Comparison of Interpolation Methods for Numerical Analysis on Helicopter Noise

Choongmo Yang, Paulus Lahur and Takashi Aoyama
Japan Aerospace Exploration Agency (JAXA)
7-44-1, Jindaijihigashi-machi, Chofu, Tokyo 182-8522, Japan

INTRODUCTION

JAXA has been developing its own full helicopter simulation code[1] by combining accurate CFD solver and acoustic solver. The flow solver is using moving overlapped grid method, which is one of the most advanced techniques for tip-vortex capturing at present. The moving overlapped grid system is composed of three different types of grids (blade grid, inner and outer background grids), and simple bi-linear interpolation method is used to exchange the information between each grid during calculation. The acoustic code is based on Ffowcs Williams and Hawkings (FW-H) formulation using the pressure distribution on blade surface obtained by the CFD code as input data.

The previous researches have shown its ability to capture the distinct peak of BVI noise for several problems. Also the code is expanding its ability to solve the flow-field including tail-rotor and fuselage configuration for interaction noise analysis. The noise generated by a maneuvering rotorcraft would be the next important challenge in the way to full helicopter simulation. To get accurate noise signals for the noise problems of helicopter, it is very important to solve the fully unsteady flow field with high-accuracy for several rotor revolutions. Because of complex movement of helicopter including rotor-rotation and flight motion of all parts of helicopter, the importance of accurate and fast interpolation algorithm between overlapped grids is growing up. Specially, for the computation of a maneuvering helicopter, the computing efficiency becomes one of the bottle-necks.

In this paper, the several interpolation algorithms are implemented and compared for elapsed computing time. Compared to linear index searching algorithm, Alternating Index Searching algorithm, which will be explained later, shows better performance to accelerate calculation with overlapped grid system. Alternating Index Searching algorithm searches the nearest point by checking neighbor points to alternate its index direction for a given point. Alternating Digital Tree (ADT)[2] algorithm is also known to have excellent searching speed to find the intersection line or plane between two meshes. But if we restrict the interpolation for a structured grid, which can be characterized by linear indexing along computational coordinate axis, then alternating index searching algorithm shows better performance in searching operations. So, in this paper, three different alternating index searching algorithms are compared to linear index searching methods during interpolation operations. By applying this algorithm, efficient massive computation can be achieved for the full helicopter configuration. The results of this study can provide understanding of the characteristics for helicopter noise, which can be valuable in full helicopter design.

NUMERICAL METHODS

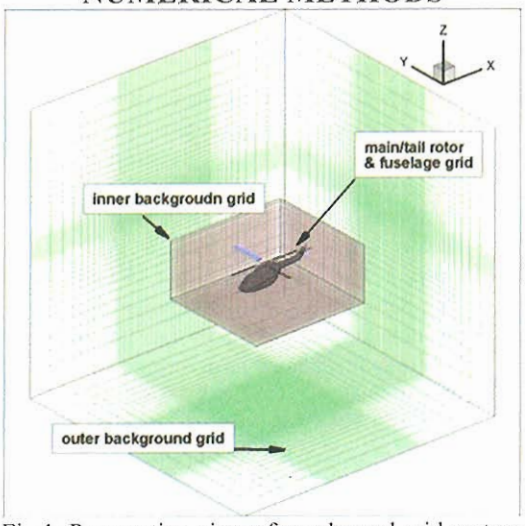


Fig.1: Perspective view of overlapped grid system

Overlapped Grid System:A moving overlapped grid system with three different types of grids (rotor grid, inner and outer background grids) is used to simulate BVI of helicopter. Figure 1 shows a perspective view of grid system for the whole computational domain of grid system. The inner background grid is placed around the rotor disk. The outer background grid covers the whole computation region with a sparse grid density. The flow data are exchanged between the inner and outer background grids, and between rotor grid and inner-background grid. The body-fitted blade grid in O-H topology moves along with the blade motion including rotation, flapping, feathering, and lagging.

