SENSITIVITY ANALYSIS AND OPTIMIZATION OF AERODYNAMIC CONFIGURATIONS WITH BLEND SURFACES

by

A. M. Thomas
Graduate Research Assistant
and
S. N. Tiwari
Principal Investigator

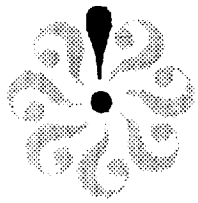
Final Report
For the Period ending December 31, 1996

Prepared for

National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23681-0001

Under

Research Grant NCC1-68
Dr. Robert E. Smith, Jr.
Technical Monitor
ISSD-Scientific Application Branch



August 1997

Department of Mechanical Engineering • College of Engineering & Technology
Old Dominion University • Norfolk, Virginia 23529-0247

SENSITIVITY ANALYSIS AND OPTIMIZATION OF AERODYNAMIC CONFIGURATIONS WITH BLEND SURFACES

by

A. M. Thomas
Graduate Research Assistant

and

S. N. Tiwari
Principal Investigator

Final Report
For the Period ending December 31, 1996

Prepared for

National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23681-0001

Under

Research Grant NCC1-68
Dr. Robert E. Smith, Jr.
Technical Monitor
ISSD-Scientific Application Branch

August 1997

Department of Mechanical Engineering • College of Engineering & Technology• Old Dominion University • Norfolk,Virginia 23529-0247

SENSITIVITY ANALYSIS AND OPTIMIZATION OF AERODNAMIC CONFIGURATIONS WITH BLEND SURFACES

ABSTRACT

A novel (geometrical) parametrization procedure using solutions to a suitably chosen fourth order partial differential equation is used to define a class of airplane configurations. Inclusive in this definition are surface grids, volume grids, and grid sensitivity. The general airplane configuration has wing, fuselage, vertical tail and horizontal tail. The design variables are incorporated into the boundary conditions, and the solution is expressed as a Fourier series. The fuselage has circular cross section, and the radius is an algebraic function of four design parameters and an independent computational variable. Volume grids are obtained through an application of the Control Point Form method. A graphic interface software is developed which dynamically changes the surface of the airplane configuration with the change in input design variable. The software is made user friendly and is targeted towards the initial conceptual development of any aerodynamic configurations. Grid sensitivity with respect to surface design parameters and aerodynamic sensitivity coefficients based on potential flow is obtained using an Automatic Differentiation precompiler software tool ADIFOR. Aerodynamic shape optimization of the complete aircraft with twenty four design variables is performed. Unstructured and structured volume grids and Euler solutions are obtained with standard software to demonstrate the feasibility of the new surface definition.

ACKNOWLEDGEMENTS

This is a final report on the research project "Surface Modeling and Optimization Studies of Aerodynamic Configurations." Within the guidelines of the project, special attention was directed toward research activities in the area of "Sensitivity Analysis and Optimization of Aerodynamic Configuration with Blend Surfaces." The period of performance of this specific research ended December 31, 1996.

The authors with to express their thanks to Drs. Arthur C. Taylor and Gene J. W. Hou of Old Dominion University, Professors Malcolm I. G. Bloor and Michael J. Wilson of the University of Leeds, U.K., and Drs. Jamsid S. Abolhassani and Michael K. Bockelie of Computer Science Corporation for their assistance during the entire period of this study. Dr. Robert Weston of NASA Langley Research Center provided the X-window graphics program XGEOM and the potential flow code PMARC used in this study.

This work, in part, was supported by the NASA Langley Research Center through Cooperative Agreement NCC1-68. The cooperative agreement was monitored by Dr. Robert E. Smith, Jr. of ISSD-Scientific Applications Branch, NASA Langley Research Center, Mail Stop 125.

TABLE OF CONTENTS

LIST OF TABLES

REFRENCES

1. Sloof, J.W., "Computational Methods for Subsonic and Transonic Aerodynamics Design," AGARD, AGARD Report 712, Paper 3, 1983.

2. Raymer, D.P., "*Aircraft Design: A Conceptual Approach*," AIAA, Washington, 1989.

3. Labrujere, TH.E., and Sloof, J.W., "Computational Methods for the Aerodynamic Design of Aircraft Components," Annu. Rev. Fluid Mech., Vol. 25, September 1993, pp. 183-214.

4. Bezier, P., "*Numerical Control: Mathematics and Applications*," John Wiley and Son, 1970.

5. Tiller, W., "Rational B-Splines for Curve and Surface Representation," *Computer Graphics and Applications*, Volume 3, N0. 10, September 1983, pp. 61-69.

6. Blomgren, R.M., "B-Spline Curves," Class notes, Boeing document B-7150-BB-WP-2811D-4412, 1981.

7. Lee, E.T.Y., "A Treatment of Conics in Parametric Rational Bezier Form," Boeing document, Boeing, Seattle, February 1981.

8. "Initial Graphics Exchange Specification, Version 3.0," Doc. No. NB-SIR 86-3359, NIST, Gaithersburg, Md., 1986.

9. Van Dam, A., "PHIGS+, Functional Description Revision 3.0," *Computer Graphics*, Vol. 22, No. 3, July 188, pp. 125-218.

10. Bloor, M.I.G., and Wilson, M.J., "Generating Blend Surfaces Using Partial Differential Equations," *Computer Aided Design*, Vol. 21, No. 3, July 1989, pp. 165-171.

11. Bloor, M.I.G., and Wilson, M.J., "Using Partial Differential Equations to Generate Free-form Surfaces", *Computer Aided Design*, Vol. 22, No. 5, August 1990, pp. 202-212.

12. Bloor, M.I.G., and Wilson, M.J., "Representing PDE Surfaces in Terms of B-Splines," *Computer Aided Design*, Vol. 22, No. 8, Feb. 1990, pp. 324-331

13. Margason, R.J., Kjelgaard, S.O., Sellers, W.L., Morris, C.E.K., Walkey, K.B., and Shields, E.W., "Subsonic Panel Methods-A Comparison of Several Production Codes," AIAA Paper 85-0280, January 1985.

14. Thompson, J.F., Warsi Z.U.A., and Mastin, C.W., "*Numerical Grid Generation*," North-Holland, 1985

15. Thompson, J.F., "Grid Generation Techniques in Computational Fluid Dynamics", *AIAA Journal*, Vol. 22, No. 11, 1984, pp. 1505-1523.

16. Eiseman, P.R., "A Multi-Surface Method of Coordinate Generation," *J. Comp. Physics*, Vol. 33, No. 1, 1979, pp. 118-150.

17. Eriksson, L.E., "Generation of Boundary-Conforming Grids Around Wing-Body Configurations Using Transfinite Interpolation," *AIAA Journal*, Vol. 20, No. 10, 1982, pp. 1313-1320.

18. Baker, T.J., "Mesh Generation by a Sequence of Transformations," *Applied Numerical Mathematics*, Vol. 2, No. 1, December 1986.

19. Weatherhill, N.P., and Forsey, C.R., "Grid Generation and Flow Calculations for Aircraft Geometries," *Journal of Aircraft*, Vol. 22, No. 10, October 1985, pp. 855-860.

20. Benek, J.A., Buning, P.G., and Steger, J.L., "A 3-D Chimera Grid Embedding Technique," *AIAA 7th Computational Fluid Dynamics Conference*, Cincinnati, Ohio, June 1985, AIAA Paper 85-1523.

21. Smith, R.E., Eriksson, L.E., and Everton, E., "Algeabric Grid Generation Around Complex 3-D Aircraft Configurations," AIAA Paper 90-3425, 1990.

22. Vigneron, Y., and Lejal, T., "Calculation of Transonic Flow around an Aircraft Configuration with Motorized Nacalle," ICAS Paper 84-2.10.2.

23. Sawada, K., Takanashi, S., "A Numerical Investigation on Wing/Nacelle Interferences of USB Configuration," AIAA Paper 87-0455, January 1987.

24. Baker, T.J., "Three-Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets," AIAA 87-1124, January 1987.

25. Lohner, R.L., and Parikh, P., "Generation of Three-Dimensional Unstructured Grids by the Advancing-Front Method," *International Journal for Numerical Methods in Fluids*, Vol. 8, April 1988, pp. 1135-1149.

26. Jameson, A., Baker, T.J., and Weatherhill, N.P., "Calculation of Inviscid Transonic Flow Over a Complete Aircraft," AIAA-86-0103, 1986.

27. Mavriplis D., and Jameson, A., "Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes," AIAA-87-0353, 1987.

28. Lohner, R., Morgan, K., and Zienkiewicz, O.C., "Adaptive Grid Refinement for Compressible Euler Equations," *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*, Edited by Babuska et. al., Wiley, July 1986, pp. 281-297.

29. Formaggia, L., Peraire, J., Morgan, K., and Peiro, J., "Implementation of a 3D explicit Euler Solver on CRAY Computer," *Proceedings of 4th International Symposium on Science and Engineering on CRAY Supercomputers*, Minneapolis, May 1988, pp. 45-65.

30. Sobieszczanski-Sobieski, J. "The Case for Aerodynamic Sensitivity Analysis," Paper presented to NASA/VPI SSY Symposium on Sensitivity Analysis in Engineering, September 25-26, 1986.

31. Yates, E.C., "Aerodynamic Sensitivities from Subsonic, Sonic, and Supersonic Unsteady, Nonplanar Lifting-Surface Theory," NASA TM-100502, September 1987.

32. Murthy, D.V., and Kaza, K.V., "Application of a Semianalytical Techniques for Sensitivity Analysis of Unsteady Aerodynamic Computations," AIAA Paper 88-2377, April 1988.

33. Grossman, B., Haftka, R.T., Kao, P.J., Polen, D.M., Rais-Rohani, M., Sobieszcz anski-Sobieski, J., "Integrated Aerodynamic-Structural Design of a Transport Wing," AIAA Paper 89-2129, AIAA/AHS/ASEE Aircraft Design, Systems and Operations Conference, Seattle, WA, July 31 - August 2, 1989.

34. Livne, E., Schmit, L.A., and Friedmann, P., "An Integrated Approach to the Optimum Design of Actively Controlled Composite Wings," AIAA Paper 89-1268, *Proceedings of the AIAA/ASME/ASCE/AHS/ASC 30th Structural, Structural Dynamics and Materials Conference*, Mobile AL, April 3-5, 1989, Part II, pp. 933-945.

35. Elbanna, H., and Carlson, L., "Determination of Aerodynamic Sensitivity Coefficients in the Transonic and Supersonic Regimes," AIAA Paper 89-0532, January 1989.

36. Elbanna, H., and Carlson, L., "Determination of Aerodynamic Sensitivity Based on the Three-Dimensional Full Potential Equations," AIAA Paper 92-2670, January 1992.

37. Taylor, A. C., III, Hou, G. W., and Korivi, V. M., "Sensitivity Analysis Applied to the Euler Equations : A Feasibility Study with Emphasis on Variation of Geometric Shape," AIAA Paper 91-0173, January 1991.

38. Hou, G. W., Taylor, A. C., III, and Korivi, V. M., "Discrete Shape Sensitivity Equations for Aerodynamic Problems," AIAA Paper 91-2259, June 1991.

39. Baysal, O., and Eleshaky, M. E., "Aerodynamic Sensitivity Analysis Methods for the Compressible Euler Equations," *Recent Advances in Computational Fluid Dynamics*, (ed. O. Baysal), ASME-FED Vol. 103, 11th Winter Annual Meeting, November, 1990, pp. 191-202.

40. Taylor, A. C., III, Hou, G. W., and Korivi, V. M., " Sensitivity Analysis, and Design Optimization For Internal and External Viscous Flows," AIAA Paper 91-3083, September 23-25, 1991, Baltimore, MD.

41. Baysal, O., and Eleshaky, M. E., "Aerodynamic Design Optimization Using Sensitivity Analysis and Computational Fluid Dynamics," AIAA Paper 91-0471, January 1991.

42. Eleshaky, M.E., and Baysal, O., "Aerodynamic Shape Optimization Via Sensitivity Analysis on Decomposed Computational Domains," 4th AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization, Part 1, September 21-23, 1992, Cleveland, OH, pp. 98-109.

43. Korivi, V. M., Taylor, A. C., III, Newman, P.A., Hou, G. W., and Jones, H.E., "An Approximately Factored Incremental Strategy For Calculating Consistent Discrete Aerodynamic Sensitivity Derivatives," AIAA Paper 92-4746, September 1992.

44. Newman, P.A., Hou, G. W., Jones, H.E., Taylor, A. C., III, and Korivi V.M., "Observations on Computational Methodologies for Use in Large-Scale, Gradient-Based, Multidisciplinary Design," AIAA Paper 92-4753, September 1992.

45. Korivi, V. M., Taylor, A. C., III, Hou, G. W., Newman, P.A., and Jones, H.E., "Sensitivity Derivatives for Three-Dimensional Supersonic Euler Code Using Incremental Iterative Strategy," AIAA CP-933, *AIAA 11th Computational Fluid Dynamics Conference*, July 1993, pp. 1053-1054.

46. Bischof, C., and Griewank, A., "ADIFOR: A Fortran System For Portable Automatic Differentiation," AIAA Cp-9213, Fourth AIAA/USA/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization, September 1992, pp. 433-441.

47. Green, L.L., Newman, P.A., and Haigler, K.J., "Sensitivity derivatives for Advanced CFD Algorithm and Viscous Modelling Parameters Via Automatic Differentiation," AIAA Cp-933, *AIAA 11th Computational Fluid Dynamics Conference*, July 1993, pp. 260-277.

48. Vatsa, V. N., and Wedan, B. W., "Development of a Multigrid Code for 3-D Navier-Stokes Equations and its Application to a Grid-Refinement Study," *Computers and Fluids*, Vol. 18, No. 4, 1990, pp. 391-403.

49. Tiller, W. "Geometric Modelling Using Non-Uniform Rational B-Splines, Methematical Techniques," Siggraph tutorial notes, ACM, New York, 1986.

50. Bloor, M. I.G., and Bloor, M. J., "Generating Blend Surfaces Using Partial Differential Equations," *Computer Aided Design*, Vol. 21, No. 3, April 1989, pp. 165-170.

51. Somayajula, G., "Grid sensitivity analysis," *Finite Element in Analysis and Design*, Elsevier Science Publishers B.V., 1991, pp. 307-315.

52. Sadrehaghighi, I., Smith, R.E., Tiwari, S.N., "An Analytical Approach To Grid Sensitivity Analysis," AIAA Paper 92-0660, 30th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 6-9, 1992.

53. Smith, R. E., and Sadrehaghighi, I., "Grid Sensitivity in Airplane Design," *Proceedings of the fourth International Symposium on Computational Fluid Dynamics*, Vol. 1, September 9-12, 1991, Davis, California, pp. 1071-1076.

54. Farin, G., *Curves and Surfaces for Computer-Aided Geometric Design A Practical Guide*, Academic Press, 1989.

55. Nicolai, L., *Fundamentals of Airplane Design*, Distributed by the University of Dayton, Dayton, OH, 1975.

56. Coons, S., "Surfaces for Computer Aided Design," Technical Report, MIT, Project MAC-TR41, 1967.

57. Smith, R. E., and Kerr, P. A., "Geometric requirements for Multidisciplinary Analysis of Aerospace Vehicle Design", AIAA Paper No. 92-4773, September 1992.

58. Piegle, L., "On NURBS: A Survey," *IEEE Computer Graphic and Application*, Volume II, No. 1, January 1991, pp. 55-71.

59. Sadrehaghighi, I., Smith, R.E., Tiwari, S.N., "Grid and Design Variables Sensitivity Analysis For NACA Four-Digit Wing-Sections," AIAA Paper 93-0195, 30th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 11-14, 1993.

60. Burgreen, G.W., Baysal, O., and Eleshaky, M. E., "Improving the Efficiency of Aerodynamic Shape Optimization Procedures ," AIAA Paper 92-4697, 4th AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization, Part 1,September 21-23, 1992, Cleveland, OH, pp. 87-97.

61. Baals, D.D., Robin, A. W., Harris, R. V., "Aerodynamic Design Integration of Supersonic Aircraft," *Journal of Aircraft*, Vol. 7, No. 5, 1970, pp. 385-394.

62. Robins, A. W., Dollyhigh, S. M., Beissner, F. L., Geiselhart, K., Martin, G.L., Shields, E. W., Swanson, E. E., Coen, P. G., and Morris, S. J., "Concept Development of a Mach 3.0 High-Speed Civil Transport," NASA TM 4058, September 1988.

63. Tiller, W., "Geometric Modelling Using Non-Uniform rational B-Splines," Siggraph Tutorial Notes, ACM, New York, 1986.

64. Hutchison, M.G. , Mason, W.H., Grossman, B., and Haftka, R.T., "Aerodynamic Optimization of an HSCT Configuration Using Variable-Complexity Modeling," AIAA Paper 93-0101, 31th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 11-14, 1993.

65. Piegl, L., and Tiller, W., "Curve and Surface Construction Using Rational B-Splines," *Computer Aided Design*, Vol. 19, No. 9, November 1987, pp. 485-498.

66. Woodward, C. D., "Skinning Techniques for Interactive B-Spline Surface Interpolation," Vol. 20, No. 8, October 1988, pp. 441-451.

67. Smith, R. E., Bloor, M. I. G., Wilson, M. J., and Thomas, A. M., "Rapid Airplane Parametric Input Design (RAPID)," AIAA-95-1687, June, 1995.

68. Beck, J. M., Farouki, R. T., Hinds, J. K., "Surface Analysis Methods," *IEEE CGA*, December 1986, pp. 18-35.

69. Overmars, M. H., "Forms Library A Graphical User Interface Toolkit for Silicon Graphics Workstation," Department of Computer Science, Utrecht University, Netherlands, November 1992.

70. Abolhassani, J.S., Sadrehaghighi, I., Smith, R.E., and Tiwari, S.N., "Application of Lagrangian Blending Function for Grid Generation Around Airplane Geometries," *Journal of Aircraft*, Volume 27, No. 10, October 1990, pp. 873-877.

71. Huband, G. W., Shang, J. S., and Aftosmis, M. J., "Numerical Simulation of an F-16A at Angle of Attack," *J. Aircraft*, Vol. 27, October 1990, pp. 886-892.

72. Corcoran, E., "Calculating Reality," *Scientific American*, January 1990, pp. 101-109.

73. Thomas, J.L., and Salas, M.D., "Far-Field Boundary Conditions for Transonic Lifting Solutions to the Euler Equations," AIAA Paper 85-0020, AIAA 23rd Aerospace Sciences Meeting, Reno NV, January 14-17, 1985.

74. Thompson, J.F., Warsi, Z.U.A., and Mastin, C.W., *Numerical Grid Generation: Foundations and Applications*, North-Holland, NewYork, 1985.

75. Gordon, W. N., and Hall, C. A., "Construction of Curvilinear Coordinate Systems and Application to Mesh Generation," *International Journal for Numerical Methods in Engineering*, Vol. 7, 1973, pp 461-477.

76. Eiseman, P.R., "A Multi-Surface Method of Coordinate Generation," *Journal of Computational Physics*, Volume 33, No. 1, 1979, pp. 118-150.

77. Smith, R.E., "Two-Boundary Grid Generation for Solution of the Three - Dimensional Navier-Stokes Equations," NASA TM-83123, May 1981.

78. Sorenson, R., "A Computer Program to Generate Two-Dimensional Grids About Airfoils and Other Shapes by the use of Poisson's Equation," NASA TM-81198, May 1980.

79. Thompson, J., "A Composite Grid Generation Code for General 3D Regions-the EAGLE Code," *AIAA Journal*, Volume 26, No. 3, March 1988.

80. Steinbrenner, J., Chawner, J., and Fouts, C., "The GRIDGEN 3D Multiple Block Grid Generation System," WRDC TR 90-3022, July 1990.

81. Akdag, V., and Wulf, A., "Integrated Geometry and Grid Generation System for Complex Configurations," NASA CP 3143, Apri 1992.

82. Peraire, S., and Morgan, K., "A General Triangular Mesh Generation," *International Journal of Numerical Methods*, Vol 31, No. 4, November 1987, pp. 61-72.

83. Watson, D. F., "Computing the N-Dimensional Delauny Tesselation With Application to Voronoi Polytopes," *Computational Journal*, Vol. 24, No. 2, May 1981, pp. 167-172.

84. Yerry, M. A., and Shepard, M. S., "Automatic Three-Dimensional Mesh Generation by the Modified Octree Technique," *International Journal of Numerical Methods*, Vol. 20, December 1984, pp. 1965-1990.

85. Phai, V. N., "Automatic Mesh Generation With Tetrahedron Elements," *International Journal of Numerical Methods*, Vol. 18, November 1982, pp. 237-289.

86. Lohner, R, "Some Useful Data Structures For The Generation of Unstructured Grids," *Computational Applied Numerical Methods*, Vol. 4, January 1988, pp. 123-135.

87. Eiseman, P. R., "A Multi-Surface Method of Coordinate Generation," *Journal of Computational Physics*, Vol. 33, August 1979, pp. 118-129.

88. Eiseman, P. R., and Smith, R. E., "Applications of Algebraic Grid Generation , Applications of Mesh Generation to Complex 3-D Configurations," *Advisory Group for Aerospace Research and Development* (AGARD)-CP-464, 1989, pp. 4-12.

89. Smith, R. and Everton, E., "Interactive Grid Generation for Fighter Aircraft Geometries," *Numerical Grid Generation in Computational Fluid Mechanics '88*, Pine Ridge Press Ltd., 1988, pp. 805-814.

