INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids **26**: 713–724 (1998)

A SIMPLE GRID GENERATION METHOD

QUANBAO ZHOU*
Silsoe Research Institute, Wrest Park, Silsoe, Bedford MK45 4HS, UK

SUMMARY

This paper presents a simple grid generation method which adopts the uni-directional interpolation idea but only interpolates one co-ordinate between two opposite boundaries. The use of this new scheme showed that (a) it is convenient to use; and (b) compared with the built-in grid generator of certain commercial computational fluid dynamics (CFD) code, it saves time significantly. © 1998 John Wiley & Sons, Ltd.

KEY WORDS: grid generation

1. INTRODUCTION

In the past two decades, since the pioneering work of Thompson *et al.* [1–2], the role of numerical grid generation in computational fluid dynamics (CFD) has received much attention. Since then considerable advances have been made in grid generation techniques. In the class of structured grids, algebraic [3], elliptic [4], composite or hybrid [5,6] and adaptive [7] grid generation schemes have been devised. Some general generators are available, such as Eagle [8], 3DGrape [9], INGRID [10], GRID2D/3D [11] and FLAGG [12]. Most of them, however, require a considerable amount of CPU time (some may exceed the time for flow simulation) due to the effort to gain the technique's flexibility [12].

Many commercial CFD codes, including Phoenics (Cham), Fluent, CFX4 (AEA Technology), and FIDAP (Fluid Dynamics International), also offer the facilities for mesh generation. They are often called preprocessors or built-in grid generators. However, customers often find them confusing and time-consuming due to various limitations, especially when defining a domain surrounded by curved boundaries. This paper will present a new and simple algebraic grid generation method.

2. GRID GENERATION

For ease of description, first consider a two-dimensional domain, as shown in Figure 1. For this physical domain, the inner grid can be determined using a multi-directional interpolation scheme as follows [13–15]

* Correspondence to: Silsoe Research Institute, Wrest Park, Silsoe, Bedford MK45 4HS, UK.

CCC 0271–2091/98/060713–12$17.50

Received November 1996
Revised May 1997

$$R(\xi,\eta)=[(1-\xi)\ \xi]\begin{bmatrix}R(0,\eta)\\R(1,\eta)\end{bmatrix}+[R(\xi,0)\ R(\xi,1)]\begin{bmatrix}1-\eta\\\eta\end{bmatrix}-[(1-\xi)\ \xi]\begin{bmatrix}R(0,0) & R(0,1)\\R(1,0) & R(1,1)\end{bmatrix}\begin{bmatrix}1-\eta\\\eta\end{bmatrix} \tag{1}$$

On the right hand side of Equation (1), the first and second terms are interpolators between two constant η boundaries and two constant ξ boundaries respectively, while the last one is deliberately designed for the purpose of preserving the four boundaries. Although Equation (1) maps four boundaries correctly, it has disadvantages, in that the grids near the curved boundaries may not be uniform, such as shown in Figure 2—a rectangular sitting on a quarter of a circle. If the circle represents a cylinder and the problem is to simulate the flow around the cylinder (in which case the grid has to be reflected against its y-axis for symmetrical flows or both axes for transient flows), Figure 2 shows that using Equation (1), the gridding near the cylindrical surface is rather poor. Another disadvantage of the above equation is that no partial derivatives of the boundary curves have been included and therefore the grid generated may not be locally orthogonal to boundaries. Several improvement methods have been proposed [3,11] and given below are expressions of Shih *et al.*'s.

$$x(\xi,\eta)=x'(\xi,\eta)+\Delta x(\xi,\eta), \tag{2}$$

$$y(\xi,\eta)=y'(\xi,\eta)+\Delta y(\xi,\eta). \tag{3}$$

The detailed expressions of $x'(\xi,\eta)$, $y'(\xi,\eta)$, $\Delta x(\xi,\eta)$ and $\Delta y(\xi,\eta)$ are listed by Shih *et al.* [11]. To use Equations (2) and (3), four constants, K_1, K_2, K_3 and K_4 need to be determined. Shih *et al.* proposed the values of 0.3, 0.3, 0.1 and 0.1 to be used corresponding to K_1, K_2, K_3 and K_4, respectively, but gave no reason why these values were chosen. Figure 3(a) shows the grid generated using this method for a domain the same as that of Figure 3. Better gridding near the cylindrical surface was obtained using Shih *et al.*'s algorithm, but overlapping may occur (Figure 3(a)) when K_1, K_2, K_3 and K_4 are set to values of 0.3, 0.3, 0.1 and 0.1 respectively, as proposed by Shih *et al.* This overlapping can be avoided by adopting smaller values of K_i ($i=1, 2, 3, 4$), such as $K_1=0.06$, $K_2=0.06$, $K_3=0.02$ and $K_4=0.02$ (Figure 3(b)). These values were obtained using a trial-and-error method and therefore should not be regarded as optimal. Clearly, these values have a significant effect on gridding but the way to obtain the optimum values may be time-consuming.