Table 1 shows the specification of each grid. Most of the grid is concentrated in inner-background grid, which captures the trace of tip vortex during several rotations around rotor disk. The number of grid points in span-wise direction is considerably increased to match the grid density of the blade grid with that of the inner background grid. The grid spacing of the inner background grid corresponds to 0.05c, where c is the chord length.

Table 1: Specification of each grid.

inner background grid	(X×Y×Z) 450×400×80 = 14,400,000
outer background grid	(X×Y×Z) 83×79×49 = 321,293
blade grid	(chord×normal×span) × blade (83×25×131) × 1 = 271,825
total	~15,000,000 points
inner background spacing	0.05c (=0.006*R*)

Numerical Schemes for Aerodynamics: A three-dimensional numerical flow solver for the compressible Euler equation is used to analyze the detailed behavior of tip vortex.

For the calculation of blade grid, inviscid flux vectors are separated using Roe's flux difference splitting (FDS) algorithm, with third-order accuracy using a TVD scheme. For the time integration, second-order Euler backward scheme is used in the conventional delta form. A diagonalized ADI method with an upwind flux-split technique is used in the linearized implicit part for the discretionary governing equations. A detailed derivation of the governing equation and numerical schemes is described in a previous work by Aoyama et al.

For the calculations over background grid, the flux difference across cell interface is divided also using a compact TVD scheme to get third order accuracy. MUSCL cell interface value is modified to achieve 4th-order high accuracy in the background Cartesian grid. Simple High-resolution Upwind Scheme (SHUS) is employed to obtain numerical flux. SHUS is one of the Advection Upstream Splitting Method (AUSM) type approximate Riemann solvers and has small numerical diffusion. The four stage Runge-Kutta method is used for the present calculation. The free stream condition is applied for the outer boundary of the outer background grid.

Aeroacoustics: The prediction method of the far field acoustic pressure is based on the combination of CFD technique with an acoustic equation solver. Although direct computation can be used to get the noise solution directly from the flow calculation with CFD based methods, this is available only in the near field in spite of huge computing cost. At present, the best way is the coupling with the integral method for far-field prediction. Acoustic analogy, which is re-arranged into the Ffowcs Williams-Hawkings Equation, is widely used and still under construction for better applications. Retarded time solution to the Ffowcs-Williams and Hawkings equation, neglecting quadruple noise, can be written in the form of Formulation1 by Farassat[3]. The prediction of rotor noise is conducted in the following procedures: 1) calculation of sound pressure of the noise source, 2) acoustic prediction computation at the observer position, and 3) post-processing of the noise data in the way of sound level using visualization or audible converting.

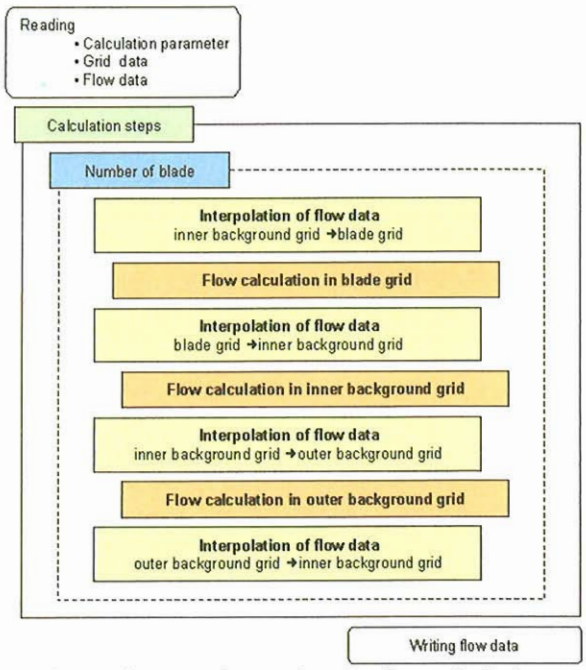
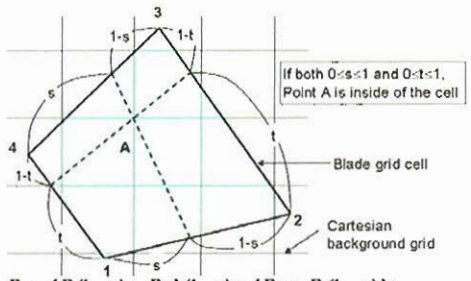