90. Abolhassani, J.S., and Smith, R.E., "Three-Dimensional Grid Generation About a Submarine," *Numerical Grid Generation in Computational Fluid Mechanics*, Pineridge Press Limited, Swansea, UK, 1988, pp. 505-515.

91. Eiseman, P. R., and Smith, R. E., "Applications of Algebraic Grid Generation Applications of Mesh Generation to Complex 3-D Configurations," AGARD-CP-464, 1989, pp. 4-12.

92. Farin, G., *Curves and Surfaces for Computer-Aided Geometric Design A Practical Guide*, Academic Press, 1994.

93. Baker, T. J., "Three Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets," AIAA-87-1124, May 1987, pp. 255-270.

94. Pirzadeh, S., "Structured Background Grids For Generation of Unstructured Grid by Advancing Front Method," AIAA-91-3233, September 1991.

95. Frink, N. T., Parikh, P., and Pirzadeh, S., "A Fast Upwind Solver for the Euler Equations on Three Dimensional Unstructured Meshes," AIAA-91-0102, January 1991.

96. Frink, N. T., Parikh, P., and Pirzadeh, S., "An Adaptive Remeshing Procedure For Three-Dimensional Unstructured Grids," AIAA-91-3292, January 1991.

97. Ashby, D. L., Dudley, M. R., Iguchi, S. K., Browne, L., and Katz, J., "Potential Flow Theory and Operation Guide For the Panel Code PMARC," NASA-TM-102851, January 1991.

98. Sobieszczanski-Sobieski, J. "Sensitivity Analysis and Multidisciplinary Optimization for Aircraft Design : Recent Advances and Results," NASA TM-100630, July 1988.

99. Cramer E., Frank, P., shubin, G., Dennis, J., and Lewis, R., "On Alternative Problem Formulation for Multidisciplinary Optimization,"AIAA-92-4752, September 1992.

100. Korivi, V. M., Taylor III, A. C., Hou, G. W., Newman, P. A., and Jones, H. E., "Sensitivity Derivatives for Three-Dimensional Supersonic Euler Code Using Incremental Iterative Strategy," AIAA CP-33, *AIAA* 11^{th} *Computational Fluid Dynamics Conference*, July 1993, pp. 1053-1054.

101. Carle, A., Cooper, K., Hood, R., Kennedy, K., Torczon, L., and Warren, S. K., "A Practical Environment for Scientific Programming," *IEEE Computer*, Vol. 20, No. 11, November 1987, pp. 75-89.

102. Bischof, C. H., Carle, A., Corliss, G. F., Griewank, A., and Hovland, P., "ADIFOR: Generating Derivative Codes From Fortran Programs," *Scientific Programming*, Vol. 1, No. 1, December 1992, pp. 1-29.

103. Green, L., Newman, P. A., Haigler, K. J., "Sensitivity Derivatives for Advanced CFD Algorithm and Viscous Modelling Parameters vis Automatic Differentiation," AIAA-93-3321, July 1993.

104. Somayajula, G., "Grid sensitivity analysis," *Finite Element in Analysis and Design*, Elsevier Science Publishers B.V., 1991, pp. 307-315.

105. Smith, R. E., and Sadrehaghighi, I., "Grid Sensitivity in Airplane Design," *Proceedings of the fourth International Symposium on Computational Fluid Dynamics*, Vol. 1, September 9-12, 1991, Davis, California, pp. 1071-1076.

106. Vanderplaats, G.N., " An Efficient Feasible Direction Algorithm for Design Synthesis," *AIAA Journal*, Volume 22, No. 11, October 1984, pp. 1633-1640.

107. Vanderplaats, G.N., " ADS - A Fortran Program for Authomated Design Synthesis," NASA CR-177985, September 1985.

108. Jones, W., and Abolhassani, S. J., "A Grid Generation System for Multi-disciplinary Design Optimization," AIAA-95-1689, June 1995.

109. Abbott, I.H., and Von Doenhoff, A.E., *Theory of Wing Sections*, Dover, New York, 1959.

LIST OF FIGURES

LIST Of SYMBOLS

a = local speed of sound

C = location of maximum camber

C_D = drag coefficient

C_f = skin friction coefficient

C_L = lift coefficient

C_P = pressure coefficient

C_X, C_Y = force coefficients in x and y directions

$\mathbf{D}_i$ = NURBS control point coordinate

e = energy per unit volume

F = physical model

f = objective function

G = dependent parameter

$\bar{\mathbf{F}}, \bar{\mathbf{G}}$ = inviscid fluxes

$\bar{\mathbf{G}}_v$ = viscous flux vector

g = optimization constraints

H = independent parameter

J = jacobian of transformation

M = maximum camber

m = number of knots of a NURBS curve

M_∞ = free-stream Mach number

$N_{i,p}(r)$ = B-spline basis function

n = number of control points of NURBS

$\mathbf{n}$ = unit normal vector

$\mathbf{P}$ = vector of independent parameters

P_i = local pressure

p = degree of a NURBS curve

$\mathbf{Q}$ = vector of field variables

$\mathbf{Q}^*$ = steady-state field variables

$R_{i,p}(r)$ = NURBS basis function

Re_∞ = free-stream Reynolds number

$\mathbf{r}$ = uniform knot vector of NURBS

T = maximum thickness

t = grid stretching parameter

$\mathbf{X}_D$ = vector of design parameters

x, y, z = physical coordinates

x_1, y_1 = surface coordinates

y_c, y_T = camber and thickness curve ordinates

$\mathbf{a}$ = Vector constant for Fourier expression

$\mathbf{b}$ = Vector constant for Fourier expression

B_t = Root chord length for tail components

B_w = Root chord length for wing component

B_c = Root chord length for canard component

C = Wing chord length at crank

E = Wingtip chord length

F_1, F_2 = Parameters for airfoil definition

H_1 = Inboard wing span length

H_2 = Outboard wing span length

K_1, K_2 = Constants for grid spacing control

K_3, K_4 = Constants for grid spacing control

M = Maximum wing camber

P = Location of maximum wing camber

R_F = Fuselage length

R_0, R_1 = Parameters for fuselage radius

R_2 = Parameter for radius at rearmost point

S_1, S_2 = Derivative control design parameters

T = Maximum wing thickness

T_a = Wing taper parameter

X_t, Z_t = Coordinates of trailing tip point

X_w, Z_w = Coordinates of trailing wing point

X_c, Z_c = Coordinates of trailing crank point

a = PDE weighting factor

r = Fuselage radius

$\bar{x}$ = Airfoil independent variable

$\bar{y}$ = Airfoil dependent variable

$\bar{y}_c$ = Wing camber

$\bar{y}_t$ = Wing thickness

Greek Symbols

α = angle of attack

γ = scalar move parameter

δ = Kronecker delta

ξ, η, ζ = computational coordinates

ρ = density

τ_i = local shear stress

τ = viscous stress term

ω_i = NURBS curve weighting parameter

σ = Coordinate weighting parameter

ξ, η, ζ = Computational coordinates

$\bar{\xi}, \bar{\eta}, \bar{\zeta}$ = Computational coordinates

ν = Grid spacing control coordinate

$\bar{\nu}$ = Grid spacing control coordinate

θ = Fuselage definition variable

$\mathcal{P}$ = Set of design parameters

$\mathcal{K}$ =Set of grid control parameters

Chapter 1

INTRODUCTION

1.1 Motivation

Design and optimization of airplane components has become a primary objective for most researchers in aerodynamic community. The sudden interest can be attributed to the introduction of complex and composite materials required by advanced aerospace vehicles, such as National Aerospace Plane (NASP) and High Speed Civil Transport (HSCT) aircraft. Here, the interdisciplinary interactions are particularly important because of extreme flight conditions. The design of such vehicles requires many analyses over a wide range of engineering disciplines.

In the past, design of flight vehicles typically required the interaction of many technical disciplines over an extended period of time in a more or less sequential manner. At present, computer-automated discipline analyses and interactions offer the possibility of significantly shortening the design cycle time, while simultaneous multidisciplinary design optimization (MDO) via formal sensitivity analysis (SA) holds the possibility of improved designs. Each analysis is based on solving mathematical models describing physical laws associated with a discipline. The mathematical models are systems of algebraic, differential, or integral equations which are solved on discrete domains called "grids" on, around, and interior to the vehicle surface. The geometric requirements are the definition of the vehicle surface and the generation of grids onto which solutions of the mathematical models are obtained. In the optimization of aerospace-vehicle designs, engineering disciplines are interconnected and

affect one another. The effects can be realized in two ways: (1) The output from one discipline is the input to another. (2) The vehicle geometry changes in response to a discipline, therefore affecting other disciplines. In multidisciplinary analysis, the vehicle surface remains constant and all disciplines analyze their physics based on the same surface. Whereas, in multidisciplinary optimization the vehicle surface must be allowed to change. A complete design and optimization analysis using all the relevant disciplines is still a formidable task even for an isolated airplane component such as a wing or fuselage. The computational cost associated with such analysis can easily strain the capabilities of current supercomputers. The magnitude of this problem can be best appreciated when a discrete aerodynamic or structural design analysis can exhaust the computational capability of a medium size supercomputer. The underlying problem is the expensive cost of the analysis for each discipline involved. Clearly the aerodynamics involve non-linear physics and use of composite materials would require non-linear structural analysis as well. For a simple aeroelastic problem, the entire system matrix must be simultaneously solved using mostly implicit solvers. The extensive computational demand for such coupling of the governing equations, will likely limit MDO to only individual components such as a wing or wing-section. The cost of optimization operations are relatively small and manageable. Two general directions to overcome these difficulties have been proposed by different research groups. The first direction leads toward modifying the existing computational tools in order to obtain a relatively cheap and reliable technique for design and optimization. The usually favored direct solvers, with all their advantages, require extremely large computer storage even for 2D applications.

Creating an airplane surface or any other object surface with design parameters implies that there is an underlining set of rules or correspondences (model functions) that are driven by the parameters and independent computational variables. Surfaces grids are discrete evaluations of the surface functions, and surface

grids can be described as organized sets of points. Different discipline analyses and different techniques within a discipline most often require different grids to be generated from the surface model. In an environment where the ability to quickly change features of the geometry is nearly as important as the geometry itself, it is desirable: (1) to have the geometry model specified in terms of a small number of design parameters; (2) to visualize the geometry and interact with it to explore the envelope of possibilities; and (3) to quickly extract grids and grid sensitivity for automated analysis (both low-level and high-level) and optimization. As the geometry becomes detailed, it is imperative that a CAD model, with its general characteristics be developed, and any parameter-defined model should be upgraded with a conventional CAD system. Alternately, it would be desirable to incorporate a methodology like the one described here in a conventional CAD system.

Design parameters can be classified according to whether or not they are coupled. Uncoupled design parameters influence the solution independently and would be the major contributors to optimization process. These parameters could be geometric, flow-dependent, or grid-dependent. The geometric design parameters specify the primary shape of a typical aerodynamic surface. Flow-dependent parameters are usually free-stream conditions such as free-stream Mach number or angle of attack. The grid-dependent parameters, relatively new in aerodynamic optimization, affect the interior and boundary grids; therefore, influencing the solution and optimization process. Traditionally, geometric parameters are considered the most affluent in aerodynamic optimization, although, optimization with respect to other design parameters is gaining respectability. For optimization with respect to geometric design parameters, a perturbation in parameters affect the surface grid and the field grid which, in turn, affect the flow-field solution. There are two basic components in obtaining aerodynamic sensitivity. They are: (1) obtaining the sensitivity of the

governing equations with respect to the state variables, and (2) obtaining the sensitivity of the grid with respect to the design parameters. The sensitivity of the state variables with respect to the design parameters are described by a set of linear-algebraic relations. These systems of equations can be solved directly by a LU decomposition of the coefficient matrix. This direct inversion procedure becomes extremely expensive as the problem dimension increases. A hybrid approach of an efficient banded matrix solver with influence of off-diagonal elements iterated can be implemented to overcome this difficulty.

1.2 Literature Survey

1.2.1 Aerodynamic Design and Surface Modelling

Airplane design has historically been divided into three phases [1][1]: conceptual design, preliminary design, and detailed design. The conceptual design of an airplane usually begins with specifications for a proposed mission and rough sketches of the configuration. Geometry begins to evolve in the form of sets of connected points, and as the configuration approaches the end of the conceptual design phase, Computer-Aided Design (CAD) models are created. In the preliminary-design phase, high level analysis and testing of physical models are performed. Geometry for computational analysis and the construction of test models is extracted from the CAD model. In the detailed-design phase, the CAD model is the central design representation, now containing detailed information for manufacturing the airplane. According to Raymer [2] design drawing is often carried out with a computer-aided drafting system where the aircraft geometry is represented by character-lines on its surface. This constitutes only a partial definition of the aircraft's surface and the process of lofting between the character lines is required to create the complete aircraft surface. Thus

[1]The numbers in brackets indicate references.

there is a need for mathematical methods for representing or parametrizing curves and surfaces, which are flexible enough to represent a wide range of shapes in an easy and intuitive manner. It is also desirable to choose a method which uses few surface defining parameters so as not to overcomplicate the problem which would lead to an excessive use of computational time whilst at the same time to ensure sufficient flexibility in the surface in order to avoid trivial solutions [3].

One method of surface representation commonly used in computer-aided design applications is that of Bezier surfaces [4]. Here the defining parameters are the set of control points which form the characteristic polyhedron to which the surface then approximates. One advantage of this method is that the effect of changing a design parameter, i.e., the effect of moving a control point, on the surface shape is intuitively predictable. An improvement to this method is found in B-spline surfaces where each control point only influences the region of the surface close to it [5]. Both these two properties are useful from the point of view of the end-user.

By the late 1970s, the CAD/CAM industry recognized the need for a modeler that had a common internal method of representing and storing different geometric entities. At about the same time, three major groups looked at the possibility of using Non Uniform Rational B-Splines (NURBS). Boeing began developing the Tiger system in 1979. Integrating B-splines [6] with rational Bezier representations [7] quickly led to rational B-splines. SDRC (Structural Dynamics Research Corporation) pursued NURBS commercially and in 1978, the company started working on a modeler. The rapid proliferation of NURBS is due partly to their excellent properties and partly to their incorporation in such national and international standards as IGES [8], PHIGS+ [9], Product Data Exchange Specification, and International Standard Office, and Standard for the Exchange of Product Model Data.

These methods, however, were not suitable to the problems investigated by Bloor and Wilson [10], since even the simple cubic Bezier surface had sixteen

control points each of which had three degrees of freedom. Also, the Bezier formulation was based on design by changing small regions of the surface independently whereas they were concerned with a more global approach to design. Bloor and Wilson [11] introduced the method of generating free-form surfaces using solutions to a suitably choosen partial differential equation. By regarding a blend as a solution to a boundary-value problem and by choosing appropriate boundary conditions, they demonstrated that a solution to a suitably chosen elliptic PDE gave a smooth blending surface that had the required degree of continuity with the primary surfaces to which it joined. Bloor and Wilson [12] have extended their work for approximating surfaces, which are the solutions of partial differential equations, in terms of B-splines so that they can be represented in a form compatible with more established surface design techniques.

1.2.2 Grid Generation and Solution Methods

In recent times techniques for the automatic generation of computational meshes have received much attention. This is primarily due to the fact that there has been an increased effort in the development of algorithms for the solution of the flowfield equations. Historically, many of the fundamental developements in theoretical fluid dynamics have rested upon conformal mapping techniques for incompressible potential flow in which solutions on the boundaries can be obtained without resort to information in the field. Also panel methods [13], which utilize distribution of sources and siks on boundary surfaces, have played and continue to play an important role in aerodynamics. Recently, however, attention has been primarily focused on solution techniques for the Full Potential, Euler and Reynolds-Averaged Navier-Stokes equations. These equations are formulated on the basis of the continum hypothesis. With computers restricted in memory and speed it is not possible to consider all points in the continum domain and hence it is necessary to select a subset of points within a

domain at which flow quantities can be calculated. The combination of points and connections between points defines a mesh or grid on which numerical methods for the solution of the flow equations can be constructed. The assumption is then made that the information at these points is sufficient to describe the complete flowfield.

In the most widely used approach [14] the domain is divided into a structured assembly of quadrilateral cells. The structure in the mesh is apparent from the fact that each interior nodal point is surrounded by exactly the same number of mesh cells. Mesh generation, however, has proved to be a stubbornly difficult problem. Considerable effort has been devoted to this area in recent years as evidenced by the extensive literature on mesh generation. Numerical mesh generation techniques [15] have proved to be a powerful approach for creating meshes around complex shapes. Algebraic methods based on surface fitting [16], transfinite interpolation [17], and sequential mapping [18] have also been applied to treat a variety of geometric shapes in both two and three dimensions. All of these methods, however, encounter difficulties when applied to complete aircraft configurations consisting of a wing, fuselage, tail and nacelles. A promising technique to tackle complex configurations is the use of a multiblock structure or a splitting-up of the space around the configuration into a number of smaller and topologically simpler regions. Separate meshes can be generated for each block. In some cases [19], the mesh is required to blend smoothly together at block interfaces to provide a mesh that can be viewed as a single block by the flow solver. In other cases [20], the mesh is not required to connect smoothly at the interfaces and interpolation is needed to transfer flow information between separate blocks. Smith et al. [21] have generated grids around very complex configurations and very promising results have been obtained.

Neverthless, the generation of a mesh around a complete aircraft configuration, including engine nacelles, has resisted the efforts of researchers until fairly recently. The first published calculations using a structured, conforming mesh around

a wing/ fuselage/nacelle/pylon combination is the work of Vigneron et al. [22] More recently, Sawada and Takanashi [23] generated a structured mesh to calculate the flow over a complete aircraft with wing mounted nacelles. These are striking successes in the generation of structured hexahedral meshes around complex configurations.

The alternative approach is to divide the computational domain into an unstructured assembly of computational cells. The notable feature of an unstructured mesh is that the number of cells surrounding a typical interior node of the mesh is not necessarily constant. The nodes and the elements are numbered and, to get the information on the neighbours, we store the numbers of the nodes which belong to each element. There is no concept of directionality within a mesh of this type and that, therefore, solution techniques based upon this concept (e.g. ADI methods) will not be directly applicable. The methods which are normally adopted to generate unstructured triangular meshes are based upon either the Delauny [24] or the advancing front [25] approaches. Discretization methods for the equations of fluid flow which are based upon integral procedures, such as the finite volume or the finite element method, are natural candidates for use with unstructured meshes. The principal advantage of the unstructured approach is that it provides a very powerful tool for discretizing domains of complex shape [26,27]. In addition, unstructured mesh methods naturally offer the possibility of incorporating adaptivity[28]. Disadvantages which follow from adopting the unstructured grid approach are that the number of alternative solution algorithms is currently rather limited and that their computational implementation places large demands on both computer memory and CPU [29]. Further, these algorithms are rather sensitive to the quality of the grid which is being employed and so great care has to be taken in the generation process.

1.2.3 Sensitivity Analysis and Optimization

Sensitivity analysis (SA) provides a natural systematic means for both analyzing and predicting the behavior of physical approximations and computational systems or for identifying significant input parameters in a system. The literature on sensitivity analysis and optimization is quite extensive. The pioneering work on sensitivity analysis for MDO was started from Sobieski [30] to the CFD community for extending their present capabilities to include sensitivity analysis of aerodynamic forces. Yates [31] developed an analytical approach using an implicit differentiation in combination with linearized lifting-surface theory to evaluate the sensitivity coefficients. This can be used as a benchmark criteria for assessing the accuracy of approximate methods. Murthy and Kaza [32] developed a semi-analytical technique, using linear unsteady aerodynamics, to study an isolated wing-section and rotating propfan blades. Some aeroelastic analysis for transport wing has been investigated by Grossman et al. [33], where a coupled aerodynamic and structure model influences the design. Livine et al. [34] and a few other researchers focus on more complex interactions such as inclusions of active controls on the overall optimization process. A number of researchers have successfully pursued the quasianalytical approach to calculate sensitivity derivatives from nonlinear flow-analysis codes of varying degrees of complexity. For example, Elbana and Carlson [35] have computed wing-section aerodynamic sensitivity coefficients in transonic and supersonic flight regimes, and, more recently, they extended the work to 3D full potential equations using the symbolic manipulator MACSYMA to obtain the sensitivity coefficients. The procedure was applied to ONERA M6 wing platform with NACA 1406 wing sections [36]. The calculation of quasianalytical sensitivity derivatives is reported by Taylor et al. [37], Hou et al. [38], and Baysal et al. [39] for interior channel flows from a conventional upwind finite-volume solution strategy applied to the 2D Euler equations in body-oriented coordinates. These researchers have subsequently extended this work

to calculate sensitivity derivatives for 2D laminar flows from the thin-layer Navier-Stokes (TLNS) equations, including external flows over isolated airfoils [40]. Baysal and Eleshaky [41] presented an aerodynamic design strategy using direct differentiation of Euler equations. The procedure was applied to design a scramjet-afterbody configuration for an optimized axial thrust. This scheme was later extended to include domain decomposition capabilities in order to reduce the computational costs associated with complex configurations [42]. Another strategy has been developed by Korivi et al. [43] and Newman et al. [44], where the sensitivity equations are recast and solved in incremental iterative form. The incremental iterative form is very flexible and it increases the feasibility of solving the sensitivity equations for advanced 3D CFD codes. Korivi et al. [45] have demonstrated the use of this strategy to efficiently and accurately calculate quasianalytical sensitivity derivatives for a space-marching 3D Euler code with supersonic flow over a blended wing-body configuration.