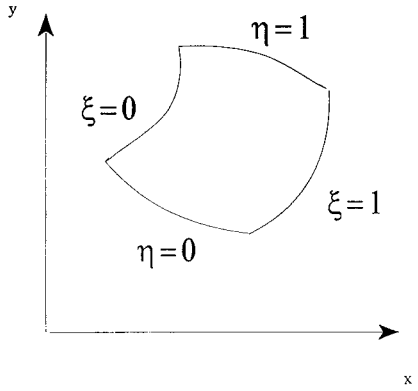

Figure 1. An arbitrary two-dimensional domain.

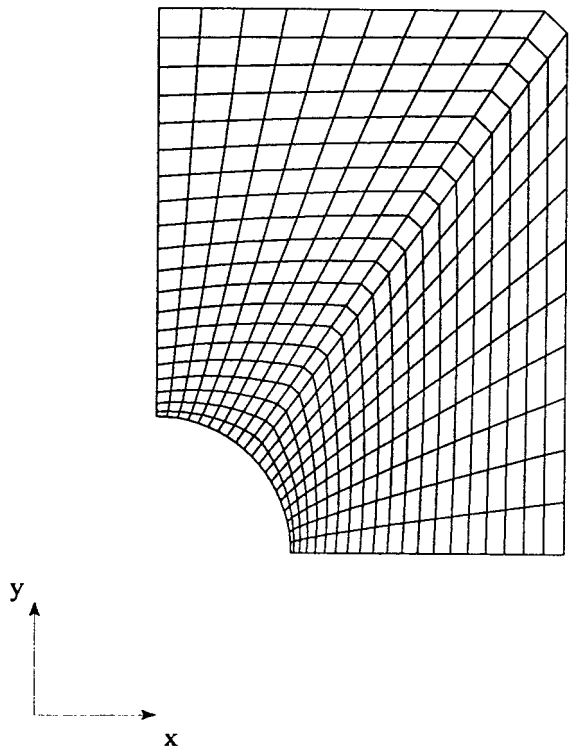

Figure 2. Grid generation using the multi-domain method, Equation (1).

In this study, an alternative interpolation scheme is proposed by adopting the uni-directional interpolation idea [1], but only interpolating one co-ordinate between two opposite boundaries. Consider the domain shown in Figure 1, interpolating the x-co-ordinate between the $\xi = 0$ and $\xi = 1$ boundaries and the y-co-ordinate between the $\eta = 0$ and $\eta = 1$ boundaries, we get the following mathematical expressions for inner grid co-ordinates.

$$x(\xi, \eta) = x(0, \eta) + b_\xi[x(1, \eta) - x(0, \eta)], \tag{4}$$

$$y(\xi, \eta) = y(\xi, 0) + b_\eta[y(\xi, 1) - y(\xi, 0)], \tag{5}$$

where, b_ξ and b_η are two stretching functions given below.

$$b_\xi = (1 - a_\eta)s_1(\xi) + a_\eta s_0(\xi), \tag{6}$$

$$b_\eta = (1 - a_\xi)r_1(\eta) + a_\xi r_0(\eta), \tag{7}$$

$$a_\xi = \frac{(1 - \xi)^m}{\xi^m + (1 - \xi)^m}, \tag{8}$$

$$a_\eta = \frac{(1 - \eta)^m}{\eta^m + (1 - \eta)^m}. \tag{9}$$

Here, a_η, $(1 - a_\eta)$, a_ξ and $(1 - a_\xi)$ are weighted functions and m is a constant which controls the effect of boundary stretching functions, $s_1(\xi)$, $s_0(\xi)$, $r_1(\eta)$ and $r_0(\eta)$ (see Figure 4), on the inner grid distribution. Figure 5 shows the relationship between a_ξ and ξ with $m = 1$, 2 and 3 corresponding to linear, quadratic and cubic transformations from $r_0(\eta)$ to $r_1(\eta)$, respectively.

