Numerical Simulation of the Slider Air Bearing Problem with Complex Shapes and Steep Walls for Hard Disk Drives: Part 2 Non-nested Multi-Grid Finite Volume Scheme

Lin Wu and D. B. Bogy

Computer Mechanics Laboratory

Department of Mechanical Engineering

University of California, Berkeley, CA 94720

ABSTACT

In this paper we present a non-nested full approximation storage (FAS) multi-grid algorithm over an unstructured triangular mesh for slider air bearing simulation that can significantly speed up the convergence rate of the implicit finite volume scheme we presented in a previous paper. The multi-grid algorithm requires no relationship between the fine and coarse meshes, which offers greater convenience and flexibility in the mesh generation and takes into account the fact that the mesh generated by Delaunay triangulation in our previous paper is not nested. When combined with the implicit and adaptive finite volume scheme that has been shown to have good high frequency error damping qualities, it achieves fast convergence. The overall simulation strategy, including the mesh generation and adaptation, the implicit finite volume scheme and the multi-grid algorithm, has proved to be an efficient way of solving the generalized Reynolds equation over sliders with complex rail systems. In addition, the steady state flying attitude is obtained by a Quasi-Newton iteration method.

INTRODUCTION

In the past several years extensive effort in the Computer Mechanics Laboratory (CML) has been devoted to the development of efficient and accurate numerical methods and corresponding software for solving the slider air bearing problem in hard disk drives. Due to its accuracy, convenience, robustness and efficiency, the CML air bearing design code is now widely accepted and used in the hard disk drive industry. Before the present development the CML air bearing design code was based on the control volume schemes for convection and diffusion type equations cast in a convenient form in Patankar's book (1980). These schemes are well known for their robustness. In that code a rectangular Cartesian mesh system is used. Such a mesh system is easy to generate and the grid lines can be adaptively moved around conveniently. Sha and Bogy (1994 and 1997) further improved its efficiency by implementing the multi-grid scheme of Shyy *et al* (1993).

The sliders in current hard disk drives have become increasingly complicated. Shaped rail systems are now widely utilized to achieve the desired design goals such as low flying height and the uniformity of flying height across the radial position range of the disk. For these sliders most rail boundaries are either curved or not aligned with the coordinate axes. Along these boundaries the etching process of manufacturing leaves a narrow recess wall region in which the recess height changes abruptly through a wall profile from the value on the air bearing surface to that of the fully recessed region. Since the air bearing pressure is built up by forcing the air to flow from an open space into a narrow gap, in most cases, the recess wall region characterized by a relatively large geometric change has large pressure gradients. This is especially true on the downwind part of the boundaries of the rail. Although in most cases the pressure gradients of the recess wall regions on the upwind parts of the rail are relatively smaller (in most cases, it was so

small, after mesh adaptation, that no mesh was added to these regions in the prior CML code), these regions have non-negligible effects on the pressure profile on the rail behind them. This is especially true when the angles between the local disk velocity vector and these rail boundaries are not 90°. As a result, it is desirable to cluster fine meshes in the recess wall regions.

In our previous paper (Wu and Bogy 1999), we fully discussed the advantages of using an unstructured triangular mesh over the rectangular Cartesian mesh, if the rail geometry is complicated. In that paper we presented triangular mesh generation and adaptation techniques and an implicit finite volume scheme. The scheme is unconditionally stable and has good high frequency error damping qualities. Even though the time step can be chosen arbitrarily large the code still does not converge fast enough, due to the nature of the Gauss-Seidel smoother employed. From the convergence history it can be seen that the error drops very quickly at the beginning, its magnitude can be reduced about two orders with only a few time steps. But after the high frequency errors have been smoothed out, the convergence rate declines. In such cases, the multi-grid technique has proved to be an efficient way to greatly improve the convergence rate. In this paper we employ a non-nested multi-grid technique that suits the nature of the new mesh generation process.