Fig.2: Diagram of procedure for flow calculation

Calculation Procedure: The dynamic blade motions such as flapping, feathering, and lagging are defined in the input data. The solver uses azimuth-wise data or 1st harmonic function data obtained by measurements or other codes (e.g. CAMRAD). In the present calculation, the collective pitch, cyclic pitch, flapping, and lagging angles measured by the wind tunnel experiment by ATIC are used. The calculation procedure is shown in the diagram of Fig. 2. The search and interpolation to exchange flow data between the grids are executed in each time step because the blade grid rotates with the rotor blade in the background grids. The computation time spent for search and interpolation is one of the disadvantages of the moving overlapped grid approach.



$$P_A = \{P_1(1-s)+P_2 s\}(1-t)+\{P_3 s+P_4(1-s)\}t$$
$$=\{P_1-P_2+P_3-P_4\}st+(-P_1+P_2)s+(-P_1+P_4)t+P_1$$

Fig.3: Diagram of bi-linear interpolation for overlapped grid system

Bi-linear Interpolation between Overlapped Grids: Figure 3 shows bi-linear interpolation of 2D for simplicity. In case of 3D, the position of the point is expressed by three scalar parameters, s, t, and u for the use of tri-linear interpolation. In this step, values of s, t, and u for each index are calculated. When all s, t, and u are between zero and one, the point is judged to be located inside of the cell.

INTERPOLATION ALGORITHMSS
(Spatial Searching Algorithm)

When using overlapped grid or other kinds of multiple meshes, solutions on one grid must be accurately interpolated and transferred to the second grid for the calculation to proceed. In the case of structured/unstructured grids or the more general multi-physics case of two independent grids with arbitrary overlap, this grid transfer operation is more complicated. It involves a search for which element of the first mesh contains each nodal point of the second mesh, and a subsequent interpolation. If the two meshes move relative to each other, then the search operation must be repeatedly invoked. It needs considerably high computational cost especially for the calculation of helicopter in maneuver.

Among several kinds of interpolations between different grids, only the detailed procedure of the data exchange from the blade grid to background grid is described here. The other procedure of the data exchange from the Cartesian background grid to the blade grid is easier than that from the blade grid to the background grid.

Assume we have a blue mesh from base grid (corresponding to base grid such as main-rotor, tail-rotor, and fuselage in the present code) consisting of elements and one or more scalar or vector values defined at its nodal points as shown in Fig.4. We also have a red mesh (corresponding to inner background grid) of elements and nodal points whose spatial extent overlaps that of the blue mesh in some arbitrary way.

The grid transfer operation is to interpolate from nodal values of the blue mesh onto nodes of the red mesh, which is shown as point A, B, and C in Fig. 4.

To speed up the searching, the searching points are checked at first whether they are located inside or outside of the box which consists of maximum and minimum values of base grid, which is expressed 'in-box' in the figure. If a red nodal point is outside in-box (point A), the point is categorized as "out of box". If the point is located within 'in-box', the search algorithm begins to work to find the target cell points. If successful (point C), the value of the point 'in box' is interpolated using cell points, 1, 2, 3 and 4. If not, it means that the point is 'in box but out of range' and it may be ignored or an extrapolation procedure may be used. If a red nodal point lies on the face (or edge) between two (or more) blue elements, it can be considered to be inside either for interpolation purposes.