The calculation of stretching functions on four boundaries is illustrated in Figure 6. As an example, the figure shows the boundary $\eta = 0$ only. The stretching function $s_0(\xi,)$ is calculated using the projection of this boundary on the x- or y-axis, depending on the difference of x- or y-co-ordinates between two end points at $\xi = 0$ and $\xi = 1$. Its mathematical expression is given below.

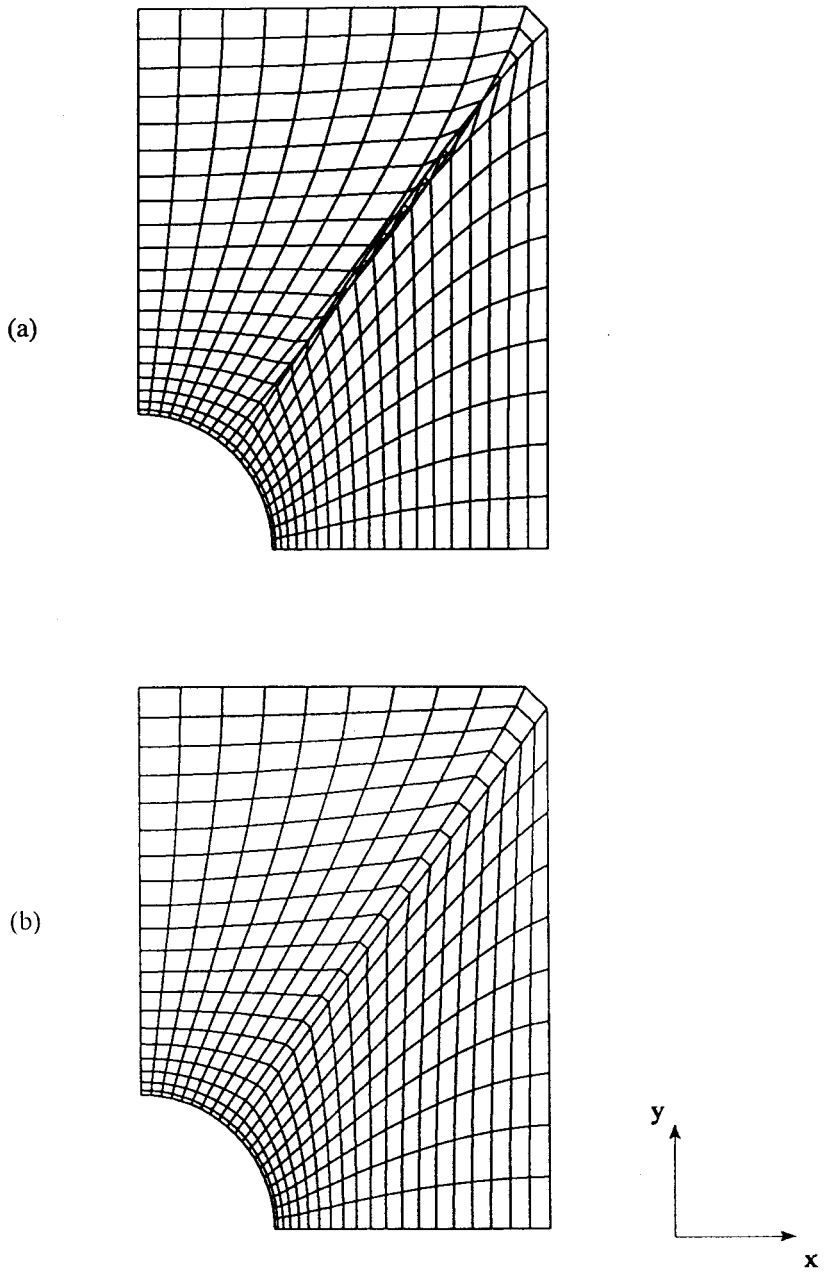

Figure 3. Grid generation using Shih *et al.*'s scheme: (a) $K_1 = K_2 = 0.3$, $K_3 = K_4 = 0.1$, (b) $K_1 = K_2 = 0.06$, $K_3 = K_4 = 0.02$.