The multi-grid methodology was originally developed for solving elliptic equations, and it was later applied to other types of differential equations with great success. It takes into account the fact that most iteration techniques are efficient at smoothing out the error components with wave lengths comparable to or smaller than the mesh size (high frequency errors). But after these error components have been quickly eliminated the convergence rate can greatly diminish with most computational time being consumed by the inefficient task of reducing the error components with wave lengths larger than the mesh size (low frequency errors). However, a

coarser mesh sees these components as relatively higher frequency error, so if subsequent iterations are done on the coarser mesh, the error can be continually reduced efficiently. Therefore, instead of using only one mesh, the multi-grid technique iteratively solves the problem on several sets of mesh ranging from very fine to very coarse by switching back and forth between them. In this manner a fast convergence rate can be achieved.

Most multi-grid techniques are designed for nested meshes, in which several finer meshes can be combined to form a coarser mesh. This makes the transfer of variables and residues between the fine and coarser meshes relatively easy. But due to the nature of the Delaunay triangulation used in our approach, it is impossible to generate nested meshes. Therefore a non-nested multi-grid technique has to be used. Compared with other triangulation techniques like the pure longest-side bisection technique (Rivara 1989) that can be used to generate nested meshes, the relaxation of the nested requirement can also give the user more flexibility in the mesh generation.

In our approach the full storage approximation multi-grid strategy of Brandt (1977) is implemented. Mavriplis and Jameson's (1987 and 1988) restriction and interpolation functions that suit non-nested triangular meshes are used to transfer the variables and residues between the meshes. The resulting multi-grid implicit finite volume scheme is robust and efficient in solving the generalized Reynolds equation. The overall efficiency of the new code can compete with the rectangular mesh CML code even with a similar number of nodes. To get comparable results, the new code is normally four to five times faster as a result of the significantly reduced number of required meshes.

Finally, the steady state flying height of the slider is found by a Quasi-Newton iteration method fully described in Dennis and Schnabel (1983).

IMPLICIT FINITE VOLUME DISCRETIZATION OF THE GOVERNING EQUATION AND THE BOUNDARY CONDITION

The governing equation for the air bearing problem is the modified Reynolds equation. In non-dimensional form, it can be written as

$$\sigma \frac{\partial}{\partial T}(PH) = \frac{\partial}{\partial X}\left(QPH^3 \frac{\partial P}{\partial X} - \Lambda_x PH \right) + \frac{\partial}{\partial Y}\left(QPH^3 \frac{\partial P}{\partial Y} - \Lambda_y PH \right) \tag{1}$$

The computational domain is divided into a series of non-overlapping triangles with the mesh generation techniques presented in Wu and Bogy (1999). The space descretization is achieved by integrating the governing equation over the control volumes, which are taken to be the dual Voronoi polygons of the Delaunay triangulation, while the variables are stored at the vertices of their dual triangles. The semi-discretized form of the scheme can be written as

$$A_i H_i \frac{\partial}{\partial T}(P_i) = R_i\left(P_i^{n+1}, P_j^{n+1}\right), \tag{2}$$

where $R_i\left(P_i^{n+1}, P_j^{n+1}\right)$ is the residue that comes from space discretization that can be written as

$$R_i\left(P_i^{n+1}, P_j^{n+1}\right) = \sum_{j=1, i\neq j}^{M} C_{ij} P_j^{n+1} - C_i P_i^{n+1}, \tag{3}$$

where P_j are the pressures at the vertices surrounding vertex i. The unsteady term is approximated by backward difference. The resulting fully implicit scheme can be written at each time step as

$$\left(\frac{A_i H_i}{\Delta T} + C_i\right) P_i^{n+1} - \sum_{j=1, i\neq j}^{M} C_{ij} P_j^{n+1} = \frac{A_i H_i}{\Delta T} P_i^n . \tag{4}$$

The steady state solution is found by marching in time. The definition and physical meaning of each term can be found in Wu and Bogy (1999). C_i and C_{ij} are functions of P_i^{n+1} and P_j^{n+1}, so the equations are non-linear. A lagging technique can be used to linearize the discretized equations. We can simply calculate these coefficents using the solution of the previous time step. But this requires the final solution to be found through iteration. Each time step (or outer iteration) we update the coefficents once, and the resulting linear algebra equations are solved by a two sweep Gaussian-Seidel iteration method described in Wu and Bogy (1999) (inner iteration).