Application of the quasianalytical methods requires the construction and evaluation of many derivatives, and for advanced CFD codes, the task of constructing exactly all of these required derivatives "by hand" is extremely complex. Reference [43] shows that failure to consistently differentiate the turbulence modeling terms can result in unexpectedly large errors in the sensitivity derivatives that are calculated. A promising possible solution to this problem may be found in the use of a technique known as automatic differentiation (AD). Automatic differentiation is a chain-rule-based technique for evaluating the derivatives of functions defined by computer programs with respect to their input variables and has been investigated since 1960. Progress towards a general-purpose AD tool has been made with the development of ADIFOR by a joint effort of Argonne National Laboratory and Rice University. ADIFOR differentiates programs written in Fortran 77; that is, given a Fortran procedure (or collection of procedures) that describe a "function" and an indication of which variables in parameter lists or common blocks correspond to

"independent" and "dependent" variables with respect to differentiation, ADIFOR produces Fortran 77 code that computes the derivatives of the dependent variables with respect to the independent ones. ADIFOR has recently been tested by Bischof et al. [46] and Green et al. [47] in applications to an advanced CFD flow-analysis code called TLNS3D [48]. In these studies, a high Reynolds number, turbulent, 3D transonic flow over the ONERA M6 wing was selected as the example problem.

1.3 Objectives of Present Study

After reviewing relevant literature, it is aparent that parametrization of aircraft geometry plays an important role in the design process. Despite the differences in various approaches towards aircraft design it is agreed upon to identify an early stage in the design process during which general questions considering the aircraft's configuration be studied; when, in order to meet whatever requirements exist, various alternative design solutions must be considered. In the past, when considering the question of the physical properties of new design, designers have had to rely upon their own knowledge and experience, and, further along in the design process, model testing. However, the increasing sophistication of numerical methods and the increasing power of computer hardware have meant that the properties of new design can be analysed by computer long before any physical model is created. Furthermore, whereas the main use of numerical methods has been an alternative to model testing, there is an increasing trend towards their use in the design process as a tool for optimization. Development of an efficient and reliable surface definition, grid generation, grid sensitivity and optimization for conceptual design of aerodynamic shapes appears essential.

An important ingredient of grid sensitivity and surface optimization is the surface parameterization. The most general parameterization would be to specify every grid point on the surface as a design parameter. This, although convenient,

is unacceptable due to high computational cost. It is essential to keep the number of parameters as low as possible to avoid a surge on computational expenses. An analytical parameterization, may alleviate that problem but it suffers from lack of generality. A compromise would be using spline functions such as a Bezier or Non-Uniform Rational B-Spline (NURBS) function to represent the surface [49]. In this manner, most aerodynamically inclined surfaces can be represented with few control (design) parameters. This method has its own disadvantages, like the definition of wing fuselage intersection. The method of generating blend surfaces was a key area which led to the investigation of free form surfaces. Generation of free form blend surfaces was investigated by Bloor et al.[50]. The surfaces which they generated were quite interesting and the applications ranged from telephone handset to hull of a ship. They used the solution of fourth order partial differential equation to generate blend surfaces. In this study the idea was explored and was used towards generating aerodynamic shapes.

With the advance in computers much research have been directed towards the development of graphic interface, which could accurately represent the surface on a computer screen. Most of the available graphics software have the capability of dynamic translation and rotation. It was realized after reviewing the literature that the need for a graphic interface which could help the designer view the dynamic change in surface with the change in design variable was extremly helpful. This would act as an additional tool in the initial conceptual development of surfaces.

The second main objective of this study is to do a grid sensitivity and surface optimization. Unlike aerodynamic considerations, the grid sensitivity analysis has been used on structural design models for a number of years. In this context, grid sensitivity can be thought as perturbation of structural loads, such as displacement or natural frequency, with respect to finite element grid point locations [51]. Two basic approaches have been cited for grid sensitivity derivatives. The first approach, known

as implicit differentiation, is based on implicit differentiation of discretized finite element system. The other, which is based on the variation of continuum equations, is known as variational or material derivative approach. The main objective here is to develop a fast and inexpensive method for grid sensitivity to be used on an automated aerodynamic optimization cycle. Among two major classes of grid generation systems (Algebraic, Differential), algebraic grid generation systems are ideally suited for achieving this objective. The explicit formulation, resulting in a fast and suitable grid, enables direct differentiation of grid coordinates with respect to design parameters [52,53]. The development of software packages like ADIFOR, which could compute the derivatives in a manner that could save the time and effort of analytical methods was extremely helpful. This study involves the application of this software to compute both the grid and flow sensitivity towards an optimization study.

The organization of this study is as follows. The physical and geometric representations of a typical model are derived in Chap. 2. Chapter 3 discusses the graphical user interface. The grid generation algorithm for both structured and unstructured is described in Chap. 4. The method of solution is provided in Chap. 5. Chapter 6 discusses the theoretical formulation and aerodynamic sensitivity equation. The results are presented and discussed in Chap. 7. Finally, some concluding remarks are provided in Chap. 8.

Chapter 2

PHYSICAL MODEL

2.1 Computer-Aided Geometric Design (CAGD)

In the late 1950s hardware became available that allowed the machining of 3D shapes out of blocks of wood or steel [54]. These shapes could then be used as stamps and dies for products such as the hood of a car. The bottleneck in this production method was soon found to be the lack of adequate software. In order to machine a shape using a computer, it became necessary to produce a computer-compatible description of that surface. The most promising description method was soon identified to be in terms of parametric surfaces. The theory of parametric surfaces was well understood in differential geometry. Their potential for the representation of surfaces in a Computer-Aided Design (CAD) environment were not known. The exploration of the use of parametric curves and surfaces to represent objects in computational environment [55] can be viewed as the origin of Computer Aided Geometric Design (CAGD).

Surfaces can be defined by implicit algebric equations or explicit parametric-algeabric equation [56]. Parametric equations have dominated CAGD because of their intrinsic simplicity for modelling complex objects.

In the development of parametric curves and surfaces, two different approaches have evolved [57]. They are referred to here as "interpolative" and "approximative". In an interpolative representation, points and derivatives on the curve or surface are used to control the formula defining the curve or surface. Langrangian and

Hermite interpolation formulas are examples of this approach. In an approximative approach points not necessarily on the curve or surface control the formula defining the curve or surface. The Bezier and B-Spline representations are examples of this approach.

In the design process using an interactive CAD system, the approximative approach is highly advantageous. After prescribing an initial set of control points, the designer can pick and drag points and simultaneously observe the change in the shape of the surface.

2.1.1 Geometric Representation of Wing Section

The most commonly used approximative representation is the *Non-Uniform Rational B-Spline* (NURBS) function. The NURBS provide a powerful geometric tool for representing both analytic shapes (conics, quadrics, surfaces of revolution, etc.) and free-form surfaces [58]. The relation for a NURBS curve is

$$\mathbf{X}(r) = \frac{\sum_{i=0}^{n} N_{i,p}(r)\omega_i \mathbf{D}_i}{\sum_{i=0}^{n} N_{i,p}(r)\omega_i} \qquad \mathbf{X}(r) = \begin{Bmatrix} x(r) \\ y(r) \end{Bmatrix} \qquad \mathbf{D}_i = \begin{Bmatrix} X_i \\ Y_i \end{Bmatrix} \tag{2.1}$$

$$i = 0, ..., n$$

where $\mathbf{X}(r)$ is the vector valued surface coordinate in the r-direction, $\mathbf{D}_i$ are the control points (forming a control polygon), ω_i are weights, and $N_{i,p}(r)$ are the p-th degree B-Spline basis function defined recursively as

$$N_{i,0}(r) = \begin{Bmatrix} 1 & r_i \leq r \leq r_{i+1} \\ 0 & otherwise \end{Bmatrix}$$

$$N_{i,p}(r) = \frac{r - r_i}{r_{i+p} - r_i} N_{i,p-1}(r) + \frac{r_{i+p+1} - r}{r_{i+p+1} - r_{i+1}} N_{i+1,p-1}(r). \tag{2.2}$$

The r_i are the so-called knots forming a uniform knot vector

$$\mathbf{r} = \left\{ \underbrace{a \cdots a}_{p+1}, r_{p+1}, \ldots, r_{m-p-1}, \underbrace{b \cdots b}_{p+1} \right\} \tag{2.3}$$

where the end knots a and b are repeated with multiplicity $p+1$. The degree, p, number of knots, $m+1$, and number of control points, $n+1$, are related by

$$m = n + p + 1. \tag{2.4}$$

For most practical applications the knot vector is normalized and the basis function is defined on the interval $(a = 0, b = 1)$. Equation (2.1) can be rewritten as

$$\mathbf{X}(r) = \sum_{i=0}^{n} R_{i,p}(r)\mathbf{D}_i \qquad R_{i,p}(r) = \frac{N_{i,p}(r)\omega_i}{\sum_{i=0}^{n} N_{i,p}(r)\omega_i} \qquad i = 0, \ldots, n \tag{2.5}$$

where $R_{i,p}(r)$ are the *Rational Basis Functions*, satisfying the the following properties among many others found in [59]

$$\sum_{i=0}^{n} R_{i,p}(r) = 1 \qquad R_{i,p}(r) \geq 0. \tag{2.6}$$

Figure 2.1 shows a six control point definition of the cambered airfoil obtained by Eq. (2.5). The points at the leading and trailing edge are fixed. Two control points at 0% chord are used to affect the bluntness of the section. The effect of the movement of the control points to create another airfoil is shown in Fig. 2.2. Figure 2.3 shows the effect of increasing the weight of the middle control point. It is seen that the curve is pulled towards the control point. An arc length distribution of the unit line is used for the knot vector

An interactive program based on Eqs. (2.1-2.5) have been developed. The program is menu driven, where after prescribing an initial set of control points, the designer can pick and drag these points and simultaneously observe the change in shape of the curve. Figure 2.4 shows the snap shot view of the interactive program. The cursor is drawn as a cross hair and different options are available in the pull down

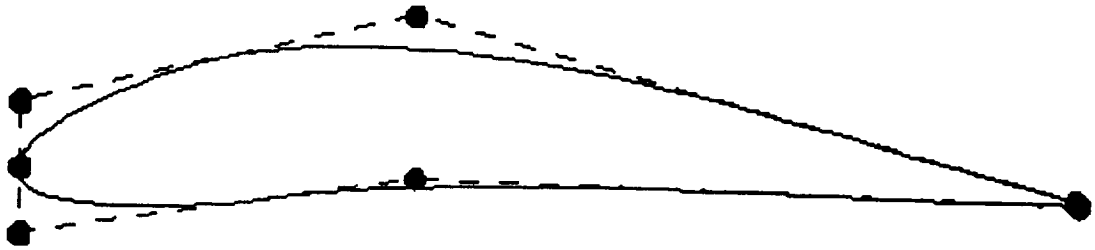


Fig. 2.1 Six control point wing section definition.

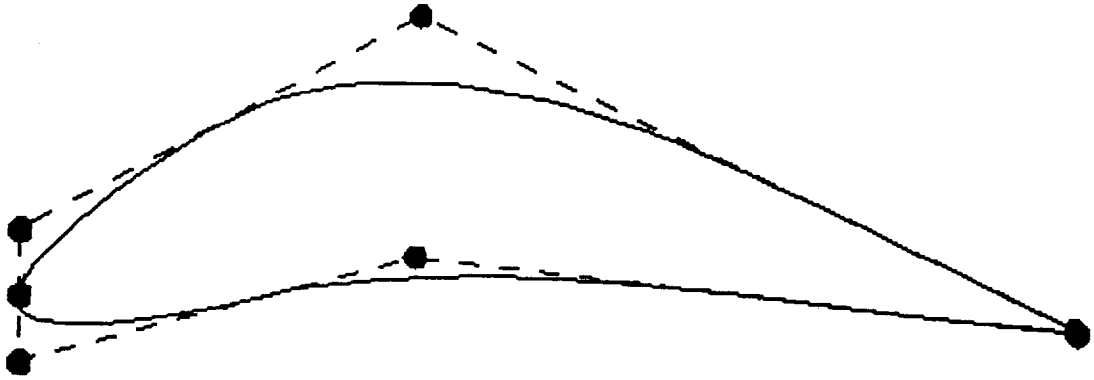


Fig. 2.2 Effect of moving the control points.

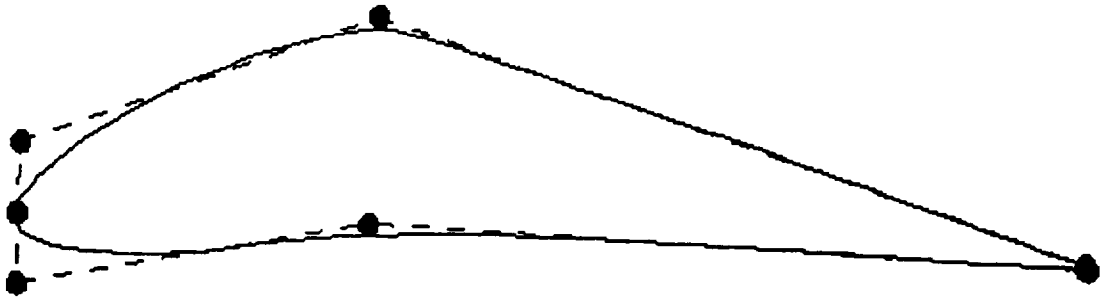


Fig. 2.3 Effect of increasing the weight of control point.

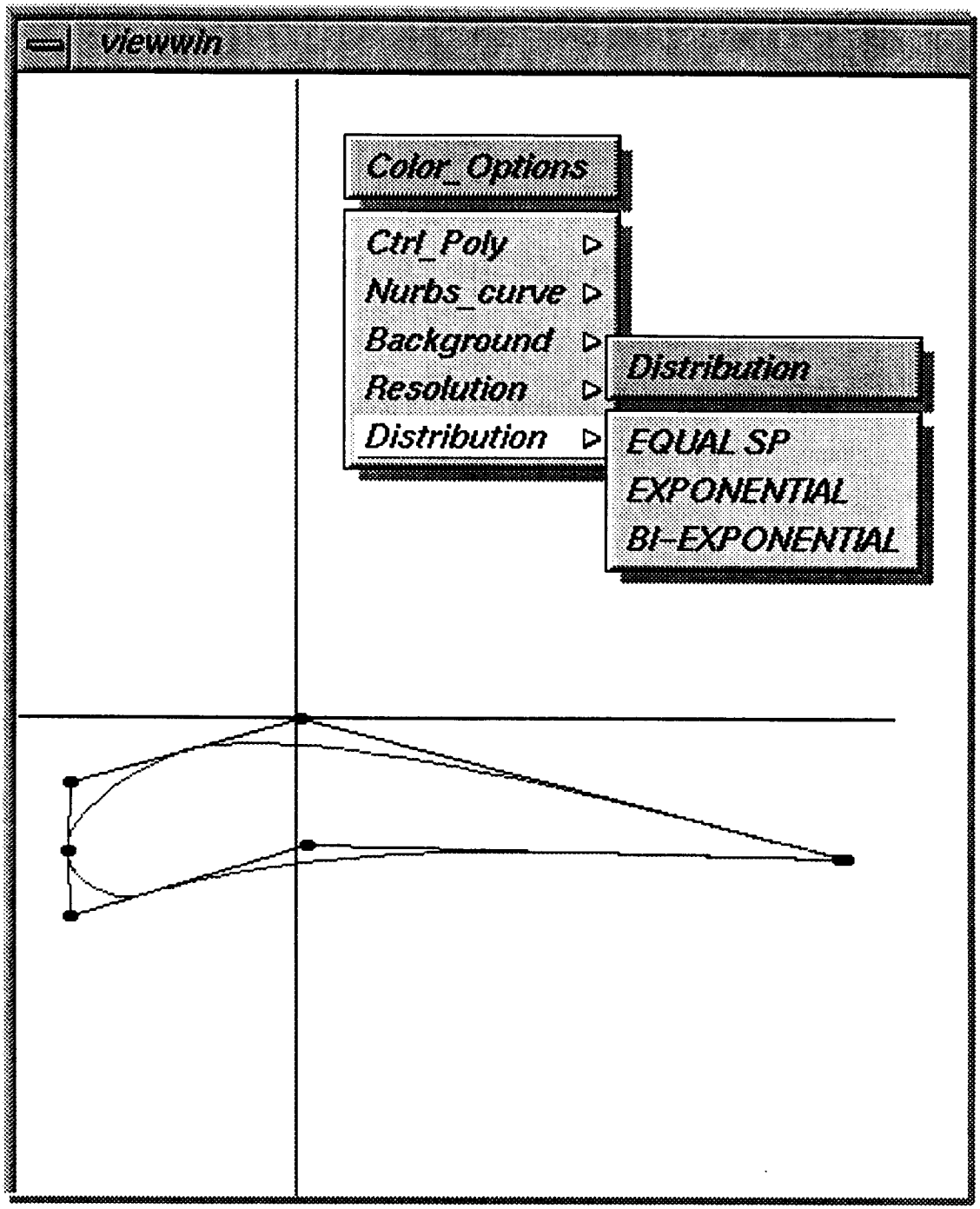


Fig. 2.4 Snapshot of graphic interface of NURBS curve.

menu. Weights associated with each of the control point can also be changed. The distribution of points for the NURBS curve is set to arc length formulation which can be also changed by th user. The program has the capability to output the NURBS curve into a predefined file.

2.2 Surface Design and Parametrization

The description of parametric surfaces in Computer-aided design can be broadly classified into the categories of shape representation and shape design [60]. Shape design is typically accomplished in an interactive manner i.e., the designer starts with a sketch and refines it untill it meets the requirements. Characteristically to the representation approach, there already exists a prototype of the model and numeric information that describes it, for which the corresponding computer model is processed automatically within the appropriate tolerance. Shape modifications have been of interest in both CAD/CAM and graphics for atleast two decades. In CAD/CAM, shape serves such purposes as aesthetics (free form design) , smoothing (removing wiggles, bumps etc.), satisfying special design requirements such as generating hard points or hard lines and adjustment of geometry (eg. spline based variational geometry). In graphics, shape can be used to generate a large variety of shapes or to perform animation based on subtle modifications. In any case, shapes generated by a computer system are rarely immediately acceptable, and subsequent modifications are required. The available techniques for modifications depend on the underlying representation scheme. Using splines, the modifications can be accomplished by, moving control points (Refinement can be made by either degree elevation or knot insertion), using special blending functions such as tensioned splines, and using rational polynomials with weight.

For an aerospace vehicle such as High-Speed Civil Transport (HSCT), the traditional approach to design is for aerodynamics and performance disciplines to initially create the vehicle surface[61]. The process is to define the planform, wing, fuselage, engine nacelles, and major control surfaces with aero/performance independent-design parameters. For instance, the wing is specified by the planform description, wing section, dihedral angles, and twist angles. Several sections are required for a wing. Approximately 50-100 independent parameters are required to specify a rough vehicle surface [62]. Usually a sparse set of points on the component surfaces which can be thought of as a very coarse grid becomes the surface description for analyses. A refined definition of a vehicle surface is obtained by applying Computer-Aided Design (CAD) techniques to sparse definition. The input to the CAD system is the sparse definition. CAD is used to create a patch definition of each vehicle components, and add surfaces such as fillets and wingtips.

A patch is represented mathematically as

$$\mathbf{X}(u,v) = \sum_{i=0}^{m}\sum_{k=0}^{n} h_{i,k} H_i^m(u) H_k^n(v) \tag{2.7}$$

$$0 \leq u, v \leq 1.$$

where u and v are parametric coordinates, h is a matrix of surface definition parameter and $H_i^m(u)$, and $H_k^n(v)$ are interpolation functions respectively in the u and v directions.

For the case of bicubic surfaces (m and n=3) the matrix of defining parameters is

$$\left[h_{i,k}\right] = \begin{bmatrix} x(0,0) & x_v(0,0) & x_v(0,1) & x(0,1) \\ x_u(0,0) & x_uv(0,0) & x_uv(0,1) & x_u(0,1) \\ x_u(1,0) & x_uv(1,0) & x_uv(1,1) & x_uv(1,1) \\ x(1,0) & x_v(1,0) & x_v(1,1) & x(1,1) \end{bmatrix} \tag{2.8}$$

The elements of $h_{i,k}$ are the corner points of the patch derivatives with respect to

the parameter variables at the corner points and cross derivatives with respect to the parametric variables at the corner points. For a bicubic patch there are 48 defining parameters, and a refined vehicle surface may consists of several hundred patches.

In this study two methods of representing the vehicle surface are considered. The first is the most general approximative representation i.e., Non-Uniform Rational B-Spline [63] (NURBS) and the second is a novel parametrization procedure which uses the solution to a suitably choosen fourth order Partial Differential Equation [64] (PDE) to represent the surface.

The commercial environment in which the two parametrization procedures was investigated requires that it should satisfy the following:

(1) provide flexibility to design geometry

(2) give a set of tools the designer can invoke at any stage of the design process

(3) work in a reliable, fast and accurate manner

(4) operate such that any modifications should preserve the entire continuity of the geometry, and

(5) provide analytical equation defining surface to perform design optimization.

2.2.1 M-6 Wing NURBS Representation

A NURBS surface [65] is the rational generalization of the tensor product nonrational B-Spline surface and is defined as

$$\mathbf{S}(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)\omega_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)\omega_{i,j}} \tag{2.9}$$

where $\omega_{i,j}$ are the weights, $P_{i,j}$ form a control net, and $N_{i,p}(u)$ and $N_{j,q}(v)$ are the normalized B-Splines of degree p and q in the u and v directions. The knot vectors are

$$\mathbf{U} = \left\{ \underbrace{0,\cdots,0}_{p+1}, u_{p+1}, ..., u_{r-p-1}, \underbrace{1,\cdots,1}_{p+1} \right\} \tag{2.10}$$

$$\mathbf{V} = \left\{ \underbrace{0, \cdots, 0}_{q+1}, v_{q+1}, ..., v_{s-q-1}, \underbrace{1, \cdots, 1}_{q+1} \right\} \tag{2.11}$$

where r = n + p + 1 and s = m + q + 1.