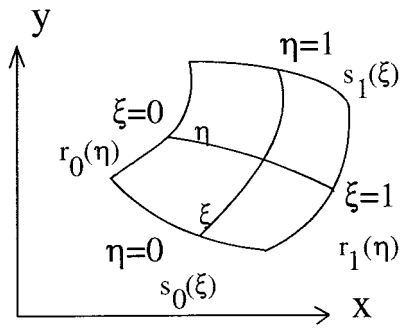

Figure 4. Stretching functions and their corresponding boundaries.

$$s_0(\xi) = s_{x0}(\xi) \quad \text{if} \quad [x(1,0) - x(0,0)] \neq 0, \tag{10}$$

$$s_0(\xi) = s_{y0}(\xi) \quad \text{else}, \tag{11}$$

where

$$s_{x0}(\xi) = \frac{x(\xi,0) - x(0,0)}{x(1,0) - x(0,0)}, \tag{12}$$

$$s_{y0}(\xi) = \frac{y(\xi,0) - y(0,0)}{y(1,0) - y(0,0)}. \tag{13}$$

Similar expressions for $s_1(\xi)$, $r_1(\eta)$ and $r_0(\eta)$ are given in Appendix A. To ensure that the interpolation is made right within the required domain, b_ξ and b_η must increase monotonically and vary between 0 and 1. Such a requirement can be met if the x-co-ordinate on the constant η boundaries increases (or decreases) monotonically from $\xi = 0$ to $\xi = 1$ and the y-co-ordinate on the constant ξ boundaries increases (or decreases) monotonically from $\eta = 0$ to $\eta = 1$. This is usually the case for many practical problems, or alternatively the requirement can be met by dividing the whole domain into several subdomains, and then generating the mesh for each subdomain.

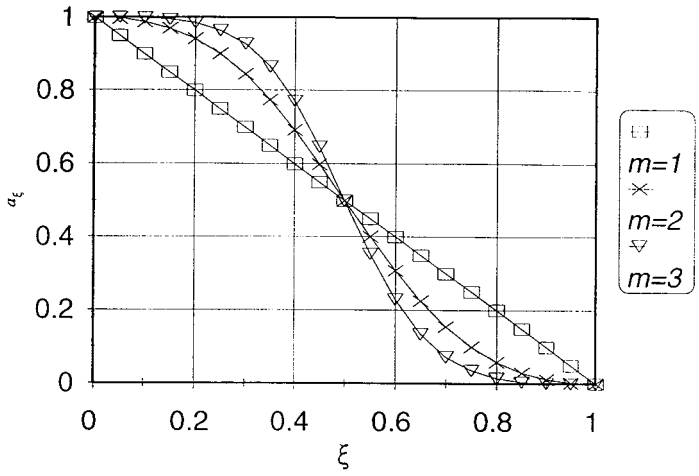

Figure 5. The effect of m on interpolation coefficient.

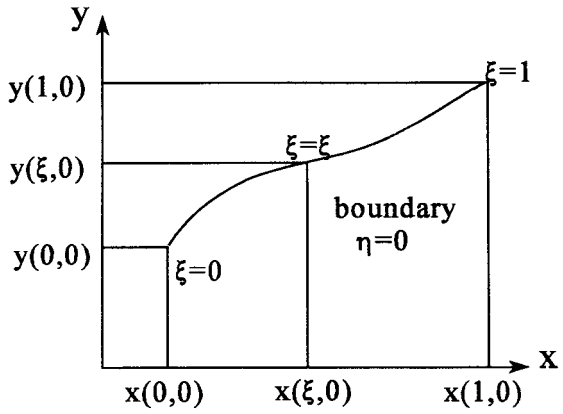

Figure 6. Boundary $\eta = 0$ and its projections on x- and y-co-ordinates.

Before applying the above scheme to the domain shown in Figures 2 and 3, let us first examine a simple square with uniform grid distributions on two vertical boundaries and non-uniform distributions on two horizontal ones. The significance of m on the grid generation is clearly demonstrated (Figure 7). In this particular case, using $m = 2$ has advantages because it gives orthogonal grids on all the boundaries. Figure 8 shows the grid generated using the present method for the domain shown in Figures 2 and 3. The boundary points were generated using an appropriate subroutine with uniform distributions on $J = 1$ ($\eta = 0$) and $J = NJ$ ($\eta = 1$) and the following distributions on $I = 1$ ($\xi = 0$) and $I = NI$ ($\xi = 1$). Here, I and J are indices of nodes in x- and y-co-ordinates, respectively, and NI and NJ are numbers of points in ξ- and η-directions, respectively.

On $I = 1$:

$$x(1, J) = 0; \qquad y(1, J) = 0.5 + 1.5((J-1)/(NJ-1))^{1.5}.$$

On $I = NI$:

$$x(NI, J) = 0.5 + 1.0((J-1)/(NJ-1))^{1.5}; \qquad y(NI, J) = 0.$$

Better gridding near the cylinder surface was achieved compared with that shown in Figure 2. The advantages of the present method over Shih *et al.*'s are that the former requires less computer time and the latter may produce overlapping. However, as described in Zhou [16], the present method also has disadvantages in that the orientation of the co-ordinate system may affect the inner gridding, although such effect is limited in practice. For complex domains with very distorted boundaries, the generation of the meshes in divided subdomains which are then linked together may also be required. Dividing the complex domains into several smaller and simpler subdomains has the advantage of minimising the effect of the co-ordinate system's orientation by selecting the localised co-ordinate system for each subdomain. Of course, in many cases, such as the domain shown in Figures 2 and 3, this is not required.