The boundary condition is very simple, along the boundaries of the slider, the pressure is ambient.

GRID TRANSFER OPERATORS FOR THE MULTI-GRID ALGORITHM

In the implementation of the multi-grid algorithm, variables and residues are transferred frequently between different mesh levels. The transfer procedure has vital influence on the overall performance of the multi-grid algorithm. Mavripilis and Jameson's (1987 and 1988) grid

transfer operators have been shown to be well suited for multi-grid algorithms over unstructured triangular meshes. Here we simply adopt their techniques.

Let I_{k+1}^{k} be the operator used to transfer variables or residues from the fine mesh k+1 to the coarse mesh k, it is also called the restriction or projection operator. If I_{k+1}^{k} is operating on variables, it can simply be taken as a linear interpolation of the variables from the fine mesh nodes to the coarser mesh nodes. For example, if we want to get the pressure at vertex i, (see Fig.1) of the coaser mesh, we just need to locate the triangle IJK of the fine mesh that encloses it and do a linear interpolation. If it operates on the residue, then the residue at the vertex of a finer mesh can be distributed to the three vertices of the coaser triangle that encloses the vertex by its three area coordinates. For exmple, if we want to distribute R_I (the residue at vertex I in Fig. 2) to vertices i, j and k of a coase mesh, we simply send $L_i R_I$, $L_j R_I$ and $L_k R_I$ to vertices i, j and k respectively. L_i, L_j and L_k are area coordinates or shape functions of point I, which are equal to the area of the triangle formed by the opposite line to i, j and k and I itself and divided by the area of triangle ijk respectively. This can guarantee the conserversion of the residue in the transfer process.

Let I_{k}^{k+1} be the operator used to transfer corrections from the coarse mesh k to the fine mesh k+1, it is also called the interpolation or prolongation operator. It can simply be taken as a linear interpolation. For each vertex of mesh k+1, we just need to locate the triangle of mesh k that encloses it and linearly interpolate the corrections at the three vertices of the triangle to it.

FAS MULTI-GRID ALGORITHM

The full approximation storage (FAS) algorithm in Brandt (1977) is well suited for non-linear equations. It solves the equations by iterating over several sets of mesh. In abstract form it can be presented as follows. To simplify the expression we assume only two levels of mesh are used (k+1 represents the fine mesh and k represents the coarse mesh). Let L be the differential operator, $\underline{U}$ be the unknown vector and $\underline{F}$ be the source term, then the differential equation can be written as

$$L\underline{U} = \underline{F}. \tag{5}$$

On the fine mesh, the equation can be discretized as

$$L_{k+1}\underline{U} = \underline{F}_{k+1}, \tag{6}$$

where L_{k+1} is the discretized operator over the mesh k+1. A certain number of iterations can be carried out until the convergence rate becomes slower. Then the solution $\underline{u}^{k+1}$ and the residue $\underline{F}_{k+1} - L_{k+1}\underline{u}^{k+1}$ are transferred to the coarse mesh, and the following equation

$$L_k\underline{U} = I_{k+1}^k(\underline{F}_{k+1} - L_{k+1}\underline{u}^{k+1}) + L_k\left(I_{k+1}^k\underline{u}^{k+1}\right) \tag{7}$$

is solved there with an initial guess $I_{k+1}^k\underline{u}^{k+1}$. L_k is the discretized operator on mesh k. I_{k+1}^k is the grid transfer operator used to transfer either the solution variables or residues from mesh k+1 to

k. A solution $\underline{u}^k$ is found after enough iterations. Finally the correction on mesh k is transferred back to k+1, and the solution on mesh k+1, $\underline{u}^{k+1}$ is updated as

$$\underline{u}^{k+1} \leftarrow \underline{u}^{k+1} + I_k^{k+1}\left(\underline{u}^k - I_{k+1}^k \underline{u}^{k+1}\right), \tag{8}$$

which serves as the initial guess of the next multi-grid circle iteration. I_{k+1}^k is the grid transfer operator used to transfer the correction from mesh level k to k+1. The above process continues until the error drops to an acceptible level.