Introducing the piecewise rational basis functions:

$$\mathbf{R}_{i,j}(u,v) = \frac{N_{i,p}(u) N_{j,q}(v) \omega_{i,j}}{\sum_{k=0}^{n} \sum_{l=0}^{m} N_{k,p}(u) N_{l,q}(v) \omega_{k,l}} \tag{2.12}$$

the surface Eq. (2.9) can be written as

$$\mathbf{S}(u,v) = \sum_{k=0}^{n} \sum_{j=0}^{m} R_{i,j}(u,v) \mathbf{P}_{i,j} \tag{2.13}$$

A NURBS surface has the property $\sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u) N_{j,q}(v) = 1$ and reverts to a B-spline when all the weights are 1. A NURBS surface has the advantage of being able to represent free-form surfaces, and with the proper choice of weights, conic surfaces.

The surface skinning technique [66] is used to obtain the NURBS surface. The task of skinning is to fit a surface through an ordered set of space curves, called as section curves. The positioning of section curves in the three-dimensional space is customarily done with respect to a spine curve, from which appropriate orientation vectors can be automatically computed. The skinned surface is obtained in three steps:

1. All the cross-sectional curves are first made compatible. That is, all the curves should have the same degree and number of control points and be defined over the same knot vector.

2. Next u values and a knot vector V is calculated for interpolation with degree-q NURBS curves.

3. Using the above values, curves are interpolated through the control points calculated by Eq. (2.13).

ONERA M6 wing is used to demonstrate the skinning technique. The points

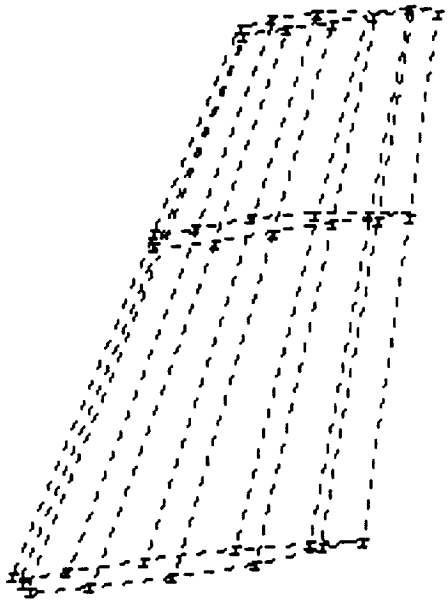


Fig. 2.5 Control point polygon for ONERA M6 wing.

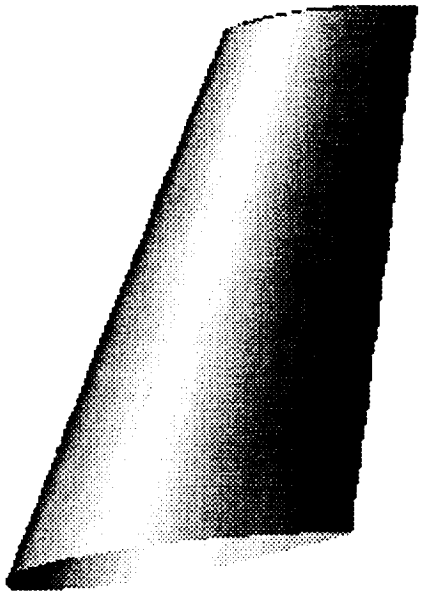


Fig. 2.6 Shaded NURBS suface for ONERA M6 wing.

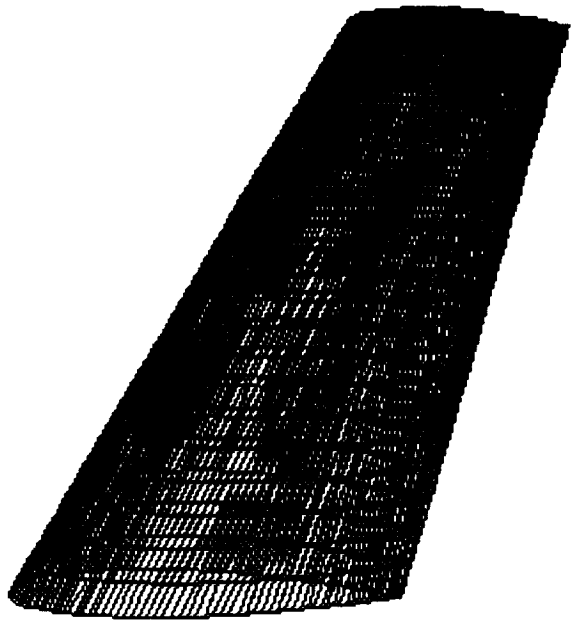


Fig. 2.7 Coarse surface grid over NURBS surface.

generated from the CAD software is used in the skinning process and the control points generated are shown in Fig. 2.5. The shaded NURBS surface is shown in Fig. 2.6. A coarse CFD grid is generated on the surface of the wing and is shown in Fig. 2.7.

2.2.2 PDE Method

The PDE method generates a surface X in Euclidean 3-space, which is a function of two parameters, i.e., X = (x(u,v), y(u,v), z(u,v)). The surface is obtained by solving a partial differential equation (PDE), in parameter u,v space, subject to boundary condition on X and its normal derivative with respect to u and v. In general, the order of PDE determines the number of derivatives of the unknown function that must be specified in the boundary condition. If control over both shapes of the curves bounding the PDE surface patch and the directions and magnitude of the coordinate vectors X_u and X_v at the edge of the patch are required then atleast a fourth order PDE is needed to generate the surface. The PDE may be written as

$$\left[\frac{\partial^2}{\partial u^2} + a^2\frac{\partial^2}{\partial v^2}\right]^2 X = 0 \tag{2.14}$$

where X = (x(u,v), y(u,v), z(u,v)).

The appropriate boundary conditions for Eq. (2.14) are the value of X and its normal derivative around the edges of the domain in the (u,v) plane. Since the generating equation, Eq. (2.14), is an elliptic PDE, the solution becomes very sensitive to the choice of boundary conditions. The boundary conditions act as a powerful tool for surface manipulation by a designer and can be used as a design parameter in an optimization process. The boundary conditions on function X are choosen that the curves forming the edges of the surface patch have the desired shape. The direction of the vector X_u and X_v are tangential to the isoparametric lines on the surface. Therefore by altering the values specified for X_u and X_v along the boundaries, one can effect the direction in which the surface moves away from the edges of the patch.

The general solution of Eq. (2.14) can be written in the form

$$X = A_0(u) + \sum_{n=1}^{0}(A_n(u)cos(nv) + B_n(u)sin(anv)) \tag{2.15}$$

where the coefficient function $A_n(u)$ and $B_n(v)$ are of the form

$$A_n(u) = a_{n1}e^{anu} + a_{n2}ue^{anu} + a_{n3}e^{-anu} + a_{n4}ue^{-anu}$$

$$B_n(v) = b_{n1}e^{nv} + b_{n2}ve^{nv} + b_{n3}e^{-nv} + b_{n4}ve^{-nv}$$

The quantities a_{n1}, a_{n2}, a_{n3}, a_{n4}, b_{n1}, b_{n2}, b_{n3}, and b_{n4} are vector valued constants that can be found for a particular solution by Fourier analysis of the condition imposed on the isoparametric lines bounding the patch.

Consider now, the problem of creating simple blends between two circular cross sections. For an illustrative purpose, consider the blend between a cylinder and a plane. It is necessary to set up the problem as a boundary value problem in (u,v) space with boundary conditions specified along curves in the (u,v) plane that corresponds to closed curves in E^3. One of the boundary curve is taken to be the plan outline of the circular cylinder. Another boundary curve which is the definition of the plane is taken to be u = 1 and again is given parametrically in terms of v. Knowing that seperable solutions to Eq. (2.15) are of the form sinusoidal function multiplied by exponential function, the choice of boundary condition must reflect this. Thus, for this example, the flat plane is considered to be at z = 0, and the curve is defined by

$$x_p = r_p cos(v)$$

$$y_p = r_p sin(v)$$

$$z_p = 0 \tag{2.16}$$

This is the boundary condition on X that is applied at u = 0. Similarly, at u = 1, the curve for the cylinder is defined by

$$x_c = r_c cos(v)$$

$$y_c = r_c sin(v)$$

$$z_p = h \tag{2.17}$$

Since the generating equation is a biharmonic like partial differential equation, it requires derivative boundary conditions which are given at the plane by

$$x_p^{'} = s_2 cos(v)$$

$$y_p^{'} = s_2 sin(v)$$

$$z_p^{'} = 0 \tag{2.18}$$

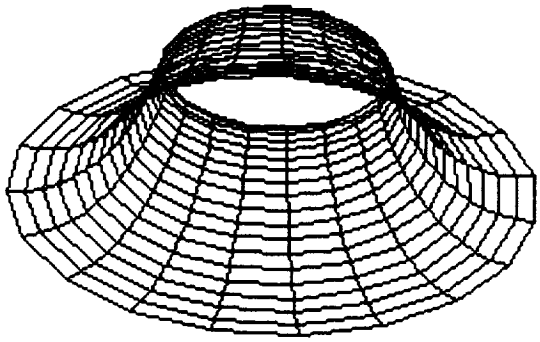
and at the cylinder by

$$x_c^{'} = 0$$

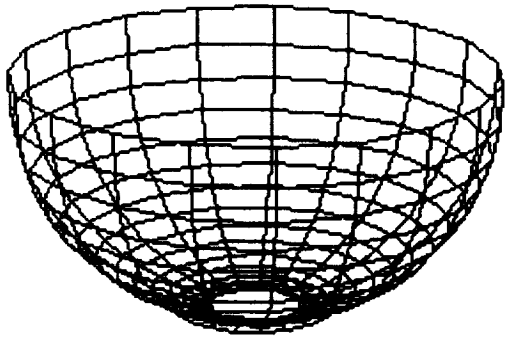
$$y_c^{'} = 0$$

$$z_p^{'} = s_1 \tag{2.19}$$

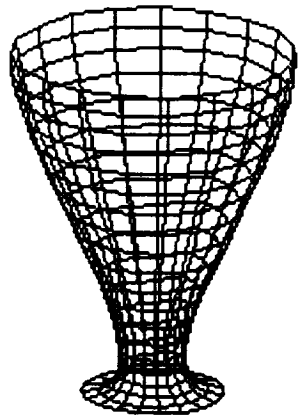
Figure (2.8) shows the blend between the circular cylinder and the plane. The different constants which act as design parameters are the radius of the cylinder, the height of the cylinder and the slope of the cylinder. For the plane it is the radius of the circular plane and its slope. The effect of varying the slopes of the cylinder and plane on the blend is shown in Figs. 2.8(b-d). In Fig. 2.8(b), the radius of the cylinder is very large compared to the plane and a very large slope is choosen for the cylinder. It is seen that the grid lines near the cylinder is orthogonal. In Fig. 2.8(c), the radius of the cylinder is reduced and also the slope of the plane. In Fig. 2.8(d) a negative value of the slope is choosen for the plane.



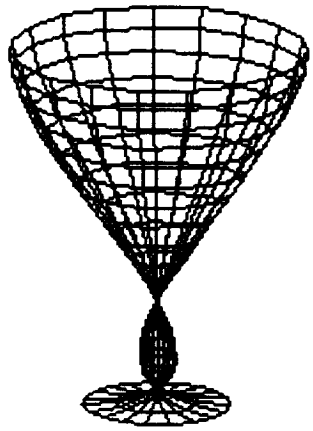
a) Regula Blend.



b) Large radius and high positive value of slope for cylinder.



c) Small value of slope for the plane.



d) Negative value of slope for the plane.

Fig. 2.6 Blend between a circular cylinder and a plane with change in design parameters.

It is concluded from these figures that the slope plays an important role in determining the blend between two cross-sectional curves.

2.3 Generation of Complete Aircraft by PDE Method

Consider an aircraft shape made up of five patches: a fuselage, an inner wing, an outter wing and vertical and horizontal tails. For simplicity the fuselage is defined algebraically. The characteristic lines which form the boundaries between adjacent surface patches are

(1) the curve where the inner wing meets the fuselage

(2) the curve where the inner and outer wing meet

(3) the curve at the tip of the outer wing and

(4) the curves for the horizontal and vertical tails

Figure 2.9 shows the different patches and sections used to represent the HSCT type configuration.

A methodology based on the above mentioned theory has been developed to define a class of airplane configurations. It directly evaluates the surface grid, volume grid, and grid sensitivity, and the main objective of the methodology is to provide a grid generation package for conceptual design that could be used in a wide spectrum of analyses (potential flow to Navier-Stokes). The methodology and associated software is developed by Smith et al. [67] and is called Rapid Airplane Parametric Input Design (RAPID).

The fuselage definition is an algebraic function which creates two surfaces - one above the fuselage intersection and one below. The airplane is considered to be symmetric about the x-z plane at $y = 0$, and only one side of the airplane is computed. The fuselage cross section is circular and is generated as a Fourier series whose axis is parallel to the x-axis, where the y and z coordinates of points on the surface are related by

$$x = R_F\xi, y = r(\xi)cos(\pi\zeta/2), z = r(\xi)sin(\pi\zeta/2)$$
$$y^2 + z^2 = r^2 \tag{2.20}$$

with

$$r(\xi) = a_0 sin(\theta) + a_1 sin(3\theta) \tag{2.21}$$

where

$$\theta = \pi((1. - a_2)\xi + a_2)$$

In the preceeding equation a_0, and a_1 are constants and θ is a parameter which lie in the range $0 \leq \theta \leq 180$. The value of $\xi = 0$ corresponds to the end point on the fuselage, and $\zeta = 0$ corresponds to a point along the curve seperating the upper and lower fuselage surfaces. The parameters for the fuselage are: R_F, the fuselage length; $a_0 and a_1$, control for the fuselage radius; and a_2, a parameter to control a finite radius at the end of the fuselage. The boundary curve separating the upper and lower fuselage surfaces is a combination of the fuselage intersection with the lifting components and cubic curves connecting the intersections. The fuselage center is optionally allowed to translate upward along a quadratic function from the trailing wing/fuselage intersection point to the end of the fuselage. This creates a "duck tail" characteristic in the fuselage which can simulate take off and landing. Figure 2.10 shows the fuselage cross section with different constants.

2.3.1 Dirichlet Boundary Condition for the PDE Solution

The curve where the outer wing and inner wing meet is taken as a plane curve (z=constant) having the shape of a simple airfoil. The airfoil shape at the crank is given by the relation

$$x = Csin(\pi v) + X_t$$

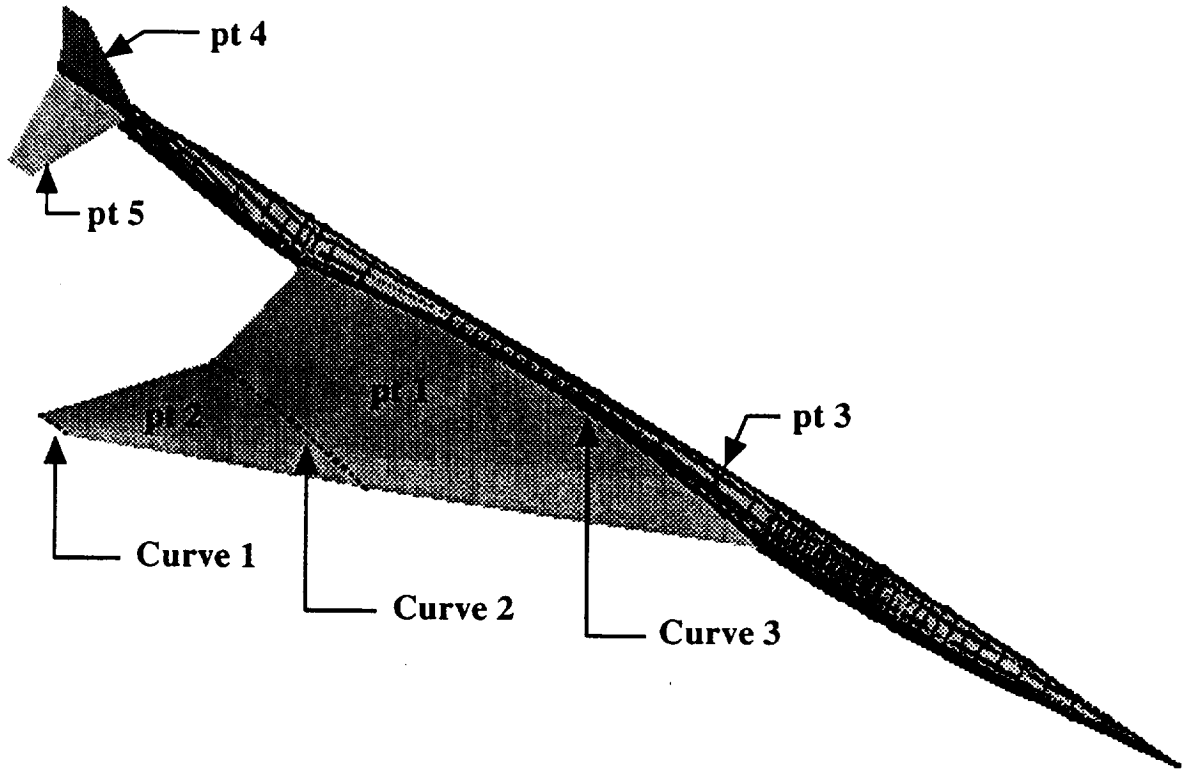


Fig. 2.9 Different patches and curves for the HSCT type representation.

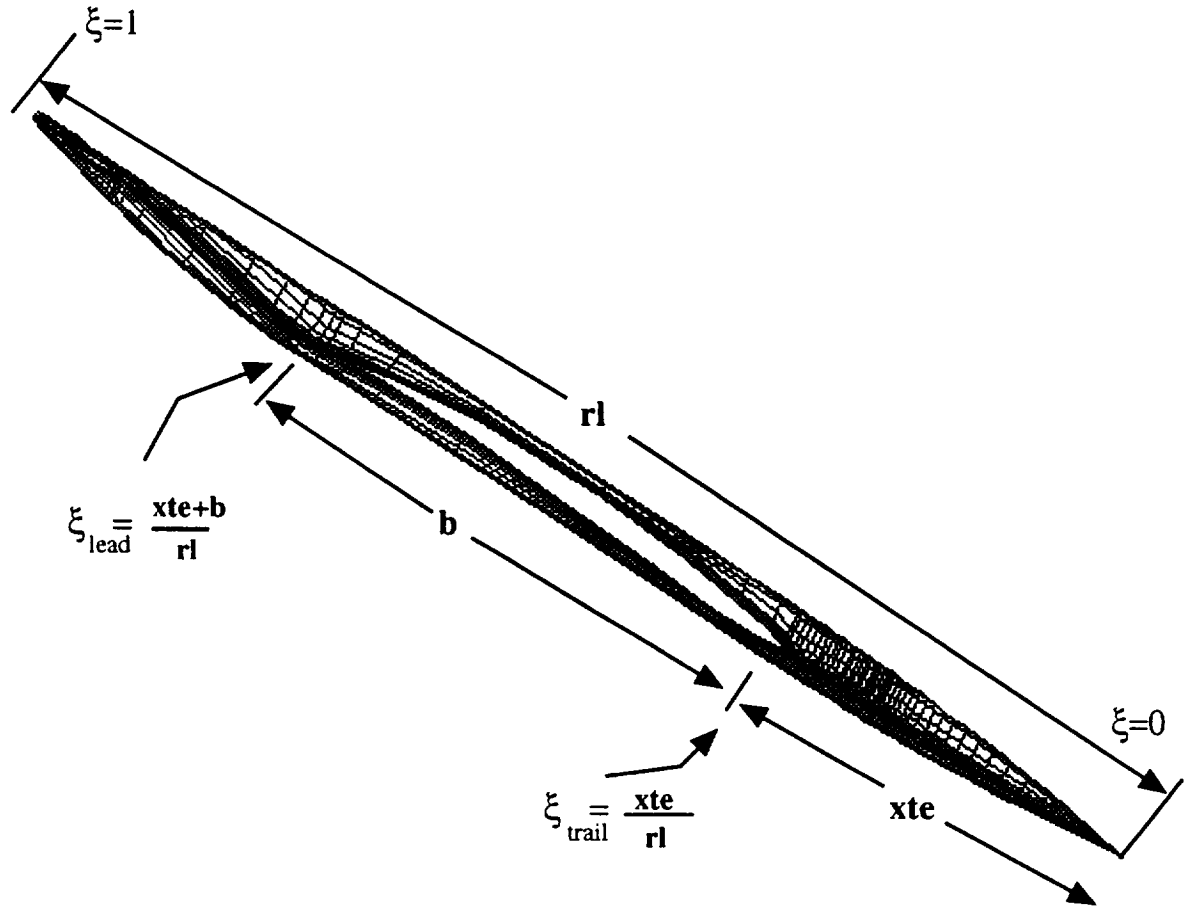


Fig. 2.10 Fuselage representation of HSCT type configuration.

$$y = y_{cam} + Y_t$$

$$z = a_0 + H_1 \tag{2.22}$$

where X_t, Y_t translates the crank boundary in a xy plane, also

$$y_{cam} = \frac{M}{L^2}(2Lx - x^2) \qquad x \leq L$$

$$y_{cam} = M\frac{(1 - 2L + 2Lx - x^2)}{(1 - L)^2} \qquad x \leq L$$

$$y_t = -\frac{T}{2}(sin(2\pi v) + P_1 sin(4\pi v) + P_2 sin(6\pi v))$$

Parameters C, T, L and M are chord, thickness, location of maximum camber and maximum camber respectively, $P_1 and P_2$ are Fourier constants. The definition of the section starts at the trailing point, proceeds beneath the camber curve, around the leading point and over the camber curve back to the trailing edge. Figure 2.11 shows the airfoil definition at the intersection of the outer and the inner wings.