It is possible to add the derivatives of the boundary curves into Equations (4) and (5) and generate the locally orthogonal grid near the boundaries. Obviously, it will sacrifice the simplicity and computer time, and we leave it for future work. As for the smoothness of the grid generated using the present method, Equations (6)–(13) provided a smooth transformation of the stretching function from one boundary to the opposite, so that the inner grid distributions vary smoothly along the boundary node distributions. Such smoothness is not found in Equation (1).

Extending Equations (4) and (5) to three-dimensional cases, the following is obtained:

$$x(\xi, \eta, \zeta) = x(0, \eta, \zeta) + b_\xi[x(1, \eta, \zeta) - x(0, \eta, \zeta)] \quad 0 < \xi < 1, \tag{14}$$

$$y(\xi, \eta, \zeta) = y(\xi, 0, \zeta) + b_\eta[y(\xi, 1, \zeta) - y(\xi, 0, \zeta)] \quad 0 < \eta < 1, \tag{15}$$

$$z(\xi, \eta, \zeta) = z(\xi, \eta, 0) + b_\zeta[z(\xi, \eta, 1) - z(\xi, \eta, 0)] \quad 0 < \zeta < 1. \tag{16}$$

In the above equations, b_ξ, b_η and b_ζ are three stretching functions in the x-, y- and z-directions, respectively. They can have different forms, but again all must vary between 0.0

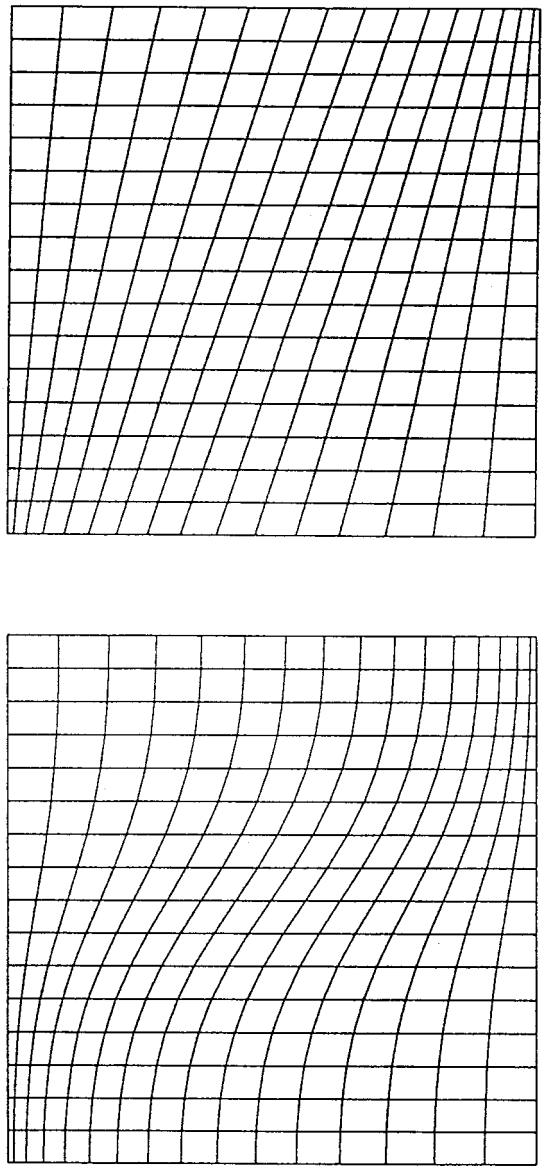

Figure 7. Grids generated for a square domain.