For our implicit finite volume scheme, the differential operator on level k+1can be written as

$$\left(L_{k+1}\underline{P}\right)_i = \left(\frac{A_i H_i}{\Delta T} + C_i\right) P_i^{n+1} - \sum_{j=1, i\neq j}^{M} C_{ij} P_j^{n+1}. \tag{9}$$

Similar expressions hold for the other levels. The source term on the fine mesh k+1 is

$$\left(\underline{F}_{k+1}\right)_i = \frac{A_i H_i}{\Delta T} P_i^n. \tag{10}$$

On the coase level k, it can be written as

$$\underline{F}_k = I_{k+1}^k (\underline{F}_{k+1} - L_{k+1}\underline{u}^{k+1}) + L_k\left(I_{k+1}^k \underline{u}^{k+1}\right). \tag{11}$$

In our actual implementation three levels of mesh are used. Figure 3 shows the multi-grid V cycles used in the simulation. To get a good initial guess we first do forty iterations on the coarsest mesh, then we linearly interpolate the solution variables to the second level mesh. Twenty iterations are carried out there before we transfer the solution variables to the third level mesh. After that, N V cycles are carried out before we adaptively refine the third level mesh according to the pressure distribution. The following V cycles are performed over the new finest mesh and the other two meshes until convergence is achieved. Each V cycle consists of one iteration on the finest mesh, then the solution variables and residues are transferred to the second level mesh using the operators we previously defined. Then four iterations are done there. Again, solution variables and residues are transferred to the first level mesh. Eight iterations are carried out before the corrections are transferred back and used to update the second level mesh solution. Four more iterations are done on the second level mesh. Finally, the finest mesh solution is updated by the correction transferred from the second level mesh, which serves as the initial guess for the next V cycle. There is no solid physical background for deciding how many outer iterations should be carried out on each mesh level to give the best result. The above choices correspond to optimized results for some sliders. Within each outer iteration, there is no need to solve the linear algebra equations extremely well, since for the final steady state solution the coefficients depend on the solution itself. In each outer iteration we linearized the equations by taking them to depend only on the solution of the previous outer iteration. In our code, about ten to twenty Gauss-Seidel iterations are used to find an approximation solution for the set of linear algebra equations with coefficients updated at the beginning by the solution of the last outer iteration. Again, the choice of the number of inner iterations corresponds to optimized results for some sliders.

INVERSE PROBLEM

In air bearing simulation the steady state flying attitude corresponding to a fixed prescribed load is more important than the pressure distribution of one fixed attitude, because it is the former that influences the performance of the hard disk drive. The steady state flying attitude is defined as the one at which the pre-enforced suspension force and pitch and roll torques are balanced by their counterparts generated by the air bearing force. We can define a vector $\underline{R} = (R_1, R_2, R_3)$, where

$$R_1 = F_{air} - F_s, \tag{12}$$

$$R_2 = \frac{(M_{air})_p + (M_s)_p + (M_{shear})_p}{F_{air}} - X_0, \tag{13}$$

$$R_3 = \frac{(M_{air})_r + (M_s)_r + (M_{shear})_r}{F_{air}} - Y_0. \tag{14}$$

F_{air} is the air bearing force, F_s is the applied suspension force, M_{air}, M_s and M_{shear} are moments caused by air bearing pressure, applied suspension force and viscous shear force, respectively. Subscripts p and r represent the projection in the pitch and roll directions. X_0 and Y_0 are coordinates of the position where the suspension force is applied. $\underline{R}$ is a non-linear function of the flying height, the pitch angle and the roll angle. The object is to find a particular flying attitude that makes $\underline{R}$ zero, which corresponds to the steady state flying attitude. The Quasi-Newton iteration method for non-linear problems fully described in Dennis and Schnabel

(1983) is implemented to find the steady state attitude. Our experience shows that generally only a few Newton steps are needed to find the steady state solution.