The second character line lies on the surface of the fuselage. It is given parametrically by the equations

$$x_f = \frac{B.x}{C} - X_d$$

$$y_f = y.T_{ap} + Y_d$$

$$z_f = a(X)^2 - (y.T_{ap} + Y_d)^2 \tag{2.23}$$

where B is the wing-root chord length, X_d, Y_d translates the wing fuselage intersection and T_{ap} scales the thickness at the wing fuselage intersection relative to the thickness at the crank. This character line is basically a curve on the fuselage, whose projection onto the vertical plane containing the fuselage axis is an airfoil shape similar to the airfoil definition given by Eq. (2.23), but scaled by a factor (B/C).

The third character line lies at the tip of the outer wing. It is given parametrically by the equations

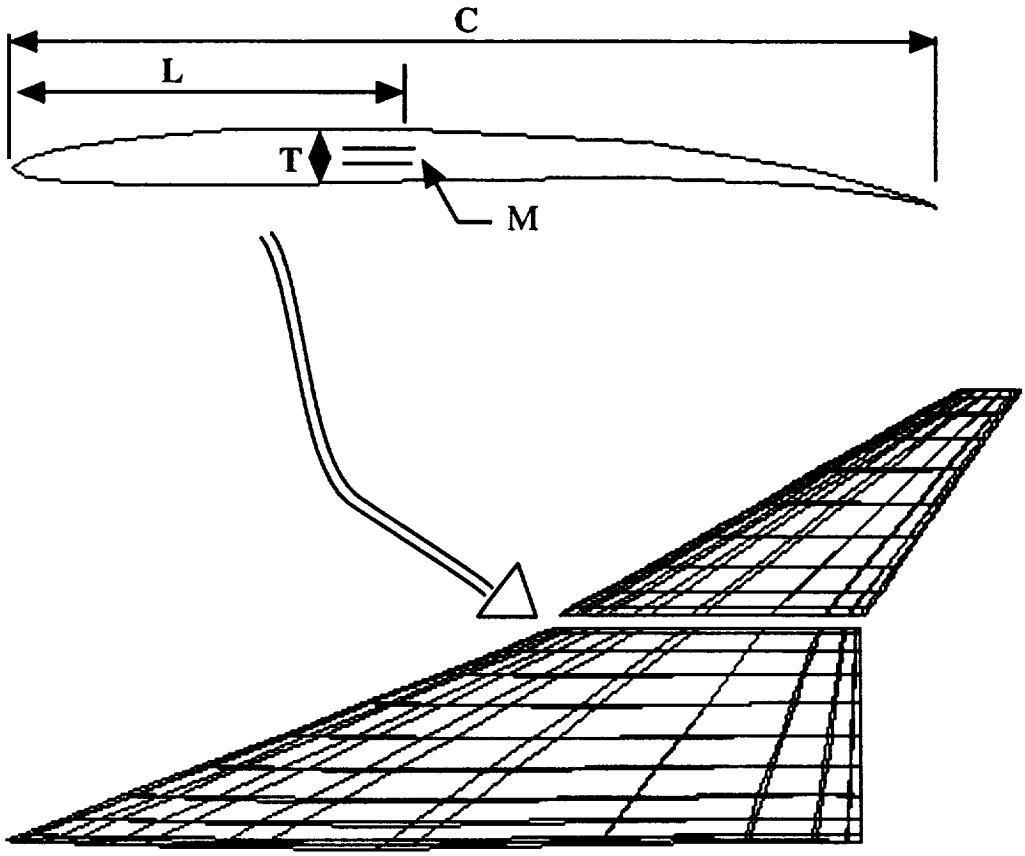


Fig. 2.11 Airfoil section definition at the intersection between outer and inner wings.

$$x_{ti} = X_c + \frac{X_{tl}.x}{C}$$

$$y_{ti} = \frac{X_{tl}.y}{C} + Y_c$$

$$z_{ti} = a_0 + H_1 + H_2 \tag{2.24}$$

where X_{tl} is the chord length at the wing tip, X_c, Y_c translates the wing tip in the xy plane, and $H_1, and H_2$ are the span length of the inboard and outboard wings respectively.

The outer wing is generated by solving Eq. (2.14) using the boundary conditions obtained from the character lines, Eq. (2.22-2.24). Similarly the inner wing is generated with character lines given by Eq. (2.22) and solving Eq. (2.23).

2.3.2 Neumann Boundary Conditions for PDE Solution

Since the governing PDE equation, Eq. (2.14), is a fourth-order equation, it requires boundary condition on the normal derivatives of X (u,v) in the (u,v) parameter plane, which in the present case means boundary conditions on X_u. The criterion used to decide how the boundary values of the tangent vector X_u is chosen is based on the fact that, if tangent continuity between the blend and the primary surface is required, then the direction of surface normal must be continuous across the blend trimline. Note that the magnitude of this vector determines the 'speed' with which the isoparametric lines move away from the boundaries of the blend.

On character-line (1), which lies at the junction of the outer and inner wing, the derivative boundary conditions are as follows:

$$\frac{x_u = S_1.x}{2}$$

$$y_u = 0$$

$$z_u = -S_1 \tag{2.25}$$

where S_1 is an adjustable design variable.

On character line (2) given by Eq. (2.25), which lies on the fuselage, the derivative boundary conditions are as follows:

$$x_f(u) = S_2.\frac{\partial y}{\partial v}.sin\pi v$$

$$y_f(u) = -S_2.\frac{\partial x}{\partial v}.sin\pi v$$

$$z_f(u) = \frac{S_2.\frac{\partial y}{\partial v}sin(va)\frac{\partial a}{\partial x} - (Y_d + y.T_{ap})(-S_2\frac{\partial x}{\partial v}sinv)}{a(X)^2 - (S_2\frac{\partial y}{\partial v}sinv)^2} \tag{2.26}$$

On the character line (3) which lies at the wing tip, the derivative boundary conditions are as follows:

$$x_{ti}(u) = S_3(\frac{x.X_{tl}}{C})$$

$$y_{ti}(u) = 0$$

$$z_{ti}(u) = 0 \tag{2.27}$$

The quantities S_1, S_2, and S_3 are adjustable design parameters whose values may be changed to control the transition of surface between inboard and outboard wing components and from wing into fuselage respectively. Figure 2.12 shows the complete PDE surface of the HSCT type configuration. Horizontal and vertical tails are added in a similar fashion as the inboard wing. All the major surface defining parameters are shown in the Fig. 2.13.

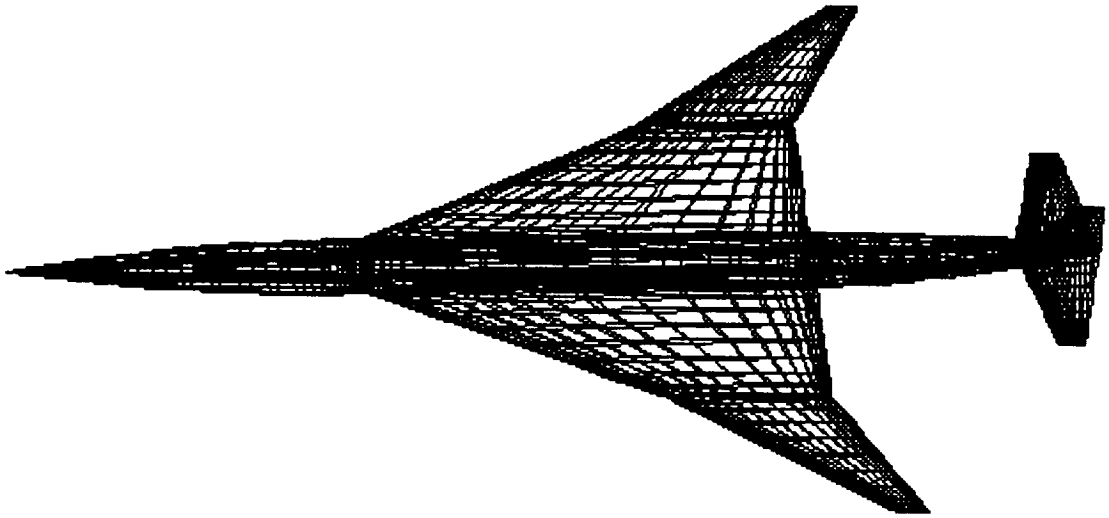


Fig. 2.12 PDE surface mesh for HSCT type configuration.

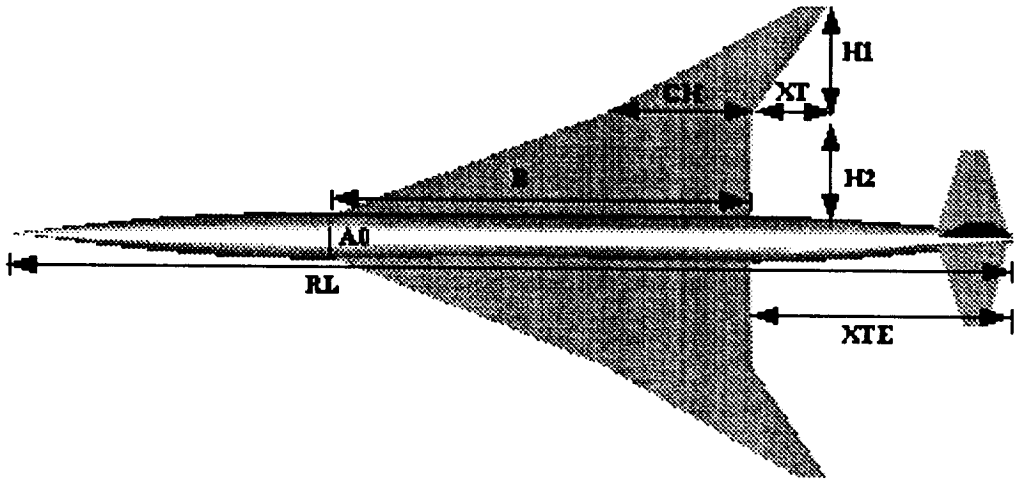


Fig. 2.13 Surface parametrization of HSCT type configuration.

Chapter 3

GRAPHIC INTERFACE

3.1 Introduction

Interactive computer graphics is based on the concept of working with a model described by information stored in the computer. For a simple application such as drafting, the model includes only the information required to generate a picture of the physical object, such as the lines in a drawing or a detailed three dimensional representation. The application areas of simulation or computer-aided design and analysis involve a more extensive model in which the graphical data are associated with additional facts or mathematical equations explaining nonvisual characteristics of the physical object [68]. An abstract entity such as a chemical process can be modeled by a graphic flow chart. The work with the computer model is of two types, the creation of the model through input by the user and the display of the resulting model by the computer. The subject of interactive input encompasses more than just the way information is transferred from the input equipment to the graphics application program. In fact, the lowest level of input functions handle this transfer. It is the higher level functions that produce a satisfactory man-machine dialogue. During the design process the operator frequently needs to delete a previously drawn object, move an object, or modify an object in some way. The input commands that the operator can use to accomplish these actions all involve interacting with the drawing that is already stored in the data base. This is a more complicated process than simply adding new data because it is first necessary to indicate to the computer which

Constructing user interfaces for programs is normally a time consuming process. They are though very important to help the user work with the program in an easy and pleasant way. In the past few years a large number of packages have appeared that help build up graphical user interfaces (so-called GUI's) in a simple way. Most of them though are difficult to use and/or expensive to buy and/or limited in their possibilities.

3.2 Interface Using Forms Library

The Forms Library [69] package which was developed at the department of computer science, Utrecht University, Netherlands is a package that is simple to use, powerful, graphically good looking and easily extendable.

The main notion in the Forms Library is that of a form. A form is a window (normally without a border) on which different objects are placed. Such a form is displayed and the user can interact with the different objects on the form to indicate their wishes. Many different classes of objects exist like for example, buttons that the user can push with the mouse, sliders with which the user can indicate a particular setting, input fields in which the user can scroll through large amount of text, etc. Whenever the user changes the state of a particular object on one of the forms displayed the application program is notified and can take action accordingly.

The forms library consists of a large number of C-routines and is simple to use. Defining a form takes a few lines of code and interaction is fully handled by the library routines. First one or more forms are defined, by indicating what object should be placed on them and where. After the form has been defined it is displayed on the screen and control is given to a library call *fl-do-forms()*. This routine takes care of the interaction between the user and the form and returns as soon as some

change occurs in the status of the form due to some user action. In this case control is returned to the program (indicating the object changed) and the program can take action accordingly, after which control is returned to the *fl-do-forms()* routine. Multiple forms can be handled simultaneously by the system and can be combined with windows of the application program.

An interface based on the forms library called Parametric Representation of Input Surface Mechanism (PRISM) is developed. The main application program where the points are generated to represent a surface is obtained from the program RAPID [67]. The application program RAPID is converted to C language and is combined with PRISM. The design variables which act as boundary condition for the solution of PDE equation are represented as different buttons on the screen. Buttons are provided for rotation and translation of the object and also to read and write a particular surface. The user can activate the program by simply typing PRISM. The program generates the surface points based on the application program which in this case is the solution of PDE equation. Each of the different sections is considered as a different surface and hence represented by a different color. As an example, a HSCT type configuration is considered and is represented by five different surfaces (fuselage, inner wing, outer wing, horizontal tail, and vertical tail). The user has the freedom to change the color of the different surfaces by selecting each of the surfaces individually. The program PRISM initially represents the surface in wireframe format which can be changed or rendered as shaded. Once the surface is represented on the viewing or main window, the user can pick any of the buttons of the different constants(design variables) and change to view the surface being changed interactively. The program runs in real time and gives a better understanding of the role played by each of the different design variables. A separate window is provided in PRISM which displays the numeric value of each of the design variables and also interactively shows the number being changed. The surface can be rotated and zoomed in and out. Once

the user is satisfied with the particular shape of the surface, it can be written out in a seperate file. Figure 3.1 shows the snapshot view of the software program PRISM. The wireframe surface mesh is shown with different values of the design variables in a seperate window. Figure 3.2 shows the change in airplane geometry when the parameters defining the wing, fuselage and grid concentration are changed. For convinence the program is also menu driven where the different options can be displayed by clicking the right mouse button. The rendering of surfaces in PRISM is done to better understand the curvature and roughness of the surface and is explained in Sec. 3.3.

3.3 Shaded-Image Rendering

High-resolution shaded raster images provide concrete visualizations of computer-generated surfaces. When features such as shadowing, specular reflection, and depth cueing are included, and the user is free to manipulate the viewpoint and the positioning and intensity of the light sources, such images are extremly useful in understanding curved surfaces.

The basic problem in generating high-resolution raster images is computing the intersections of a set of rays from the viewer's eye with the surface. The approach to this problem depends on the surface formulation in use and the level of accuracy desired.

A three-dimensional ray can be regarded as the intersection of two planes

$$a1x + b1y + c1z + d1 = 0a2x + b2y + c2z + d2 = 0 \tag{3.1}$$

assumed to be nonparallel. The planes cut the surface in algebraic curves with polynomial equations of the form

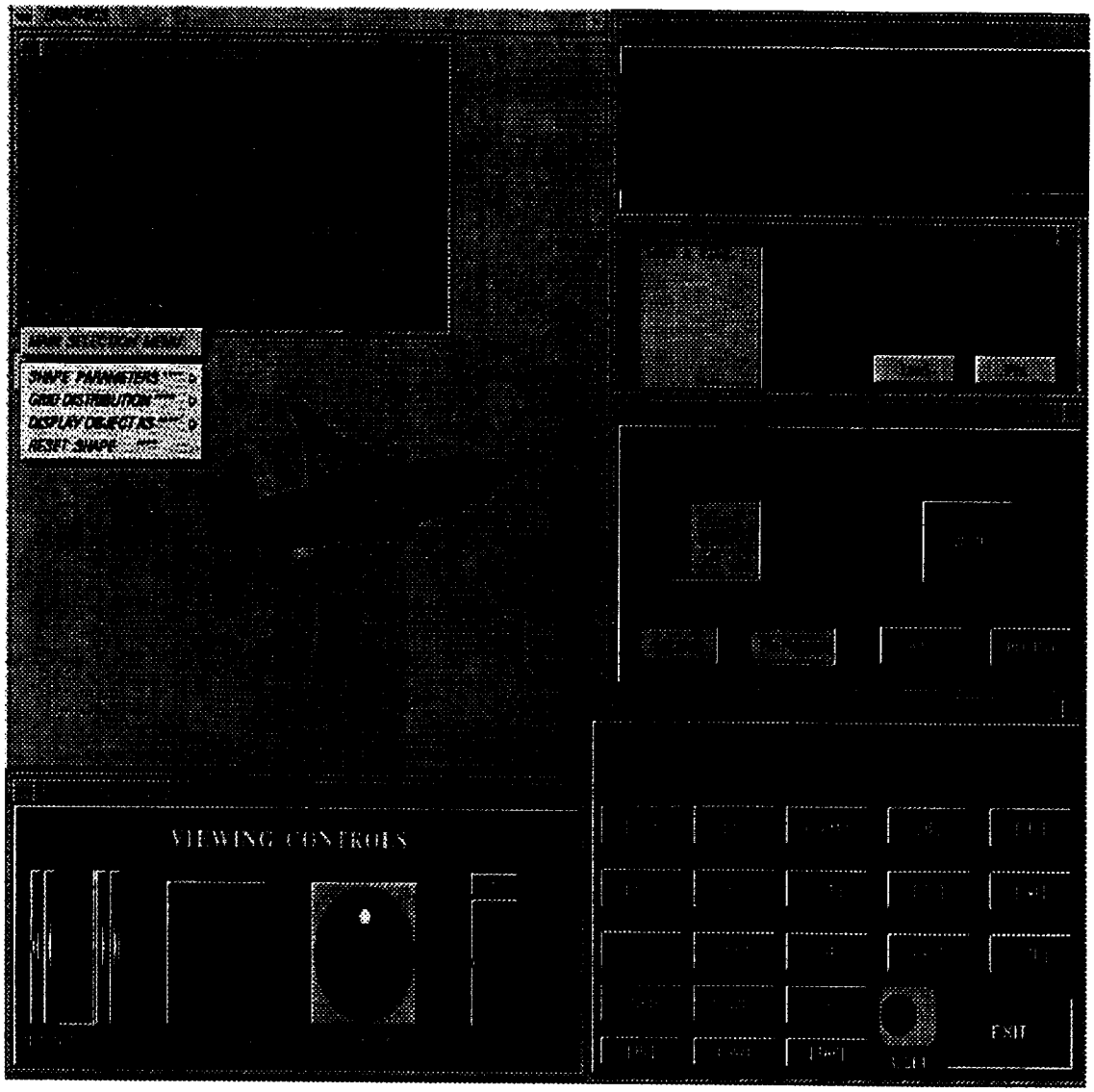


Fig. 3.1 Snapshot of the interactive software PRISM.

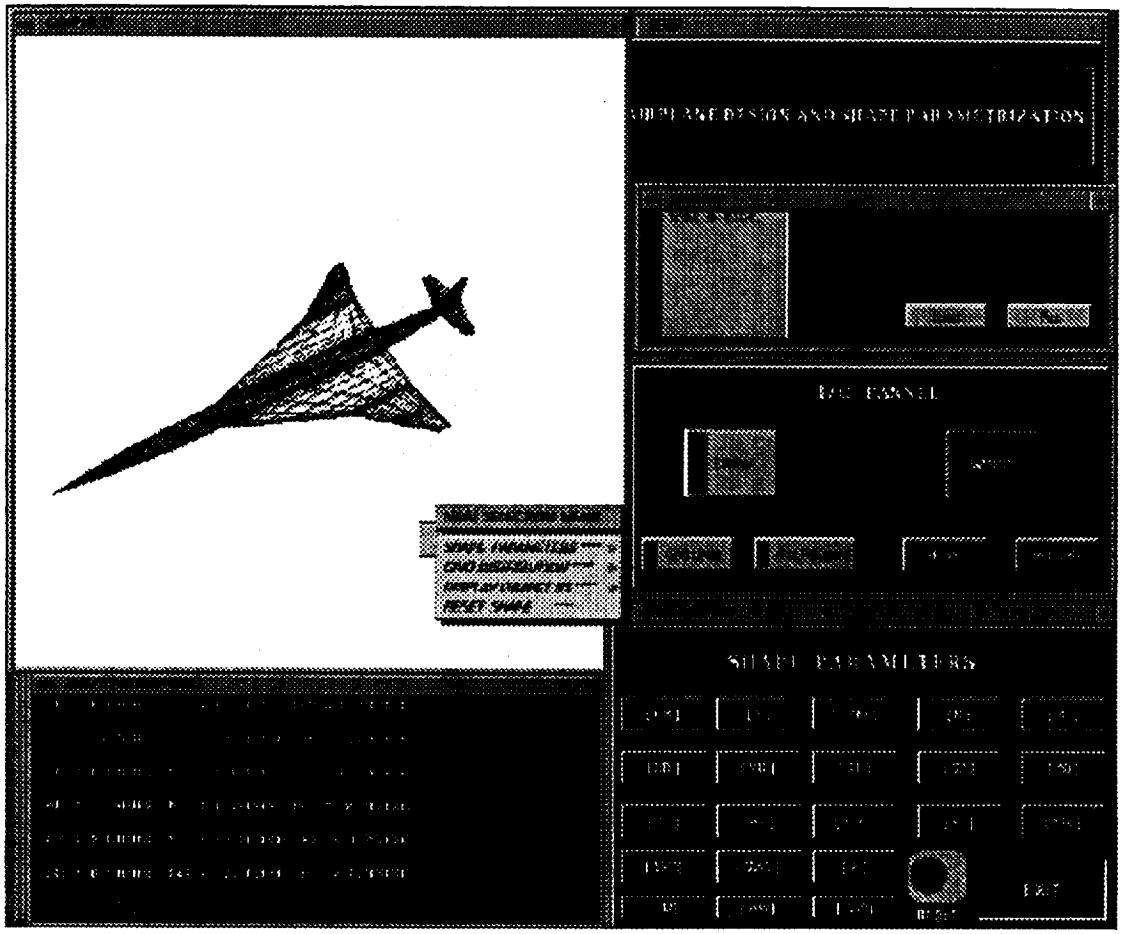


Fig. 3.2 Snapshot of the software with the surface being modified.

$$F1(u,v) = 0 \ , \ F2(u,v) = 0$$

in the parameter space of the patch. These are obtained by direct substitution of the parametric surface equation r = r(u,v) into the plane equations given by Eq. (3.1). A simple lighting model for shading surface images typically includes ambient and directional light sources. The ambient component produces a uniform level of surface illumination, independent of viewing direction, while the directional components produce both diffuse and specular reflections with intensities depending on the angles between the surface normal and the viewing and illumination directions.