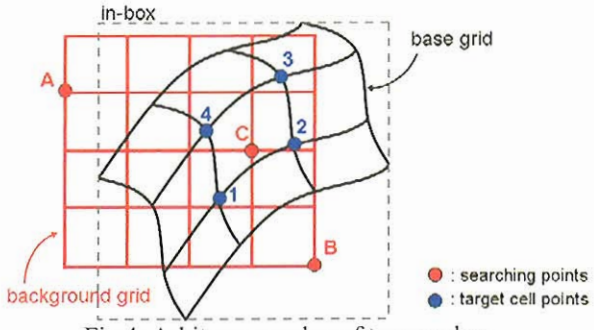


Fig.4: Arbitrary overlap of two meshes

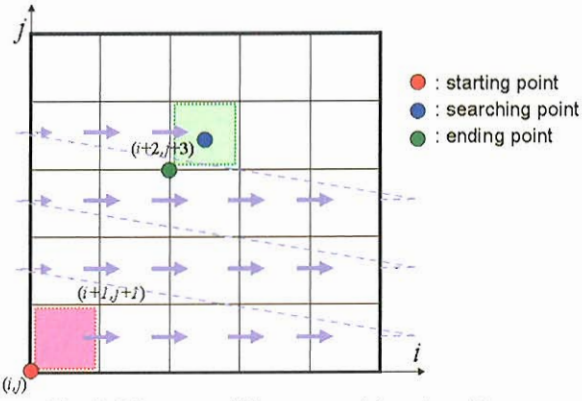


Fig.5: Diagram of linear searching algorithm

Linear Searching Algorithm: Most simple searching algorithm for the structured grid (i.e. linear index mesh) is linear searching algorithm by brute force approach. Let (i, j) be index of base grid and (I, J) be index of background grid as shown in Fig. 5. Linear searching algorithm continues checking if the searching point (I, J) is in the cell by nodal points (i, j), (i+1, j), (i+1, j+1) and (i, j+1). If the cell includes the searching point, the value of the point 'in box' is interpolated using cell points, then the next searching point of

background grid will be searched. If no cell is found to include the searching point, it is ignored and procedure exit o the next searching points.