RESULTS AND DISCUSSION

The IBM Travelstar slider (Fig. 4) and the NSIC load/unload slider (Fig. 16) are used to demonstrate the performance of the above multi-grid numerical scheme and the Quasi-Newton iteration method. Figures 5, 6 and 7 show the three initial meshes for the IBM slider. Figure 8 shows the adaptively refined third level mesh. Figures 9 and 10 show the pressure contours at the steady state attitude obtained by the triangular mesh solver and the rectangular mesh solver, respectively. They are almost the same, differing only by some small details. Comparing Fig. 8 and Fig. 9, we can see that all the regions with large geometric change or pressure gradient have been efficiently captured by our mesh generation and adaptation techniques. Figure 11 shows the comparison of the convergence rate between the fully implicit iterations on a single mesh and that of the multi-grid iteration at a fixed attitude without mesh adaptation. The single mesh is the same as the finest mesh of the multi-grid iteration. From the figure it can be seen, for the single mesh iteration, that the error initially drops very fast, only ten iterations are needed to bring the error down from about 10^{-2} to 10^{-4}. But after the high frequency error has been smoothed out, the curve flattens. It takes about another 200 iterations to further reduce the error by two orders of magnitude. The multi-grid curve shows that all error components can be continuely and efficently romoved. The log error drops almost linearly with the number of outer iterations (time steps). Figure 12 shows the flying height grid convergence comparison between the triangular mesh solver and the rectangular mesh solver. The former achieves grid convergence with much

fewer node points. This is due to the fact that the node points can be much more reasonably distributed by the triangular solver than the rectangular solver. Figures 13 and 14 show the pitch angle and roll angle grid convergence history. For the node number corresponding to flying height convergence, both codes reach grid convergence. Figure 15 shows a plot of the simulation time for finding the steady state attitude as a function of the grid size. For both solvers the simulation time increases almost linearly with the node number. For the same number of node points, the triangular mesh solver costs a little more time than the rectangular solver. But this is not always true. The rail shape of the modified IBM slider is extremely regular. In this case, the rectangular mesh solver is supposed to do a good job. But the triangular mesh solver can treat all shapes equally well, no matter how complicated they are.

To demonstrate this, the NSIC load/unload slider (Fig. 16) is simulated. Figure 17 shows the flying height grid convergence history. Again, the triangular mesh solver achieves grid convergence at a much smaller grid size. But this time, the performance of the rectangular mesh solver deteriorates substantially. Figure 18 shows the simulation time used to find the steady state flying attitude as a function of node number. This time, at a similar node number, the triangular mesh solver uses almost the same amount of time as the rectangular mesh.

SUMMARY AND CONCLUSIONS

A non-nested FAS multi-grid algorithm has been successfully employed to speed up the convergence rate of an implicit finite volume scheme that we previously designed for slide air bearing simulation. The multi-grid algorithm requires no relationship between different mesh levels. On the average nearly one order of simulation time has been saved by implementing the

multi-grid algorithm. In addition the steady state flying attitude is found by a Quasi-Newton method. Even though the unstructured nature of the grid makes the situation much more complicated than that of the structured rectangular mesh, and all the information can only be stored and retrieved through a complicated data structure, the efficiency of current code can compete with the rectangular mesh counterpart with similar grid size. To get comparable results, the triangular mesh solver is generally four to five times faster depending on the complexity of the rail shape.

ACKNOWLEDGEMENT

This work was supported by the Computer Mechanics Laboratory at University of California at Berkeley.

REFERENCES

Brandt, A., "Multi-Level Adaptive Soloutions to Boundary Value Problems," *Math. Comp.* , **31**,333 (1977).

Dennis, J. E. and Schnabel, R. B., "Numerical Methods for Unconstrained Optimization and Nonlinear Equations," Prentice-Hall, Englewood Cliffs, New Jersey (1983).

Lu, S. and Bogy, D. B., "a Multi-grid Control Volume Method for the Simulation of Arbitrarily Shaped Slider Air Bearings with Multiple Recess Levels," *Technical Report No. 94-016* (not published), Computer Mechanics Laboratory, University of California at Berkeley (1994).

Lu, S., “Numerical Simulation of Slider Air Bearings,” Doctoral Dissertation, Department of Mechanical Engineering, University of California, Berkeley (1997).

Mavriplis, D. and Jameson, A., ”Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes,” *AIAA paper 87-0353* (1987).

Mavriplis, D. J., “Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes,” *AIAA J*., **26**, 824(1988).

Patankar, S. V., “Numerical Heat Transfer and Fluid Flow,” McGraw-Hill, New York (1980).