Buttons are provided on the screen which when activated, render the surface of the geometry displayed as shaded. Colors are also provided for the user to choose for the surface rendering. A Toggle switch is provided which alternates the surface between rendering and wireframe. The surface is illuminated from a particular fixed direction which cannot be changed by the user.

3.4 Interactive HSCT Shape Design

To demonstrate the capabilities of the interactive program PRISM, a generic airplane shown in Fig. 3.3 is considered. The airplane defined with twentyone design variables and the values of each of these are shown in a separate window. The objective of the transformation process is to obtain the HSCT type configuration shown in Fig. 3.4c, by interactively changing the values of the design variables.

The design variables which are changed are ch; the wing chord length at mid section, XD; x-coordinate of wing trailing edge mid section YD; y-coordinate of wing trailing edge mid section, H1; length of inner wing component, H2; length of outer wing component, B; chord length for root wing section, TAP; ratio of thickness at root to thickness at midsection, XTL; chord length of outboard wing section, XT; x-translation of outboard wing section, TL; length of fuselage, XTE; x-translation

of wing relative to fuselage, AO; maximum fuselage diameter, A1; fuselage tapper parameter and A2; parameter controlling end fuselage diameter. The snapshot of the different shapes attained during the interactive process is shown in Fig. 3.4. The method is extremely helpful in investigating the shapes for different airplane design.

Four different airplane configurations generated with the help of PRISM are shown in Figs 3.5-3.8. Figure 3.5a shows the generic airplane defined by twentyone design variables. The software program PRISM has the capability to add or remove z-buffering and the user has the flexibility to adjust the direction of light. This feature is captured and shown in Fig. 3.5b. Configuration 2 consists of moving the wing below the symmetry plane, giving the look of a high lift configuration. Here the fuselage diameter is increased and Fig. 3.6a shows the snapshot view. The position of wing and the ducktail fuselage is shown in Fig 3.6b. Configuration 3 which represents the HSCT type configuration is shown in Fig. 3.7a. This configuration is used in this study for a detailed analysis and results are presented in latter sections. Figure 3.7b shows the view from below. Figure 3.8a shows the fourth configuration made up of a delta wing, and in Fig. 3.8b a different view is shown. These figures not only shows the capability of the software PRISM, but also the flexibility of generating different airplane configurations using the PDE methodology.

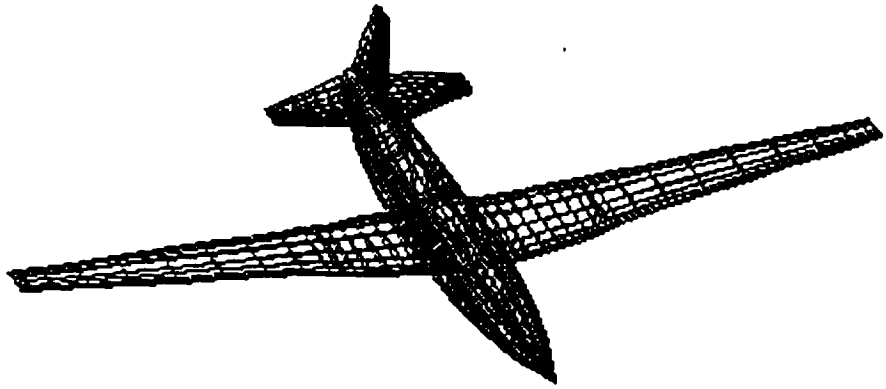


Fig. 3.3 Generic Airplane.

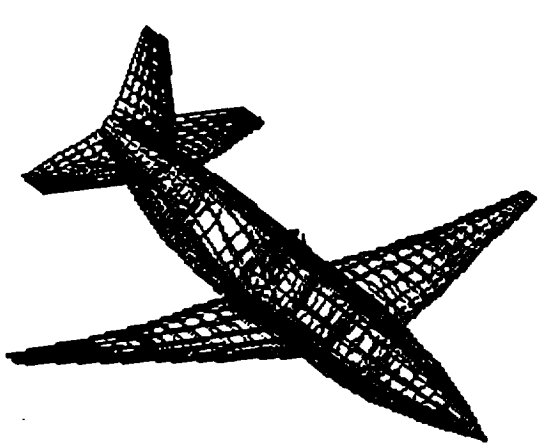


Fig. 3.4a Stage 1.

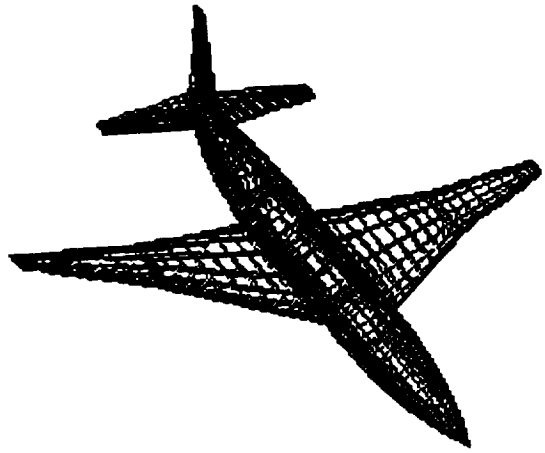


Fig. 3.4b Stage2.

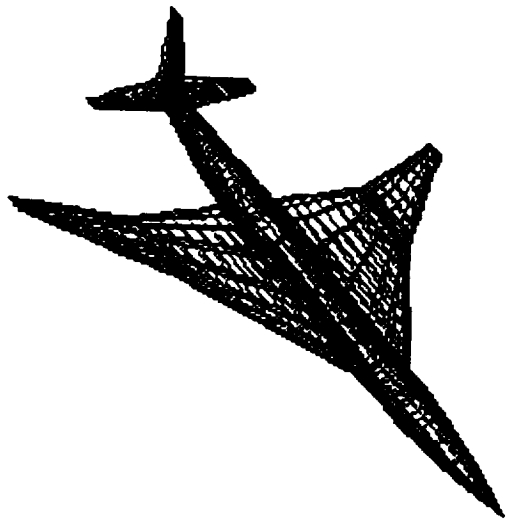


Fig. 3.4c Final HSCT configuration.

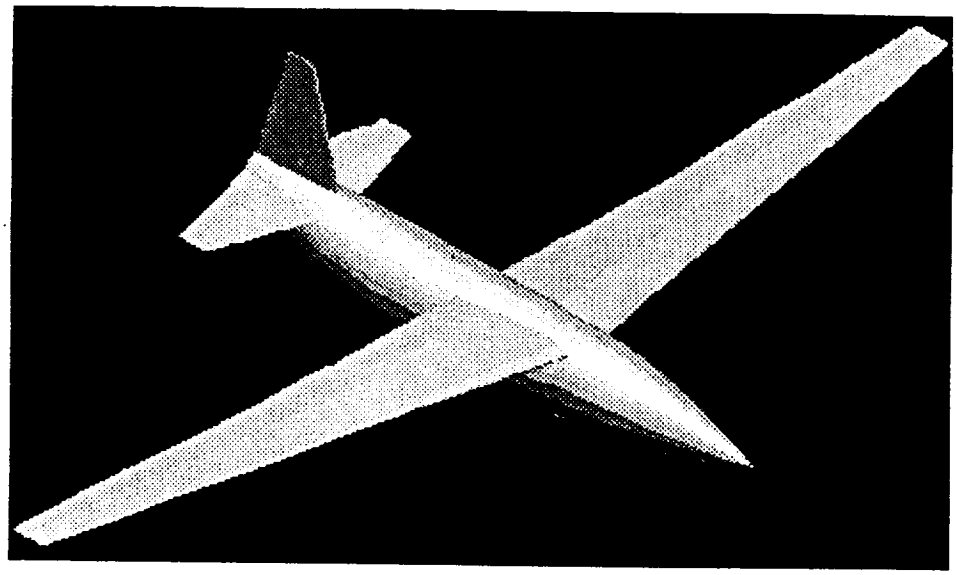


Fig. 3.5a Generic airplane.

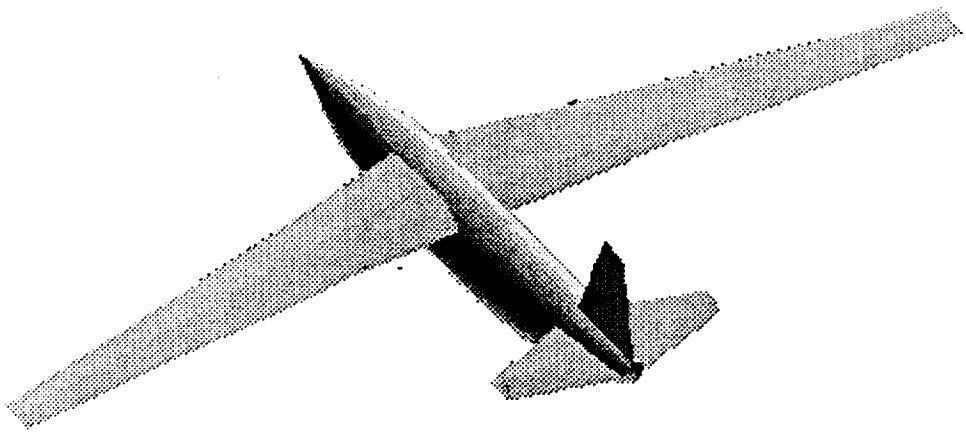


Fig. 3.5b Generic airplane with different z–buffering and different intensity of light.

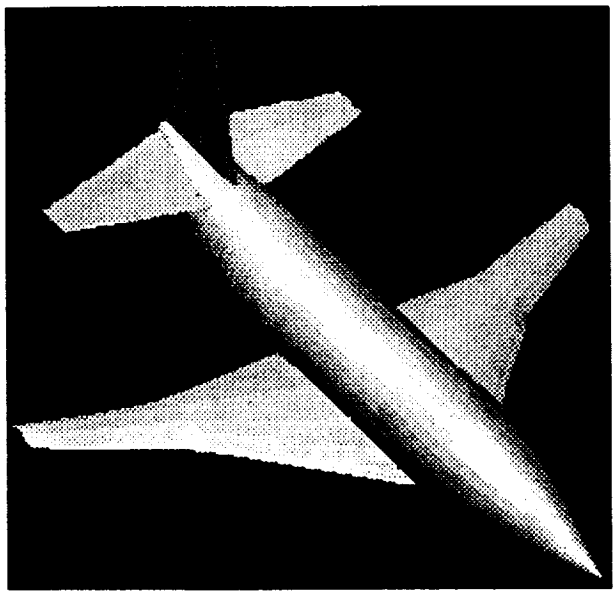


Fig. 3.6a Configuration 2 (low wing).

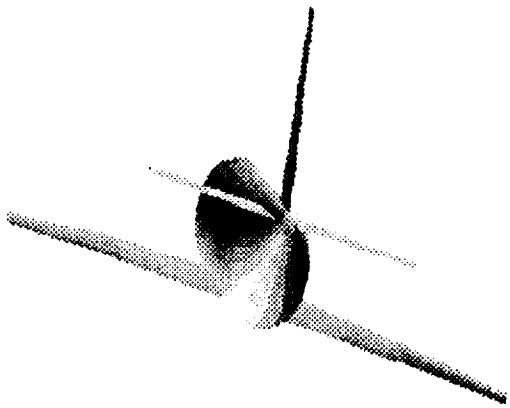


Fig. 3.6b Configuration 2 showing the low wing.

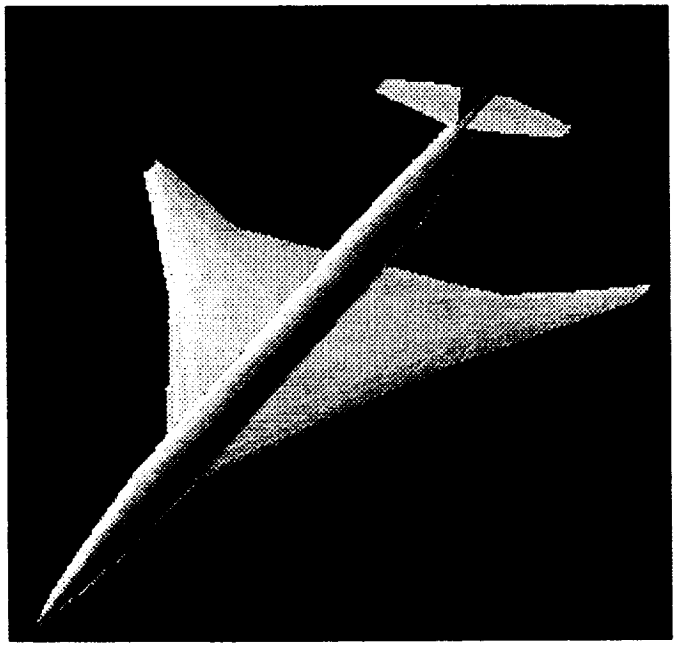


Fig. 3.7a Configuration 3(HSCT type).

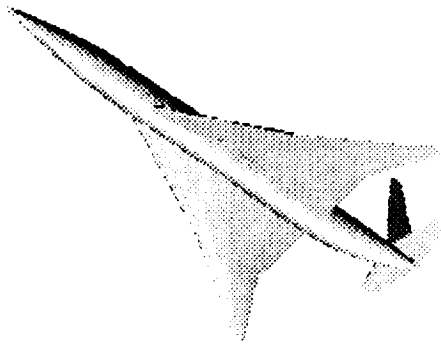


Fig. 3.7b Configuration 3 viewed from below.

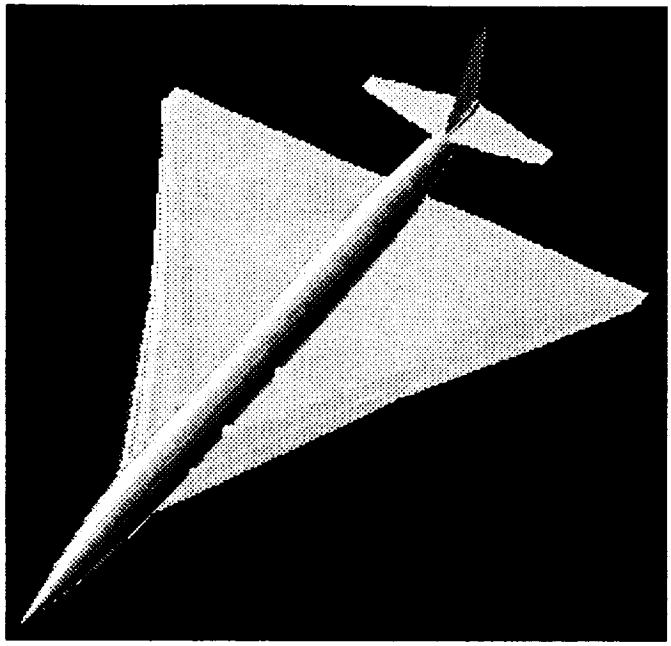


Fig. 3.8a Configuration 4 (delta wing).

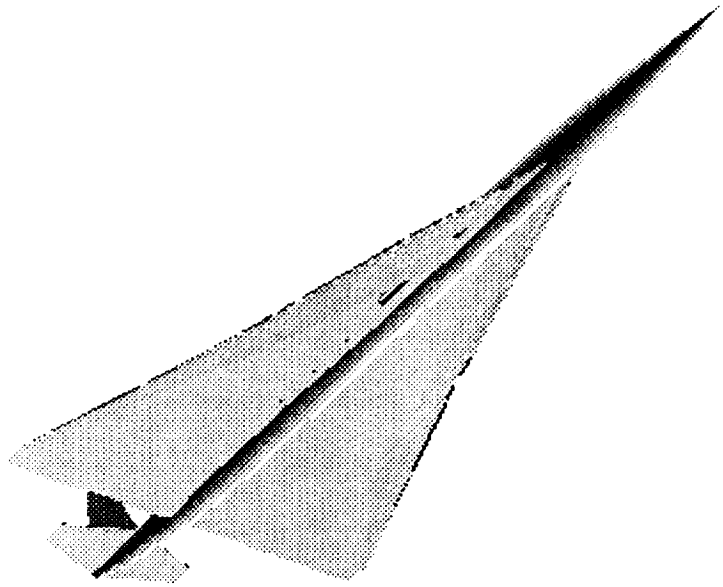


Fig. 3.9b Configuration 4 viewed from below.

Chapter 4

GRID GENERATION

4.1 Introduction

In recent times, techniques for the automatic generation of computational meshes have received much attention. This is primarily due to the fact that there has been an increased effort in the development of algorithms for the solution of the flow-field equations. Historically, many of the fundamental developments in the theoretical fluid dynamics have rested upon conformal mapping techniques for incompressible potential flow in which solutions on the boundaries can be obtained without resort to information in the field. Also panel methods, which utilize distribution of sources and sinks on boundary surfaces, have played and continue to play an important role in aerodynamics. Recently, however, attention has been primarily focused on solution techniques for the Full Potential, Euler and Reynolds-Averaged Navier-Stokes equations. These equations are formulated on the basis of the continum hypothesis. With computers restricted in speed it is not possible to consider all points within a domain at which flow quantities can be calculated. The combination of points and connections between points defines a mesh or grid on which numerical methods for the solution of the flow equations can be constructed. The assumption is then made that the information at these points is sufficient to describe the complete flowfield.

In order to study the flow-field around any aerodynamic configuration, a system of nonlinear partial differential equations must be solved over a highly complex geometry [70]. The domain of interest should be discretized into a set of points where

an implied rule specifies the connectivity of the points. This discretization, known as grid generation, is constrained by underlying physics, surface geometry, and the topology of the region where the solution is desired [71-72]. A poorly constructed grid with respect to any of the above constraints, may fail to reveal critical aspects of the true solution.

The discretization of the field requires some organization in order for the solution to be efficient. The logistic structure of the data such as grid spacing, the location of outer boundaries, and the orthogonality can influence the nature of the solution [73]. Furthermore, the discretization must conform to the boundaries of the region in such a way that boundary condition can be accurately represented [74]. This organization can be provided by a curvilinear coordinate system where the need for alignment with the boundary is reflected in routine choice of Cartesian coordinate system for rectangular region, cylindrical coordinate for circular region, etc. This curvilinear coordinate system covers the field and has coordinate lines coincident with all boundaries. To minimize the number of grid points required for a desired accuracy, the grid spacing should be smooth, with concentration in regions of high solution gradients. These regions may be the result of geometry (large surface slopes or corners), compressibility (entropy and shock layers), and viscosity (boundary and shear layers). A complex flow may contain a variety of such regions of various length scales, and often of unknown location.

Two primary categories for arbitrary coordinate generation have been identified. These are algebraic systems and partial differential systems. The algebraic systems are mainly composed of interpolative schemes such as *Transfinite Interpolation* [75], Multi-Surface Interpolation [76], and Two-Boundary Interpolation techniques [77]. The basic mathematical structure of these methods are based on interpolation of the field values from the boundary. For partial differential equation systems, a set of partial differential equations must be solved to obtain the field values.

The differential methods may be elliptic, parabolic, or hyperbolic, depending on the boundary specification of the problem. Each of these grid generation systems has its own advantages and drawbacks depending on geometry and application of the problem. Algebraic generating systems offer speed and simplicity while providing an explicit control of the physical grid shape and grid spacing. However, they might produce skewed grids for boundaries with strong curvature or slope discontinuity. Partial differential systems, although offer relatively smooth grids for most applications, are computer intensive, specially for three-dimensional cases. An alternative, a common practice in recent years, has been to originate the grid using an algebraic system and then smooth the field using a differential system. Such hybrid approach has proven to be successful and cost effective for most applications.

For complex geometries the multiblock mesh generation strategy is utilized. The idea behind multiblock mesh generation is that, instead of utilizing one global curvilinear coordinate system, several local curvilinear systems are constructed and connected together. The domain is subdivided into blocks and within each block a curvilinear system is derived. The block subdivision provides the necessary flexibility to construct structured meshes for complex geometrical shapes. The approach represents a compromise between a globally structured mesh and an unstructured mesh.