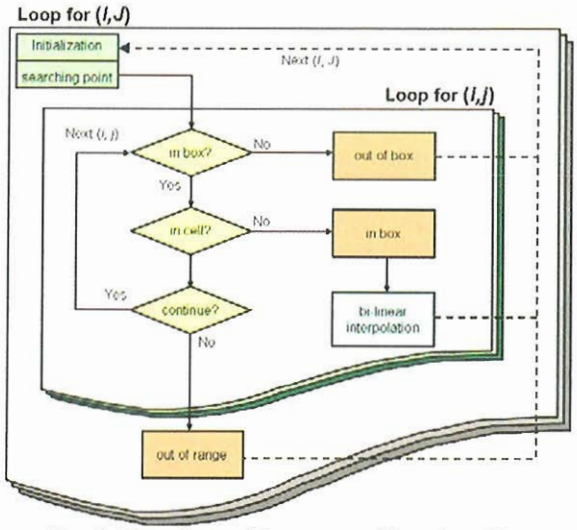


Fig. 6: Flowchart of linear searching algorithm

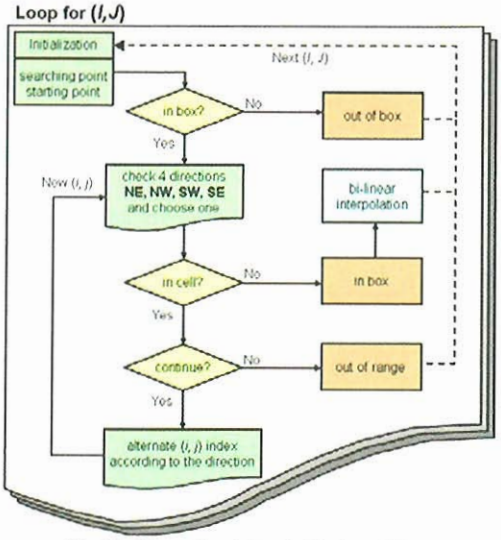


Fig.7: Flowchart for AIS algorithm

Figure 6 shows a flowchart of the linear searching algorithm. At the beginning, the grid indexed (I, J) of the background grid point is assigned to be a searching point in turn, and repeats checking for the loop of each base grid cell. During the loop of base grid (i, j), the base cell is checked whether they include the searching point or not. The searching point (I, J) of background grid should be checked in 2-D loop, and each searching point should be nested by base grid loop. This algorithm is easy for coding and secure for any kind of geometry because all cells of base grid are checked, but unnecessary index searching may be executed because the increase of index has no relation to the approaching direction to the searching point from current position. Specially, if the searching point is located 'in box but out of range', the whole loop of base grid should be repeated in vain. For a complicated geometry such as helicopter fuselage, the base grid box may include huge number of points 'in box but out of range', which give rise to undesirable computing cost during interpolation. Moreover, helicopter in maneuver is changing its position and orientation for every moment to require a massive interpolation. That is why new searching algorithms should be examined for the helicopter simulation in maneuver.

Alternating Index Searching (AIS) Algorithm: Another searching algorithm for the structured grid is Alternating Index Searching (abbreviated as AIS from now on) algorithm, which changes the direction of indexing by the judge law until the base point includes the searching point (or until it is proven out of range). This algorithm can accelerate searching by jumping index according to the directivity for searching point by increasing/decreasing (i, j) index, which is the most powerful for the structured grid. The starting point of base grid (i, j) can be chosen arbitrarily such as (1, 1) or middle point of whole base grid. Another choice is to use the ending point of previous searching because the next searching point (I, J) is probably located near the previous one, which is also valid for the structured grid. The flowchart of procedure is shown in Fig. 7.

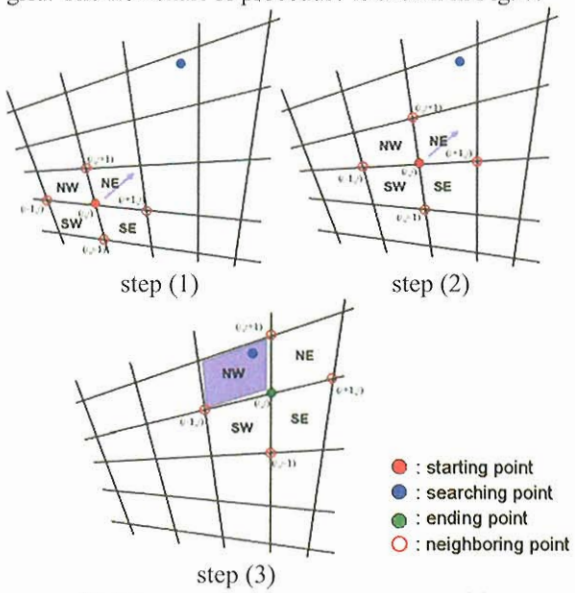


Fig.8: Steps for linear searching algorithm

Figure 8 shows examples of steps for linear searching algorithm starting from (i, j) to approach the searching point (I, J) and finally to find the cell including the searching point. In step (1), 4 neighboring cells (NE, NW, SW, and SE) of the base point (i, j) are checked to

find the approaching direction, then for the cell in the direction is checked if this cell includes the searching point. The approaching direction is determined from the relative position of spatial vectors by 5 points, (i, j), (i+1, j), (i+1, j+1), (i, j+1) and searching point. In the present example, the North-East (NE) direction is chosen but doesn't include the searching point, so the base point moves to NE by alternating index (i, j) to (i+1, j+1). In step (2), the same routine is repeated to move the base cell to NE again. In step (3), the approaching direction change to NW and the NW cell includes the searching point. Then, we can interpolate the value of searching point using 4 nodal points of NW cell: (i, j), (i+1, j), (i+1, j+1) and (i, j+1).