Rivara, M. C.,” Selective Refinement/Derefinement Algorithms for sequences of nested triangulations*,” Int. J. Numer. Methods Eng*., **28**,2889(1989).

Shyy, W. and Sun, C., “Development of a Pressure-correction/Staggered-grid Based Multi Grid Solver for Incompressible Recirculating Flows,” *Computers Fluids*, **22**,51 (1993).

Wu, L. and Bogy, D. B.,” Numerical Simulation of the Slider Air Bearing Problem with Complex Shapes and Steep Walls for Hard Disk Drives: Part 1 Unstructured Adaptive Triangular Mesh Generation Techniques and Finite Volume Schemes,” submitted to *J. Tribology* (1999).

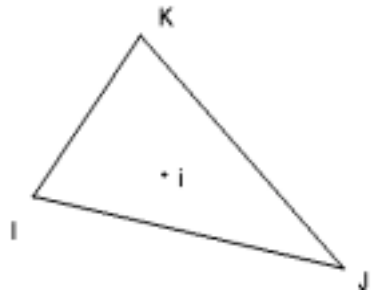

Fig. 1 Grid restriction operator.

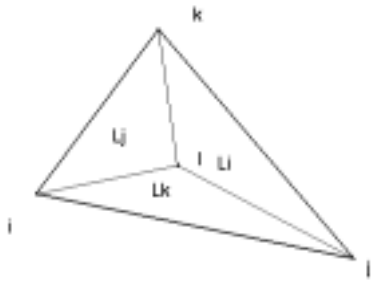

Fig. 2 Residue distribution operator.

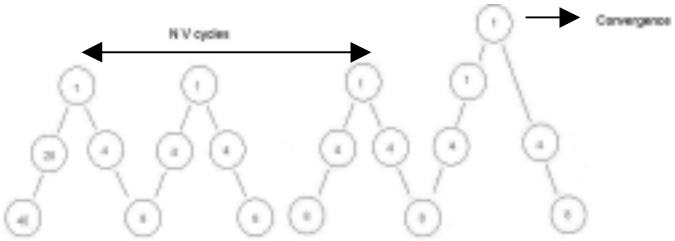

Fig. 3 The muti-grid V cycles.

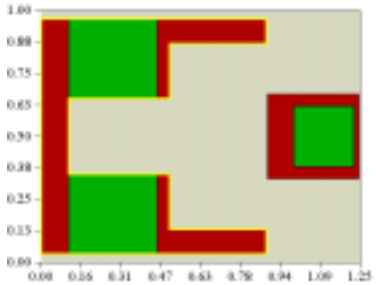

Fig. 4 The IBM Travelstar slider with slight modification.

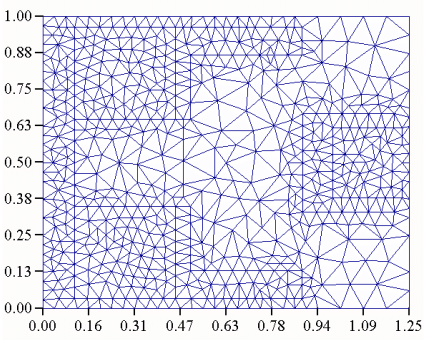

Fig. 5 The first level conforming mesh with 656 nodes.

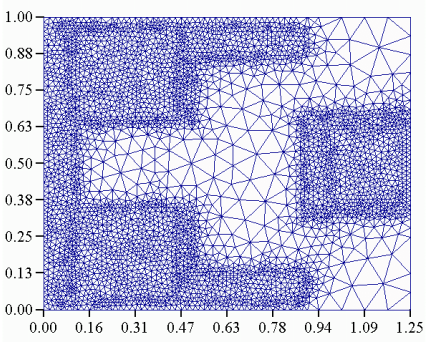

Fig. 6 The second level mesh with 4108 nodes.