An array of general purpose grid generation softwares have emerged over the past few years. Among many others, the GRAPE2D of Sorenson [78], the EAGLE of Thompson [79], and GRIDGEN by Steinbrenner et al. [80] are the most widely used. The GRIDGEN series has both algebraic and differential generation capabilities on an interactive environment. The GRAPE2D solves the Poisson's equation in two-dimension and utilizes a novel approach for determination of the boundary control functions. The EAGLE code combines techniques in surface grid generation as well as two or three-dimensional field grid generation. The ICEM/CFD has the capability of combining a full Computer Aided Design system (CAD), with grid generation

module [81]. This provides an efficient and also quick procedure to reflect the CAD model changes on grids. Most of these packages furnish a host of options with a high degree of flexibility. However, intelligent use of the majority of these options requires the user to be well versed in current grid generation techniques.

Over the past few years, an alternative technique, unstructured tetrahedral grids, has received considerable attention [82]. In an unstructured mesh, unlike structured mesh, neighbouring points in the mesh in the physical space are not the neighbouring elements in the mesh point matrix. For any particular point, the connection with other points must be defined explicitly in the connectivity matrix. A constant reference to this matrix is made during the flow solution computation. In addition to their inherent capability of discretizing complex domain with ease, unstructured grids are suitable for efficient adaptive refinement, incorporation of moving boundaries, and local remeshing. These grids also offer better control over the mesh size and point distribution. In other words, unstructured grids are more flexible than their structured counterparts simply because of their irregularities. While in structured grids, mesh lines and planes should be continuous and conform to the boundaries and adjacent lines and planes throughout a domain, no such restriction exists in unstructured grids due to their lack of directionality. Generally, since triangles and tetrahedra are the simplest geometrical shapes having areas and volumes, respectively, they can discretize an irregularly shaped domain easier than quadrilaterals and hexahedra. Furthermore, the number of neighbouring points surrounding each node in a structured grid is fixed, whereas in an unstructured mesh, this number varies from point to point. A consequence of this property of unstructured grids is that a large number of grid points on the surface of a geometry, where a fine resolution is required, do not have to be carried all the way to the outer boundaries where fewer points are needed.

There is a variety of methods for generation of unstructured grids in the literature. Among these are Watson's algorithm for Vornoi tessellation [83], the modified octree method [84] and the advancing front technique [85]. In this study, the software VGRID3D, a program for generation of three dimensional unstructured tetrahedral inviscid grids using the advancing front method [86] is used. This method is advocated here because it does not require a seperate library of modules to distribute grid points throughout the domain in advance like the Voronoi/Delauny family of unstructured grid generation techniques.

4.2 Structured Grid Generation

The majority of problems in physics and engineering can be described in terms of partial differential equations [87]. Many of these problems fall naturally into one of the three physical categories: equilibrium problems, eigenvalue problems and propogation problems. However, before solving such problems by numerical methods, a system of partial differential equations should be solved to determine the mesh. The properties of meshes generated by this approach are intimately connected to the properties of the partial differential equations used as the mesh generation equations.

Equilibrium problems are problems of steady state in which the equilibrium configuration is determined by solving a differential equation subject to boundary conditions. Such problems are known as boundary value problems and the governing equations for equilibrium problems are elliptic.

Eigen value problems may be thought of as extensions of equilibrium problems wherein critical values of certain parameters are to be determined in addition to the corresponding steady-state configurations. Propogation problems are initial value problems that have an unsteady state or transient nature. The problems involve the prediction of the subsequent behavior of a system given the initial state. The

governing equations for propogation problems are parabolic or hyperbolic.

Structured algebraic grid generation techniques can be thought of as transformation from a rectangular computational domain to an arbitrarily shaped physical domain as shown in Fig. 4.1 [88]. The transformation is governed by vector of control parameters, $\mathbf{P}$, and can be expressed as

$$\mathbf{X}(\xi,\eta,\zeta,\mathbf{P}) = \begin{Bmatrix} x(\xi,\eta,\zeta,\mathbf{P}) \\ y(\xi,\eta,\zeta,\mathbf{P}) \\ z(\xi,\eta,\zeta,\mathbf{P}) \end{Bmatrix} \tag{4.1}$$

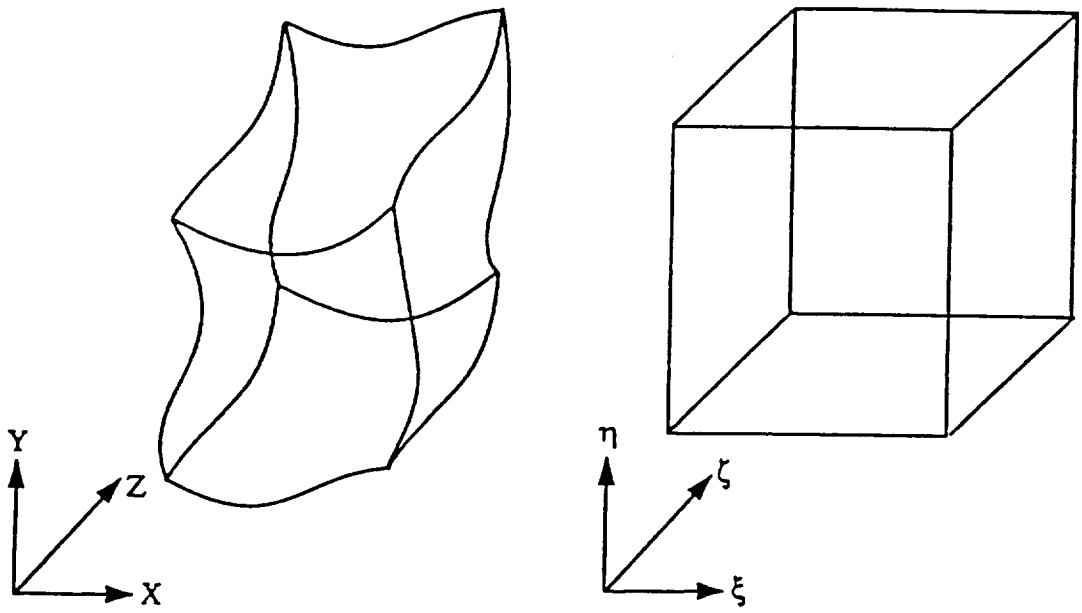
where

$$0 \leq \xi \leq 1, \quad 0 \leq \eta \leq 1, \quad \textit{and} \quad 0 \leq \zeta \leq 1.$$

The control parameter $\mathbf{P}$, is composed of parameters which control the primary shape of the boundary (design parameters), and parameters which control the grid (grid parameters). A discrete subset of the vector-valued function $\mathbf{X}(\xi_i,\eta_j,\zeta_k,\mathbf{P}) \equiv \mathbf{X}\{\ x\ y\ z\ \}^T_{i,j,k} \equiv \mathbf{X}^*$ is a structured grid for $\xi_i = \frac{i-1}{L-1}, \eta_j = \frac{j-1}{M-1}, \zeta_k = \frac{k-1}{N-1}$, where $i = 1,2,3\cdots,L$, $j = 1,2,3,\cdots,M$ and $k = 1,2,3,\cdots,N$.

Surface mesh generation is one of the most difficult and yet important aspects of the total mesh generation problem. The surface mesh influences the field mesh close to the configuration, where flow gradients are important and need to be resolved accurately. Surface meshes have the same requirements for smoothness and continuity as the field meshes for which they act as boundary conditions, but, in addition, they are required to conform to the configuration surfaces, including, lines of component intersection, and to model regions of high surface curvature.

In the software program RAPID geometric surfaces are generated using partial differential equation described in Sec. (2.6). The surface grid is created by evaluating the surface functions at discrete ξ(I) and ζ(K). In order to concentrate the grid in certain regions, such as wing/fuselage intersection, it is necessary to create control functions that map $0 \leq \xi,\zeta \leq 1$ into $0 \leq \xi,\zeta \leq 1$. The spacing of grid points within the topology constraints is very important for achieving acceptable accuracy



(a) Physical domain (b) Computational domain

Fig. 4.1 Physical and computational coordinates.

in the application of a flow analysis about the vehicle surface. A double exponential function [89] which maps the computational variables ξ, η, and ζ onto themselves is used here. The grid spacing control function is expressed as

$$\nu = K_1 \frac{e^{\frac{K_2}{K_3}\bar{\nu}} - 1}{e^{K_2} - 1},$$

$$0 \leq \bar{\nu} \leq K_3, \quad 0 \leq \nu \leq K_1,$$

$$\nu = K_1 + (1 - K_1)\frac{e^{K_4 \frac{\nu - K_3}{1 - K_3}} - 1}{e^{K_4} - 1},$$

$$K_3 \leq \bar{\nu} \leq 1, \quad K_1 \leq \nu \leq 1,$$

$$K_4 \ \ chosen \ \ \ni \ \ \frac{D\nu(K_3)}{D\bar{\nu}} \ \ \subset \ \ C^1. \tag{4.2}$$

Figure 4.2 is used to help describe the grid control parameters K_1, K_2, K_3, and K_4. Parameters K_1 and K_3 are coordinates of a point in the unit square. The quantity $\bar{\nu}$ is the independent computational variable and corresponds to the percentage of grid points in a particular direction. The quantity ν is the dependent computational variable and corresponds to the percentage of distance in the physical space along a grid curve. The parameters K_2 and K_4 are coefficients in the exponential functions defined for a particular part of the unit square. Where there is low slope in the control functions, there is a concentration in the grid points, and where there is high slope, there is dispersion in the grid points. In the RAPID methodology, Eq. (4.1) is used several times. The approach specifies a desired spacings at the $\bar{\nu} = 0$ and/or at $\bar{\nu} = 1$ and/or K_3. The parameters K_1, K_2, and K_4 are determined by a Newton-Raphson process while satisfying a first derivative continuity condition at (K_3, K_1). Figure 4.3 shows the grid distribution achieved on the fuselage by using Eq. (4.1) and Fig. 4.4 shows the grid distribution on the wing and wing fuselage intersection.

The grid control parameters are distinguished from the configuration design parameters. The design parameters are referred to as the set $\mathcal{P}$, and the grid

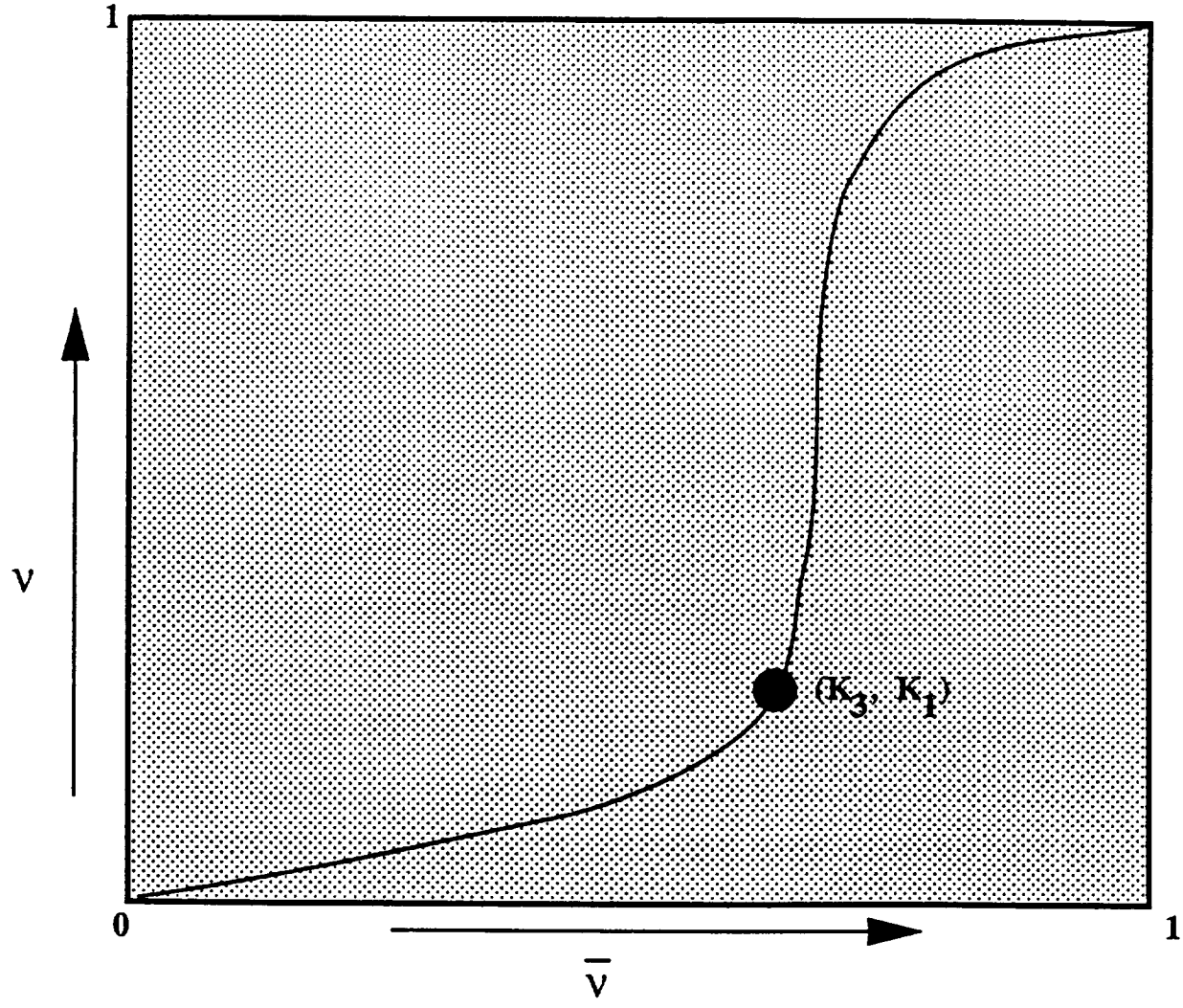


Fig. 4.2 Grid spacing control function

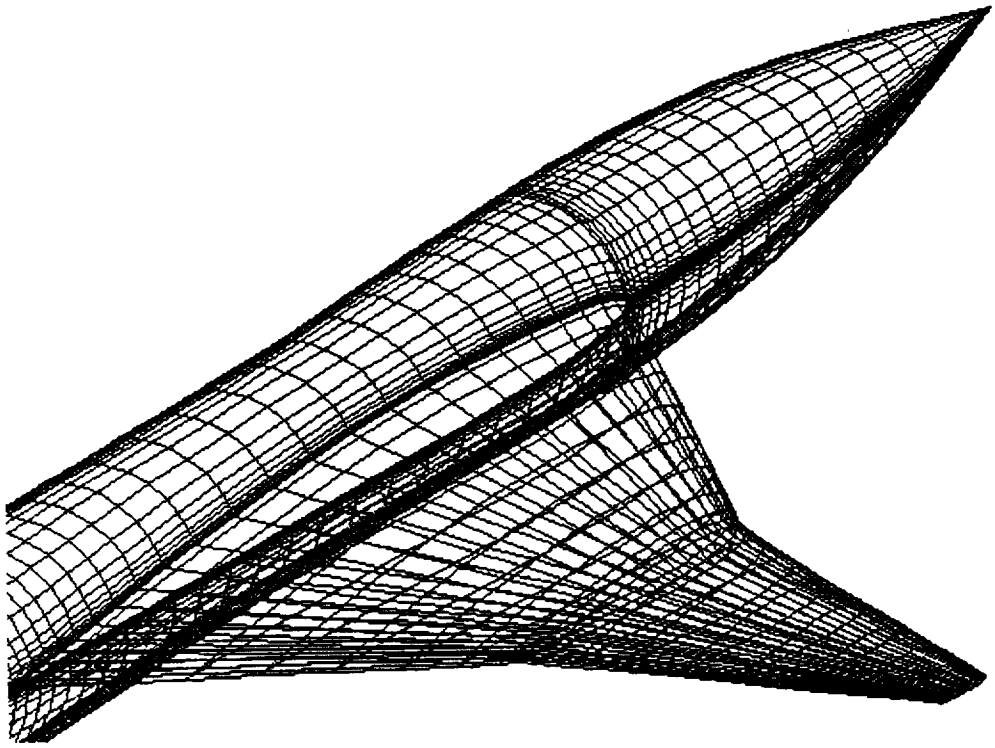


Fig. 4.4 Grid distribution on the surface of the wing and the nose of the fuselage.

parameters are referred to as the set $\mathcal{K}$. $\mathcal{K}$ which includes the grid spacing parameters described above and the volume grid control points discussed in the next section.

4.2.1 Boundary Discretization and Volume Grid Around The Airplane Geometry

The orientation of the computational coordinates relative to physical coordinate, known as grid topology, is an important aspect of the transformation procedure. In order to establish a grid topology for any geometry, it is essential to examine each component separately [90]. For any given geometry, there are several possible topologies with different characteristics in terms of efficiency, coordinate cuts, singularities, etc. For example a typical wing-section geometry, may have at least three types of different topologies (e.g., C-, O-, or H-types). The C- and O-type topologies usually produce the most efficient grid. This topology produces no singularity and it is relatively simple to implement. For wing-sections with sharp noses, a H-type topology would be more appropriate. For more complex geometries, selection of different computational coordinate systems for different regions of physical domain might be required. In this case, physical domain is mapped into several computational sub-domains, where each sub-domain is reffered as a block. Therefore, it is possible to have a boundary-fitted coordinate system for a highly complex configurations. For the present study, the airplane geometry consists of two main components: the fuselage and the wing. The fuselage has a circular like cross-section which suggests that a natural O-type (cylindrical coordinate) grids. This topology produces a nearly orthogonal grid with one line polar singularity at the nose. For the streamwise direction, it is feasible to have either a C-type or a H-type grid depending on the slope of the nose. For a fuselage with small nose slope, a H-type grid in the streamwise direction would be more appropriate. A wing has its own natural coordinates which are usually not compatible with the fuselage's coordinate system. It is possible to generate a H-, O-, or a C-type grids in the streamwise direction, and a C- or a H-type

in crosswise direction. To maintain a minimum of C^0 continuity at the interfaces, it is essential to select a compatible topology for the wing and fuselage. For most cases it is conceivable to generate a single block grid about these components, but this grid tends to be skewed for any practical purposes. A dual-block grid possesses much less skewness than a single-block grid. It consists of two large blocks, one covering the top portion of the physical domain, and the other covering the bottom portion of the physical domain. The dual-block topology is a direct consequence of using a H-type grid for the wing of zero wing-tip area. Figure 4.5 illustrates the mapping of a generic airplane geometry using a dual-block topology. A C-O type grid have been chosen for a fuselage while the wing, horizontal, and vertical tails mapped to a H-H type grids.

A Control Point Form/Transfinite Interpolation technique[91] is used to compute volume grids for the RAPID methodology. A considerable amount of information has been published on this grid generation method and its variations, and only the major steps are presented here.

Having established a grid on the configuration surface, the volume grid generation is accomplished in four major steps described below.

Step 1 is the determination of a grid in the symmetry plane. The basic functions used in RAPID are those for Bézier curves computed with the de Casteljau scheme[92]. Control points for an intermediate curve and for a far-field curve are computed from the dimensions of the fuselage, Fig. 4.6 . A set of points are distributed in the ξ-direction on the control curves obtained from the control points. Interpolation from the fuselage surface across the control curves is obtained with a de Casteljau application in the η-computational direction, and Fig. 4.7 shows the symmetry grid.

Step 2 is the determination of a three-dimensional grid surface containing the lifting components shown in Fig. 4.8. Note that in the H-topology, the top

and bottom grids are considered separately. A process similar to that used with the symmetry grid for computing control points from the fuselage and lifting surfaces is applied.

Step 3 is the determination of a cap grid. Control points are extracted from the extreme x and y grid coordinates in the lifting surface grid and the extreme z-grid coordinates in the symmetry plane grid. This is shown in Fig. 4.9. Casteljau scheme is applied with these control points, and the outer grid surface is shown in Fig. 4.10.

Step 4 is the application of Transfinite Interpolation to compute the interior grid. Figure 4.11 shows a sample grid around the HSCT type configuration. .

It is necessary to use several grid-spacing control functions and their control parameters in addition to the interpolation control points in order to achieve a good grid for a given set of design parameters. This requires some trial and error before acceptable parameters are realized. However, once an acceptable set of grid parameters $\mathcal{K}$ is found for a given set of design parameters $\mathcal{P}$, small changes in $\mathcal{P}$ do not require changes in $\mathcal{K}$. Therefore, repetitive small changes in the design parameters such as during configuration optimization, do not require the constant modification of the grid parameters. Also note that the volume grids obtained with this algorithm are computed only out to the wing tip. An additional far-field grid would be necessary for most high-level fluid analyses.

A complete volume grid which extends beyond the tip of the wing surface is computed by GRIDGEN software. A comparitive study of the grids generated from RAPID is made with standard grid generation software GRIDGEN. Among the different softwares available, the GRIDGEN software developed by MDA Engineering is used to develop grids around surfaces generated from RAPID.