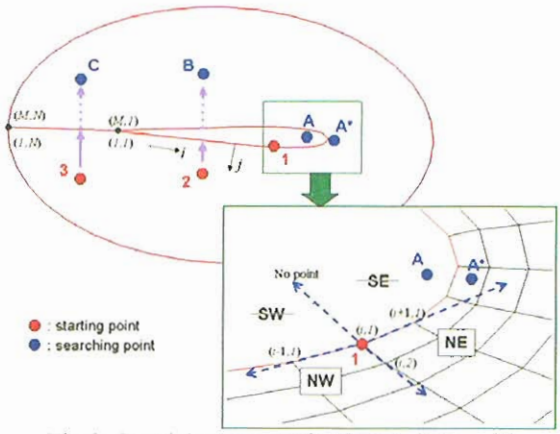


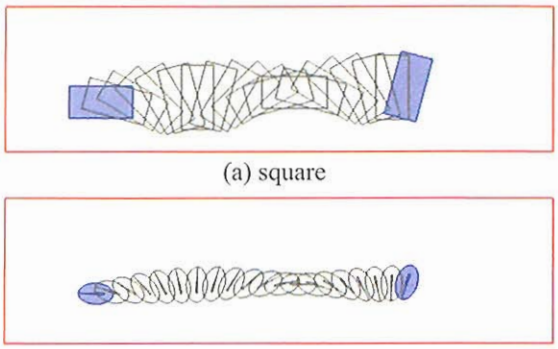
Fig.9: Special treatment for C-type base grid

Figure 9 shows three examples of special treatment for C-type base grid when using the Alternating Index Searching algorithm. These cases come from the characteristics of boundary lines of C-type grid. Case 1 occurs when starting point lies in the boundary line of j=1. If searching point is A, the final answer should be 'in box but out of range' because an approaching direction to SE cannot continue. But if searching point is A*, a extra treatment should be added for correct searching even approaching direction SE is not available,. Case 2 occurs when approaching direction has to cross the j=1 boundary starting from point 2 to approach point B. Another case occurs when approaching direction has to cross i=1 or i=M lines, which is identical in C-type grid topology. These treatment routines are included in the present research.

RESULTS AND DISCUSSION

In this section, searching efficiencies of each algorithm are compared using 2-dimensional models with different grid density for both base grid and background grid, then 3-dimensional calculating time are discussed.

Comparison of Algorithms using 2D Geometry in Motion: Searching efficiencies of each algorithm are compared using 2-dimensional models with different grid density (coarse grid or fine grid) for both base grid and background grid. Linear searching algorithm and AIS algorithm are used as shown in Table 2. Three different starting points are compared for the case of AIS algorithm. Grid number of each grid type is listed in Table 3, and 4 different base grids are shown in Fig. 9. Computing time during searching and step count (in other words, the number of alternating index operating) for each algorithm are compared. In order 1) to get an average value and 2) to eliminate effects of relative position and orientation between base grid and background grid, base grid is rotated and translated for 20 times as shown in Fig. 10. Total number of interpolated points (N), averaged computing time (T), and averaged step count (S) for each algorithm are listed in Table 4. For all cases, the results show the excellence of AIS algorithm, especially when starting from the previous end point.



(a) square

(b) airfoil

Fig.10: Diagrams of 2D Geometry in Motion (rotation and translation)

Table 2: Definition of algorithm

Algorithm Name	Exploration
LS	Linear Search algorithm
AIS-I	Alternating Index Search algorithm starting from initial point, (1,1)
AIS-M	Alternating Index Search algorithm starting from middle point
AIS-P	Alternating Index Search algorithm starting from previous goal point

Table 3: Specification of grid number

Grid type		coarse	fine
Base grid	Square	81×24	161×47
	Airfoil (with hole inside)	81×24	161×47
Background grid (including searching points)		450×80	899×159