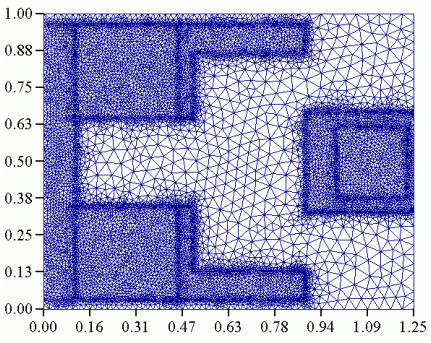

Fig. 7 The third level mesh before mesh adaptation with 12642 nodes.

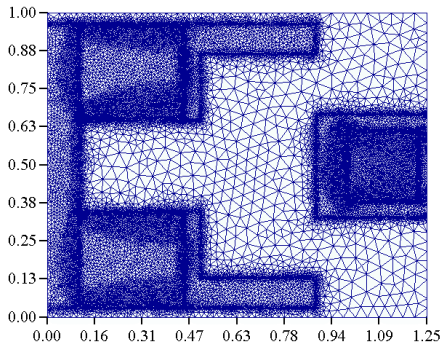

Fig. 8 The third level mesh after mesh adaptation with 18145 nodes.

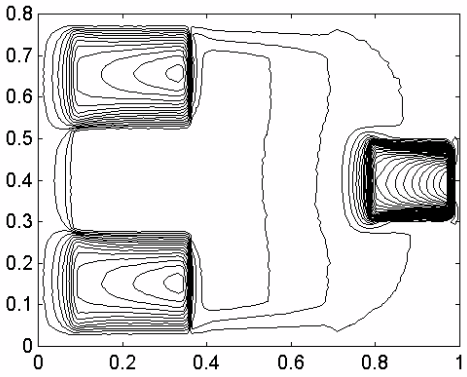

Fig. 9 The steady state pressure contour of the solution by the 18145 nodes triangular mesh solver.

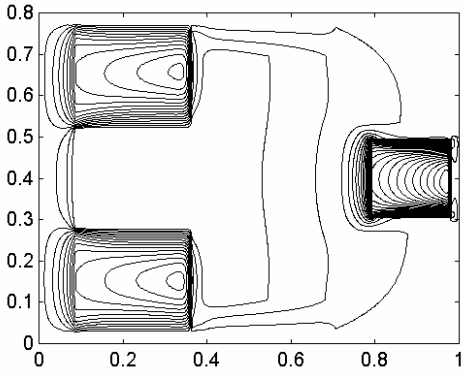

Fig. 10 The steady state pressure contour of the solution by the 148225 nodes (385X385) rectangular mesh solver.

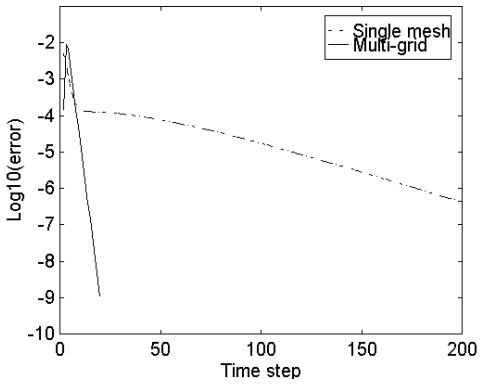

Fig. 11 The convergence history of iteration on a single mesh and multi-grid iteration.

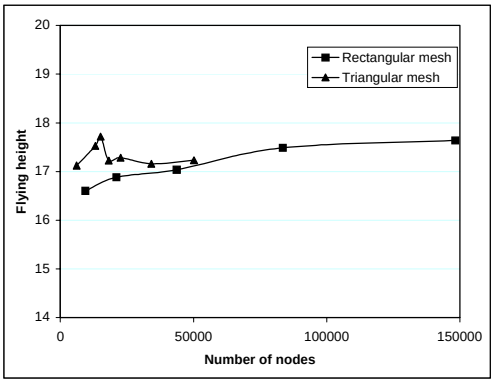

Fig. 12 The grid convergence comparison of nominal flying height (NM).

