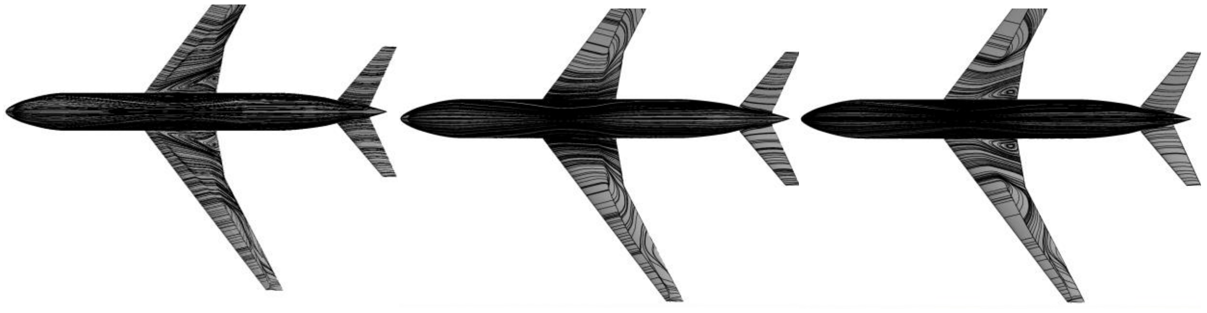


Figure 22 Comparison of C_L-C_D



(a) HexaGrid+TAS **(b) MEGG3D+TAS** **(c) Gridgen+UPACS**

Figure 23 C_p contours (AoA=4 °, medium grid)



(a) HexaGrid+TAS **(b) MEGG3D+TAS** **(c) Gridgen+UPACS**

Figure 24 Oil flow (AoA=4 °, medium grid)