Comparison of Algorithms using Full 3D Helicopter Simulation: Searching efficiencies of each algorithm are compared using overlapped grid for full 3D helicopter simulation. Figure 11 shows the ration of averaged computing times for each procedure during iteration. Most time consuming routine is the calculation of wing grid which is composed of 9 parts (4 blades for main-rotor, 4 blades for tail-rotor and fuselage). If considering the computing time of each part, calculation of inner-background grid costs most, which is reasonable in terms of huge number of grid points. Computing time for interpolation from the blade grid to background grid is about 11% when using linear searching algorithm. As more number of helicopter components should be included such as skid, stabilizer, and so on, as more importance of interpolation efficiency may arise. From the result of 2D examples, it is expected to increase computing efficiency in 3D helicopter simulation using AIS algorithm about 10%.

SUMMARY

Alternating Index Searching (AIS) algorithms are tested and adapted to achieve the speed-up of computing time for the massive computation of the full helicopter configuration. Comparison of 2D examples shows the excellence of AIS algorithm, especially when starting from the previous end point. Considering the interpolation time for overlapped grid method, AIS algorithm shows good possibility for computing efficiency in 3D helicopter simulation.

REFERENCES

1. Yang, C., Aoyama, T., and Saito, S., "Numerical Study on BVI Noise Reduction Using Active Flap Control", 31st ERF, No. 24, Florence, Italy, September (2005)
2. 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, 1-17, (1991)
3. Farassat, F., Theory of noise generation from moving bodies with an application to helicopter rotors, NASA TR R 451 (1975)
4. Steven J. Plimpton, Bruce Hendrickson, James R. Stewart: A parallel rendezvous algorithm for interpolation between multiple grids. J. Parallel Distrib. Comput. 64(2): 266-276 (2004)

Table 4: Comparison of total number of interpolated points, averaged computing time (10^{-6} sec), and averaged step count for each algorithm and grid type.

Back. grid \ Base grid		Square		Airfoil	
		coarse	fine	coarse	fine
coarse		**N=** 33059	33059	9441	9441
	LS	**T(S)=** 88386(1231)	316310(4923)	19361(1178)	68587(4702)
	AIS-I	16607(35.2)	32064(69.6)	2879(24.2)	5076(47.6)
	AIS-M	12444(20.6)	23706(40.6)	3034(21.4)	5338(41.8)
	AIS-P	4234(6.3)	7105(11.5)	1346(5.5)	1975(10.0)
fine		132357	132357	37737	37737
	LS	354072(1230.6)	1271496(4922)	78096(4702)	275950(4707)
	AIS-I	67054(35.1)	129160(69.6)	11846(24.1)	20970(47.6)
	AIS-M	50543(20.6)	95657(40.6)	12430(21.5)	21959(41.9)
	AIS-P	15610(5.4)	25114(9.9)	4835(4.1)	6847(7.1)

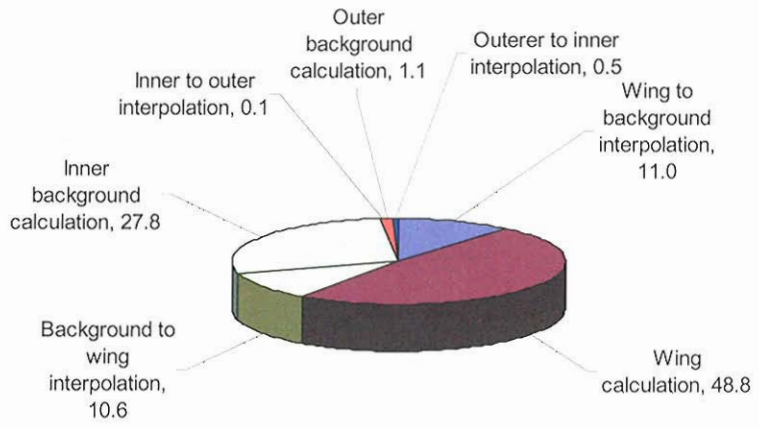


Fig.11: Diagram of averaged percentages of computing time for each procedure during iteration (%)