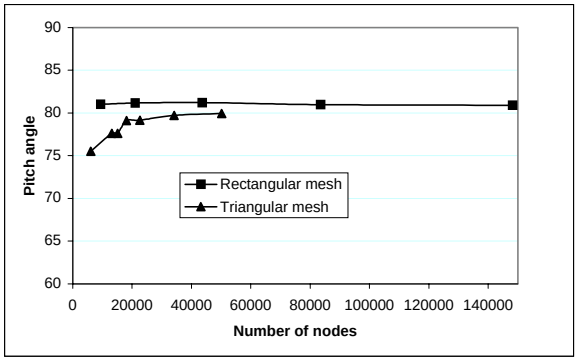

Fig. 13 The grid convergence comparison of pitch angle (μRad).

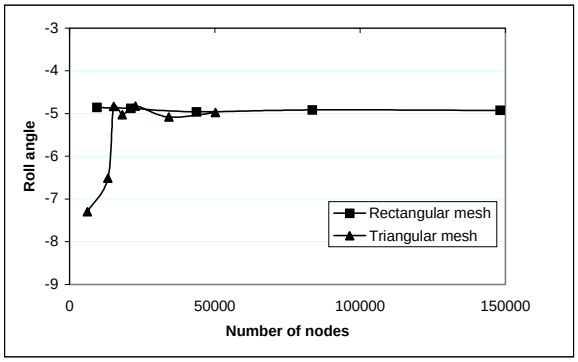

Fig. 14 The grid convergence comparison of roll angle (μRad).

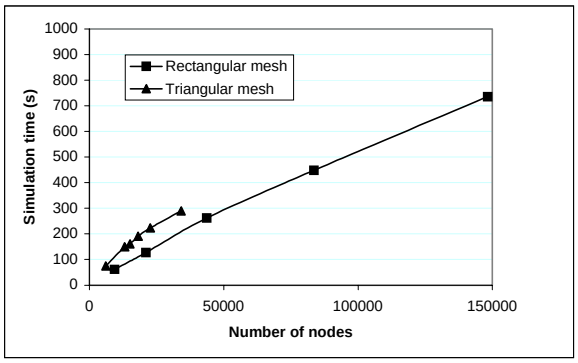

Fig. 15 The simulation time as a function of node number.

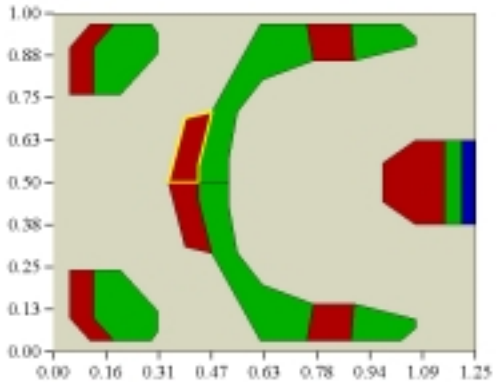

Fig. 16 NSIC load/unload slider.

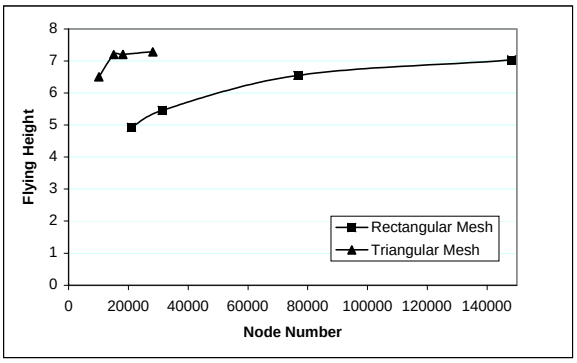

Fig. 17 The grid convergence comparison of nominal flying height (NM) of the NSIC load/unload slider.

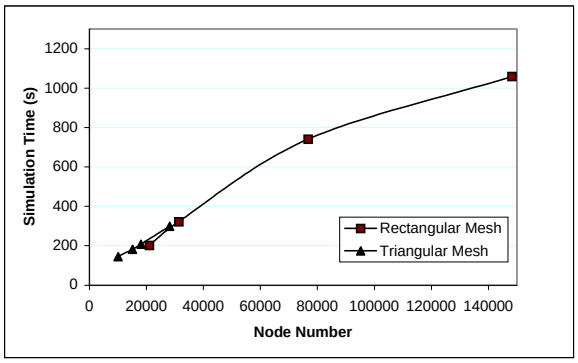

Fig. 18 The simulation time as a function of node number for the NSIC load/unload slider.