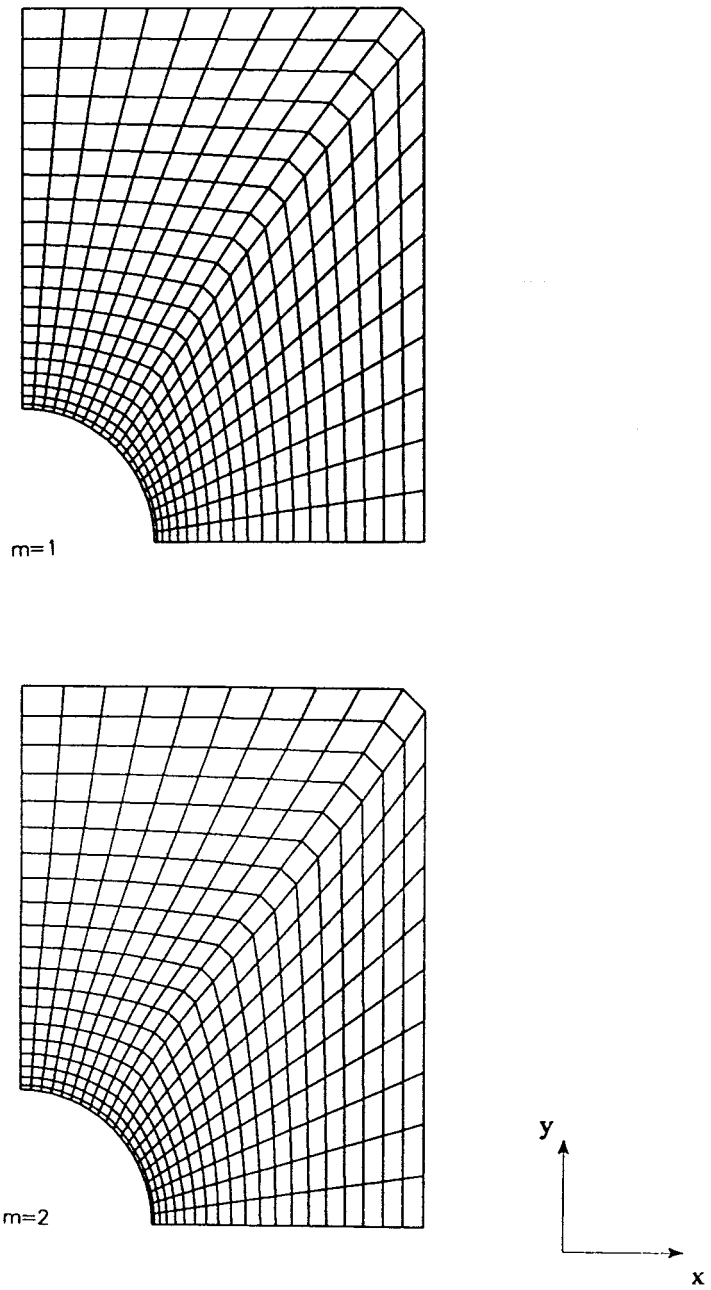

Figure 8. Grids generated using the present scheme.

and 1.0 and increase (or decrease) monotonically with ξ, η and ζ, respectively. The following examples were adopted in this study.

$$b_\xi = c_1 s(\eta = 1) + c_2 s(\eta = 0) + c_3 s(\zeta = 1) + c_4 s(\zeta = 0), \tag{17}$$

$$b_\eta = c_5 r(\xi = 1) + c_6 r(\xi = 0) + c_7 r(\zeta = 1) + c_8 r(\zeta = 0), \tag{18}$$

$$b_\zeta = c_9 t(\eta = 1) + c_{10} t(\eta = 0) + c_{11} t(\xi = 1) + c_{12} t(\xi = 0), \tag{19}$$

where, $c_1, c_2, \ldots, c_{12}$ are 12 weighted coefficients, and s, r and t represent the stretching functions on six boundary surfaces. Their detailed expressions are given in Appendix B. Similar to the two-dimensional case, three-dimensional problems must define the six surface boundaries properly in order to generate a proper grid.

The above algebraic grid generation scheme has been used for generating the inner grid for internal fluid flow simulations of agricultural flat fan nozzles [17–19]. According to Zhou *et al.* [17–19], the predicted spray performances, such as spray angle, flow rate and liquid flow distribution across the spray fan agreed well with the test data. Compared with certain built-in grid generators, the present procedure was much faster and needed less input [16].

3. CONCLUSION

A new, quick algebraic grid generation method is proposed in this paper. The new method has been used for generating meshes for internal fluid flow simulations of typical agricultural flat fan nozzles and satisfactory performances have been obtained with regard to the computer time and simulation results. The dependence of the grid quality on the selection of the orientation of the co-ordinate system would not limit its use significantly.

ACKNOWLEDGMENTS

The author thanks Dr P.C.H. Miller at Silsoe Research Institute and Dr N.H. Thomas at The University of Birmingham for their constant help in carrying out this work and preparing this manuscript. The author also thanks Silsoe Research Institute for funding the research studentship under which this work was done.

APPENDIX A: STRETCHING FUNCTION FOR A TWO-DIMENSIONAL DOMAIN

The mathematical expression for $s_1(\xi)$, $r_1(\eta)$ and $r_0(\eta)$ are given below.
For $s_1(\xi)$:

$$s_1(\xi) = s_{x1}(\xi) \quad \text{if} \quad [x(1, 1) - x(0, 1)] \neq 0$$

$$s_1(\xi) = s_{y1}(\xi) \quad \text{else}$$

where

$$s_{x1}(\xi) = \frac{x(\xi, 1) - x(0, 1)}{x(1, 1) - x(0, 1)}$$

$$s_{y1}(\xi) = \frac{y(\xi, 1) - y(0, 1)}{y(1, 1) - y(0, 1)}$$

For $r_0(\eta)$:

$$r_0(\eta) = r_{y0}(\eta) \quad \text{if} \quad [y(0, 1) - y(0, 0)] \neq 0$$

$$r_0(\eta) = r_{x0}(\eta) \quad \text{else}$$

where

$$r_{y0} = \frac{y(0, \eta) - y(0, 0)}{y(0, 1) - y(0, 0)}$$

$$r_{x0} = \frac{x(0, \eta) - x(0, 0)}{x(0, 1) - x(0, 0)}$$

For $r_1(\eta)$:

$$r_1(\eta) = r_{y1}(\eta) \quad \text{if} \quad [y(1, 1) - y(1, 0)] \neq 0$$

$$r_1(\eta) = r_{x1}(\eta) \quad \text{else}$$

where

$$r_{y1} = \frac{y(1, \eta) - y(1, 0)}{y(1, 1) - y(1, 0)}$$

$$r_{x1} = \frac{x(1, \eta) - x(1, 0)}{x(1, 1) - x(1, 0)}$$

APPENDIX B: WEIGHTED COEFFICIENTS AND STRETCHING FUNCTIONS FOR A THREE-DIMENSIONAL DOMAIN

The weighted coefficients in Equations (17)–(19) are

$$c_1 = \eta^m / T_\xi$$

$$c_2 = (1 - \eta)^m / T_\xi$$

$$c_3 = \zeta^m / T_\xi$$

$$c_4 = (1 - \zeta)^m / T_\xi$$

$$c_5 = \xi^m / T_\eta$$

$$c_6 = (1 - \xi)^m / T_\eta$$

$$c_7 = \zeta^m / T_\eta$$

$$c_8 = (1 - \zeta)^m / T_\eta$$

$$c_9 = \xi^m / T_\zeta$$

$$c_{10} = (1 - \xi)^m / T_\zeta$$

$$c_{11} = \eta^m / T_\zeta$$

$$c_{12} = (1 - \eta)^m / T_\zeta$$

and

$$T_\xi = \eta^m + (1 - \eta)^m + \zeta^m + (1 - \zeta)^m$$

$$T_\eta = \xi^m + (1 - \xi)^m + \zeta^m + (1 - \zeta)^m$$

$$T_\zeta = \xi^m + (1 - \xi)^m + \eta^m + (1 - \eta)^m$$

The 12 stretching functions are

$$s(\eta=1)=\frac{x(\xi,1,\zeta)-x(0,1,\zeta)}{x(1,1,\zeta)-x(0,1,\zeta)}$$

$$s(\eta=0)=\frac{x(\xi,0,\zeta)-x(0,0,\zeta)}{x(1,0,\zeta)-x(0,0,\zeta)}$$

$$s(\zeta=1)=\frac{x(\xi,\eta,1)-x(0,\eta,1)}{x(1,\eta,1)-x(0,\eta,1)}$$

$$s(\zeta=0)=\frac{x(\xi,\eta,0)-x(0,\eta,0)}{x(1,\eta,0)-x(0,\eta,0)}$$

$$r(\xi=1)=\frac{y(1,\eta,\zeta)-y(1,0,\zeta)}{y(1,1,\zeta)-y(1,0,\zeta)}$$

$$r(\xi=0)=\frac{y(0,\eta,\zeta)-y(0,0,\zeta)}{y(0,1,\zeta)-y(0,0,\zeta)}$$

$$r(\zeta=1)=\frac{y(\xi,\eta,1)-y(\xi,0,1)}{y(\xi,1,1)-y(\xi,0,1)}$$

$$r(\zeta=0)=\frac{y(\xi,\eta,0)-y(\xi,0,0)}{y(\xi,1,0)-y(\xi,0,0)}$$

$$t(\eta=1)=\frac{z(\xi,1,\zeta)-z(\xi,1,0)}{z(\xi,1,1)-z(\xi,1,0)}$$

$$t(\eta=0)=\frac{z(\xi,0,\zeta)-z(\xi,0,0)}{z(\xi,0,1)-z(\xi,0,0)}$$

$$t(\xi=1)=\frac{z(1,\eta,\zeta)-z(1,\eta,0)}{z(1,\eta,1)-z(1,\eta,0)}$$

$$t(\xi=0)=\frac{z(0,\eta,\zeta)-z(0,\eta,0)}{z(0,\eta,1)-z(0,\eta,0)}$$

REFERENCES

1. J.F. Thompson, Z.U.A. Warsi and C.W. Mastin, *Numerical Grid Generation Foundations and Applications*, North-Holland, Amsterdam, 1985.
2. J.F. Thompson, F.C. Thames and C.W. Mastin, 'Automatic numerical generation of body-fitted curvilinear coordinate system for field containing any number of arbitrary two dimensional bodies', *J. Comput. Phys.*, **15**, 299–319 (1974).
3. R.E. Smith and L.-E. Eriksson, 'Algebraic grid generation', *Comput. Meth. Appl. Mech. Eng.*, **64**, 285–300 (1987).
4. J.F. Thompson, 'A general three-dimensional elliptic grid generation system on a composite block structure', *Comput. Meth. Appl. Mech. Eng.*, **64**, 377–411 (1987).
5. J.L. Steger and J.A. Benek, 'On the use of composite grid schemes in computational aerodynamics', *Comput. Meth. Appl. Mech. Eng.*, **64**, 301–320 (1987).
6. J. Zhu, 'A hybrid differential-algebraic method for three-dimensional grid generation', *Int. J. Numer. Methods Eng.*, **29**, 1271–1279 (1990).
7. P.R. Eiseman, 'Adaptive grid generation', *Comput. Meth. Appl. Mech. Eng.*, **64**, 321–376 (1987).
8. J.F. Thompson, 'Composite grid generation code for general 3-D region—The EAGLE code', *AIAA J.*, **26**, 271–272 (1988).

9. R.L. Sorenson, 'Three-dimensional zonal grids about arbitrary shapes by Poisson's equation', *Numerical Grid Generation in Computational Fluid Mechanics'88*, Pineridge, Swansea, UK, 1988.
10. E.W. Dorrel and M.D. McClure, '3DEINGRID: Interactive three-dimensional grid generation', *AEDC-TR-87-40*, 1988.
11. T.I.-P. Shih, R.T. Bailey, H.L. Nguyen and R.J. Roelke, 'Algebraic grid generation for complex geometries', *Int. J. Numer. Methods Fluids*, **13**, 1–31 (1991).
12. S.A. Jordan and M.L. Spaulding, 'A fast algorithm for grid generation', *J. Comput. Phys.*, **104**, 118–128 (1993).
13. W.J. Gordon, 'Blending-function methods for bivariate and multivariate interpolation and approximation', *SIAM J. Numer. Anal.*, **8**, 158–177 (1971).
14. W.J. Gordon and C.A. Hall, 'Construction of curvilinear co-ordinate systems and applications to mesh generation', *Int. J. Numer. Methods Eng.*, **7**, 461–477 (1973).
15. N.P. Weatherill and H. Deconinck, *Numerical Grid Generation*, Lecture Series 1990-06, von Karman Institute for Fluid Dynamics, 1990.
16. Q. Zhou, 'Spray formation processes within agricultural flat fan nozzles', *Ph.D. Thesis*, The University of Birmingham, UK, 1997.
17. Q. Zhou, P.C.H. Miller, P.J. Walklate and N.H. Thomas, 'Spray angle prediction of flat fan nozzles', *J. Agri. Engng. Res.*, **64**, 139–148 (1996).
18. Q. Zhou, P.C.H. Miller, P.J. Walklate and N.H. Thomas, 'Droplet formation of a hydraulic flat fan nozzle with an internally disturbed flow', *Proc. of 12th Ann. Conf. of ILASS-Europe on Spray and Atomisation*, Lund, Sweden, 1996.
19. Q. Zhou, P.C.H. Miller, P.J. Walklate and N.H. Thomas, 'Numerical and experimental study of the effect of flat fan nozzle geometry on spray performance', *Numerical Methods in Laminar and Turbulent Flow '95*, **9**, 1645–1656 (1995).