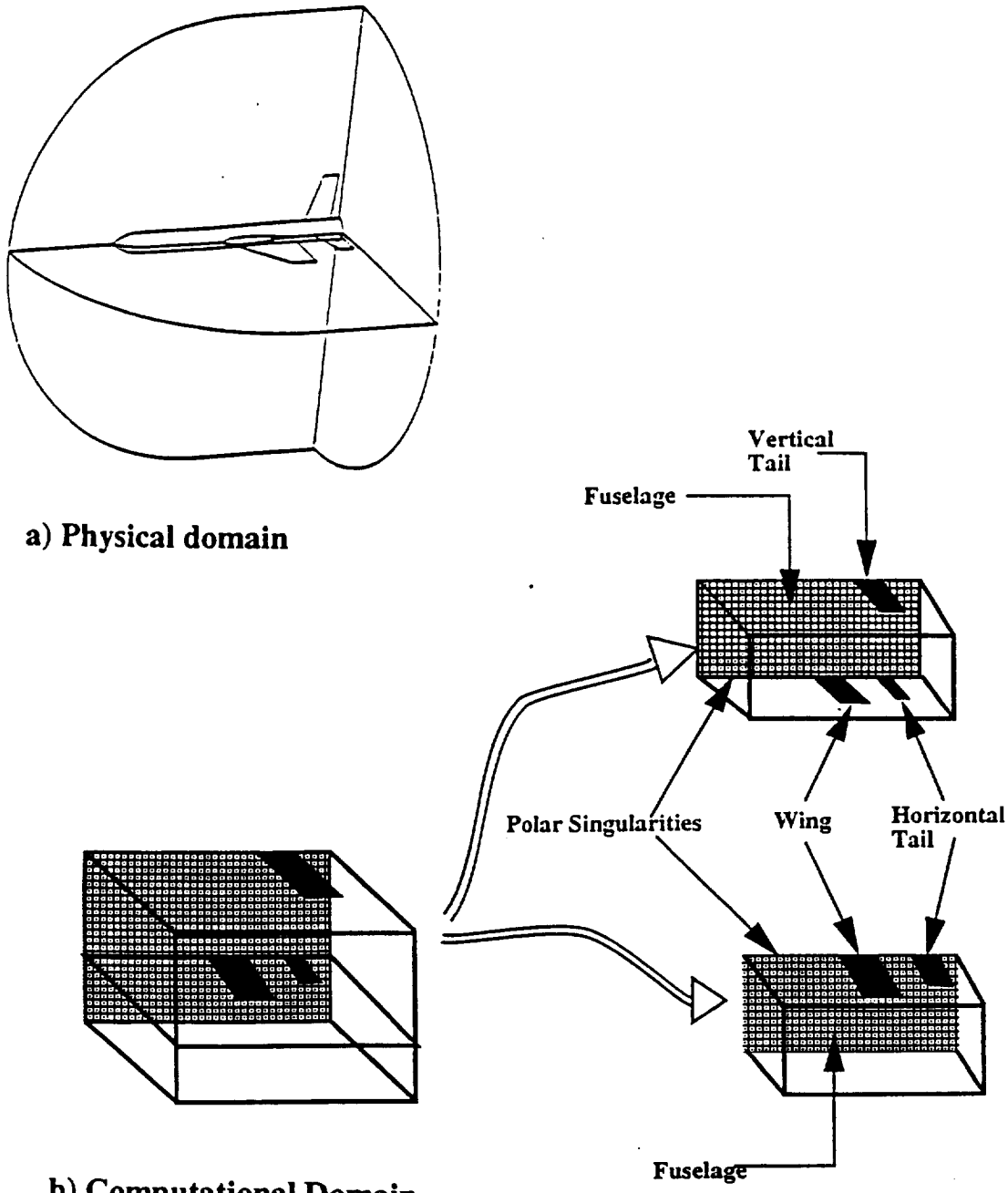


Fig 4.5 Dual-block grid topology for a generic airplane.

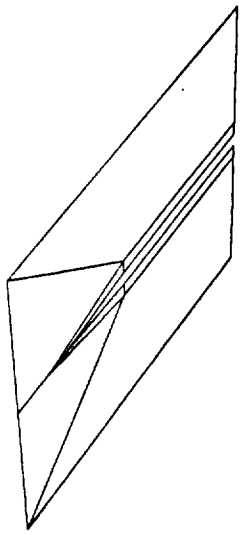


Fig. 4.6 Symmetry plane control net.

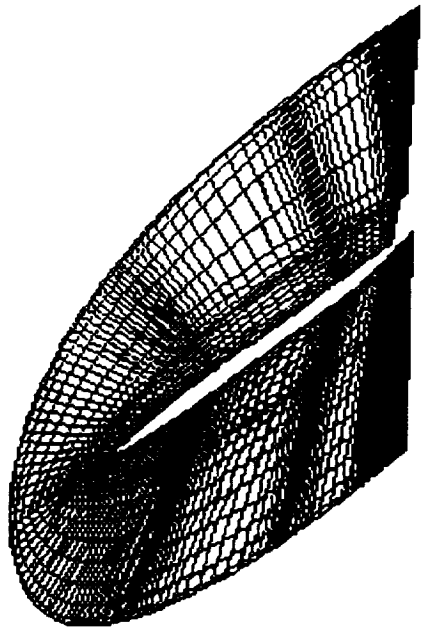


Fig. 4.7 Symmetry grid.

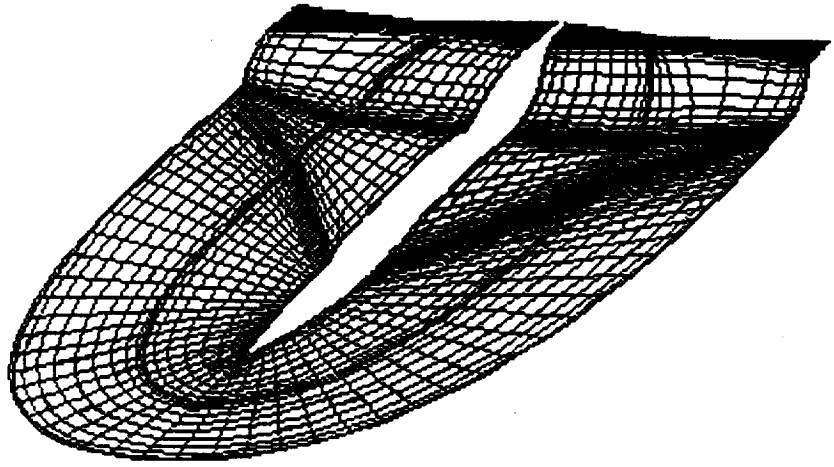


Fig. 4.8 Surface grid containing lifting Components.

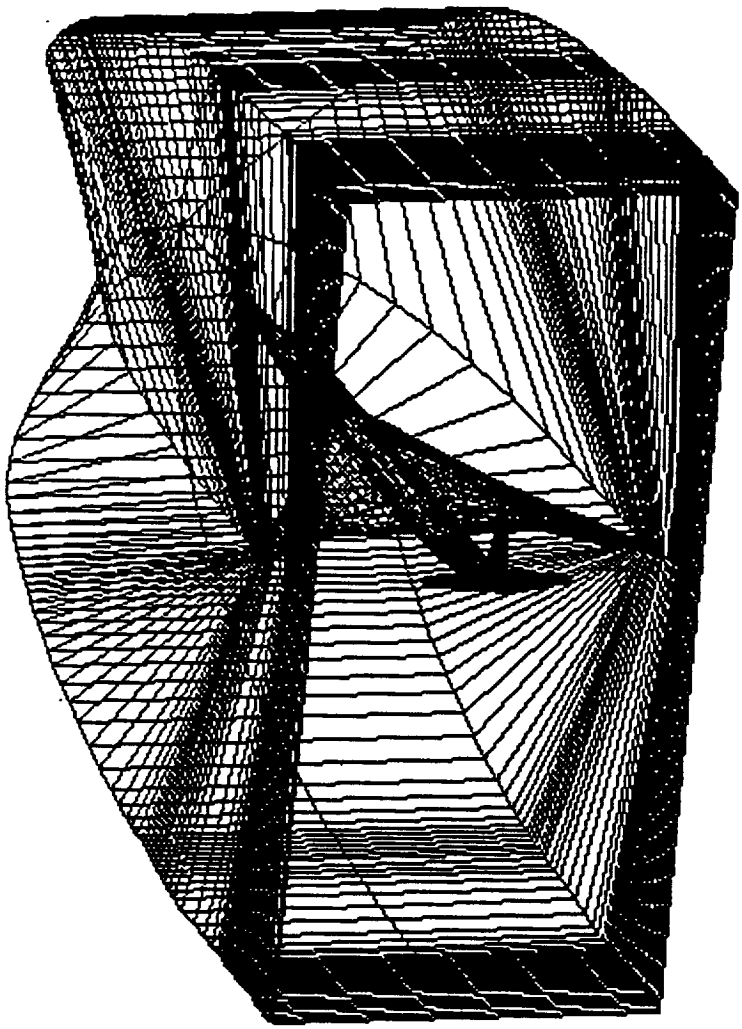


Fig. 4.9 Control point for outer grid surface.

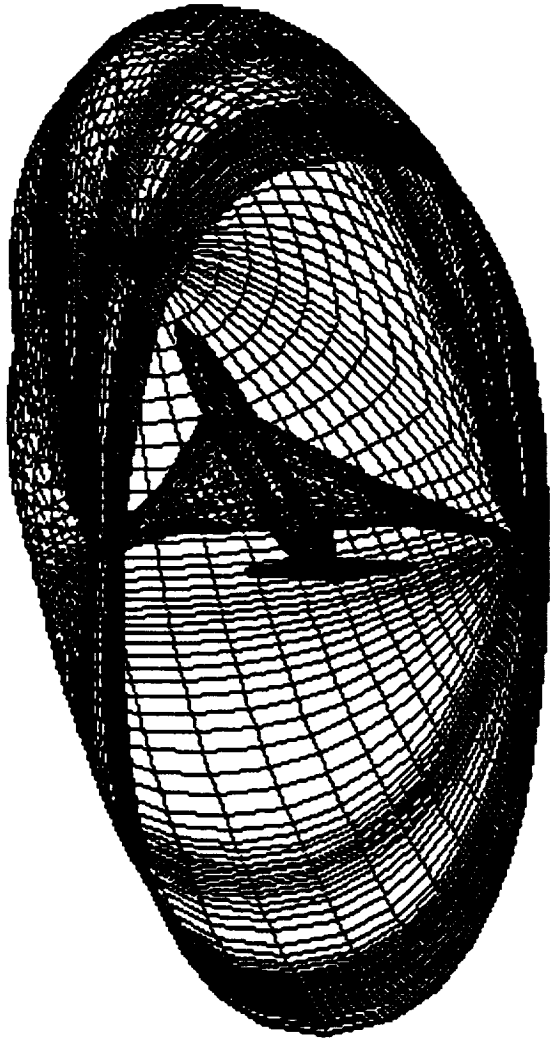


Fig. 4.10 Outer Grid Surface.

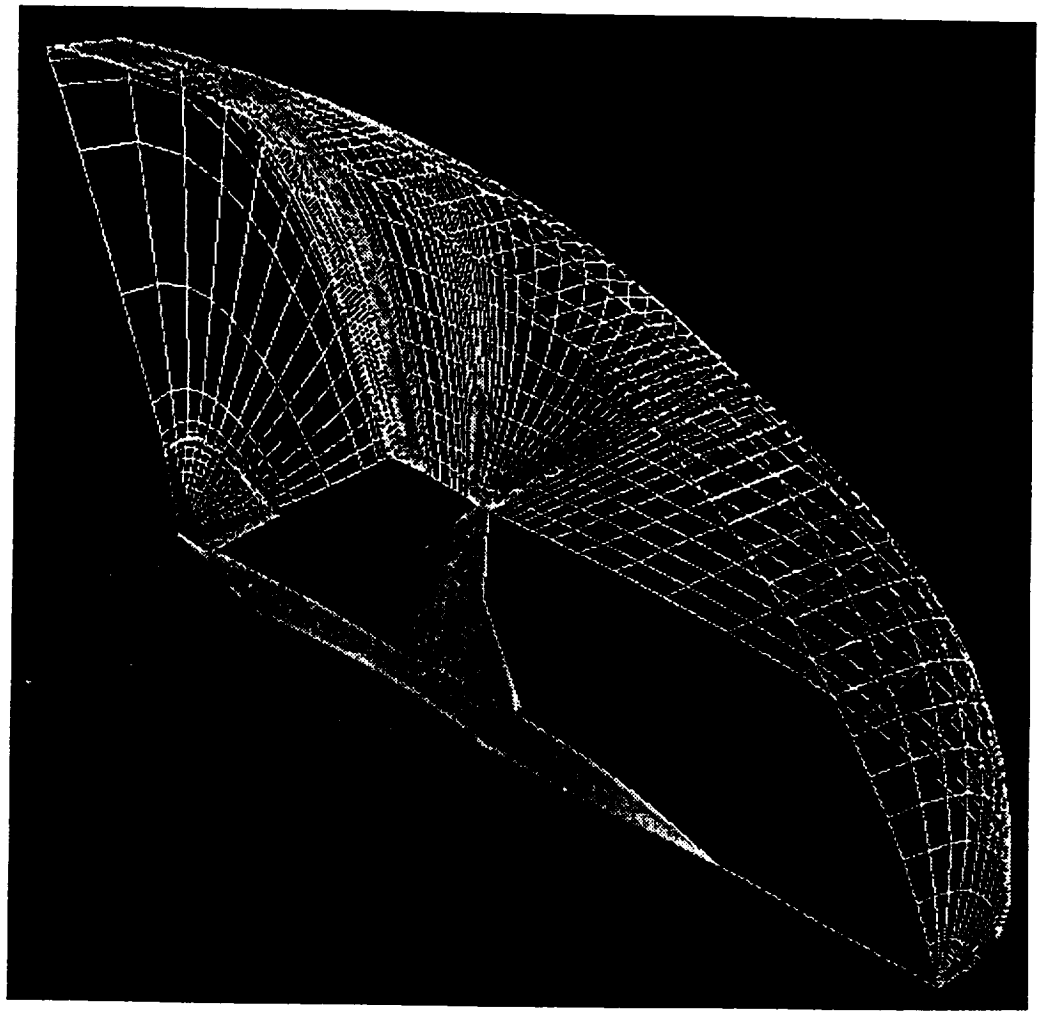


Fig. 4.11 Sample Grid Surfaces.

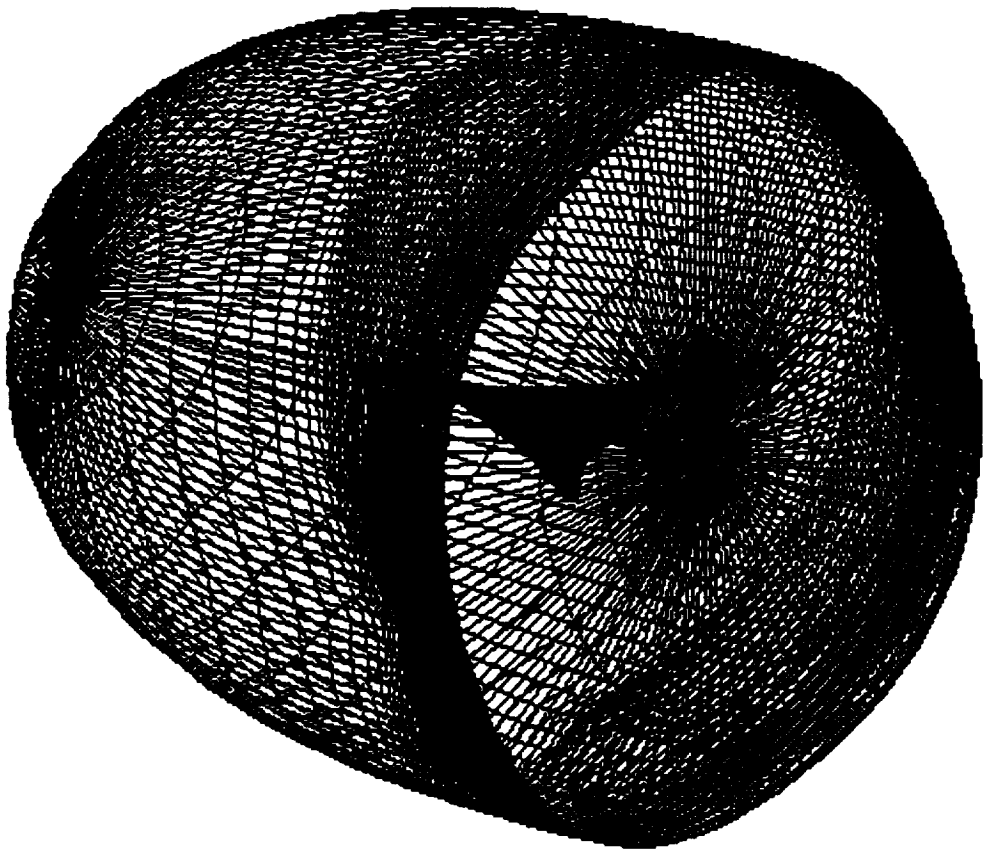


Fig. 4.12 Volume grid around the PDE surface using GRIDGEN.

The GRIDGEN software consists of three main modules namely, GRIDBLOCK, GRIDGEN2D, and GRIDGEN3D. The GRIDBLOCK begins with a display of the 3D database of the airplane configuration. The 3D lines representing the bounding edges of the blocks are drawn. Once several connectors are added to the system, they are grouped together and assigned to blocks. Then computational directions and dimensions on that block are defined. In the end flow boundary conditions and interblock connections are determined and assigned. The GRIDGEN2D is used to generate grid on the edges and surfaces. There are five modes of elliptic solvers in GRIDGEN2D. The first three solve directly for the Cartesian grid point coordinates in an iterative process and the next two solve the grid in parametric coordinates. This surface grid generation procedure is repeated for each surface in the face, and for each face in the block. The third and final step of the grid generation process is the distribution of grid points within the interior of each block. This task is performed with the batch code of GRIDGEN3D. Figure 4.12 shows the volume grid generated around the PDE surface.

4.3 Unstructured Grids

One of the greatest concerns in computational fluid dynamics is the generation of suitable grids. Although considerable effort has been devoted towards development of robust and automatic grid generation methods, the process of generating 3-D grids around complex geometries remains a formidable challenge. With the availability of large supercomputers, it is now possible to compute flowfields around complex configurations in a matter of hours. However, the process of grid generation, using conventional structured grid methods, still makes up a large portion of a typical computational effort for a complex configuration. Use of unstructured grids has grown considerably in recent years due to their ability to produce quality grids

around complex configurations with relative ease.

In recent years a wide variety of algorithms has been devised for the generation of unstructured grids around bodies of complex geometrical shapes. Among the different techniques are the Watson's algorithm for Vornoi tesselations, the modified octree method [94] and the advancing front technique. Baker's implementation and optimization of the Vornoi algorithm [93] has shown that fast and reliable grid generators for tetrahedral meshes can be produced. In this study, advancing front technique is used for grid generation, because it can easily be used for grid generation with directional refinement. Also it does not require a separate module to distribute points like the Delauny triangulation.

4.3.1 Advancing Front Technique

In the advancing front method, a grid is generated starting from the domain boundaries marching towards the interior of the computational domain. Unlike Delauny triangulation technique in which grid points are first distributed in the entire field and then connected to form cells, an advancing front introduces new points to the domain as tetrahedrons are made. The configuration of interest is first defined in terms of a number of surface patches. These patches are then triangulated to form the initial front. The front is then projected to the original surface which in this case is the NURBS and PDE surface. Next, tetrahedral cells are generated on top of triangular faces on the front by introducing new or using existing points. During this process, old faces are replaced by new ones, and the front is advanced in the field until the whole region is filled with grid cells.

The entire grid generation process is summarized in the following main steps:
(a) The boundaries of the domain to be grided are divided into a number of surface patches. These surfaces define the configuration of interest as well as the far-field boundaries.
(b) A background grid is set up to define the local grid characteristics such as grid

point spacing. The spacing interpolation in the VGRID system is based on a structured background grid [94]. This technique simplifies the specifications of grid density by introducing nodal and linear sources. The contribution from nodal sources are inversely proportional to the square of the distance and the contribution from the linear sources are modeled similar to the diffusion equation.

(c) Each surface patch is, in turn, subdivided into a number of triangles to form the first front (surface grid).

(d) The triangles are then projected on to the actual surface which in the present case is the NURBS and the PDE surface.

(e) The front is advanced in the field by introducing new points and forming tetrahedras and new faces to complete the grid.

(f) The completed grid may optionally be post-processed.

The above described procedure is applied to an ONERA M6 wing. The wing has a leading edge sweep of 30 degrees, an aspect ratio of 3.8, a taper ratio of 0.56, and symmetrical airfoil sections. The wing has a root chord of 0.67 and a semispan b of 1.0 with a rounded tip. The computational domain is bounded by a rectangular box with boundaries at

$$-6.5 \leq x \leq 11.0, 0.0 \leq y \leq 2.5 and - 6.5 \leq z \leq 6.5$$

Figure 4.13 shows the ONERA M6 wing bounded by the rectangular box. The M6 wing is attached to one of the surface of the box. Triangulations starts from the surface of the box and the M6 surface and proceeds towards the interior of the domain. Figure 4.14 shows the surface triangulation on the actual NURBS M6 surface. This triangulation is obtained by projecting the initial triangulation of the surface on to the actual NURBS surface.

Surfaces obtained from RAPID is also triangulated. In this case the HSCT type configuration is placed in the middle of the rectangular box. Figure 4.15 shows the surface mesh with the rectangular box and the HSCT type configuration. Figure

4.16 shows the surface mesh without the horizontal and vertical tails. In order to simulate the configuration with engines, two tapered rectangular boxes are placed just below the wings, and Fig. 4.17 shows the surface mesh. Figure 4.18 shows the surface mesh with horizontal and vertical tails.

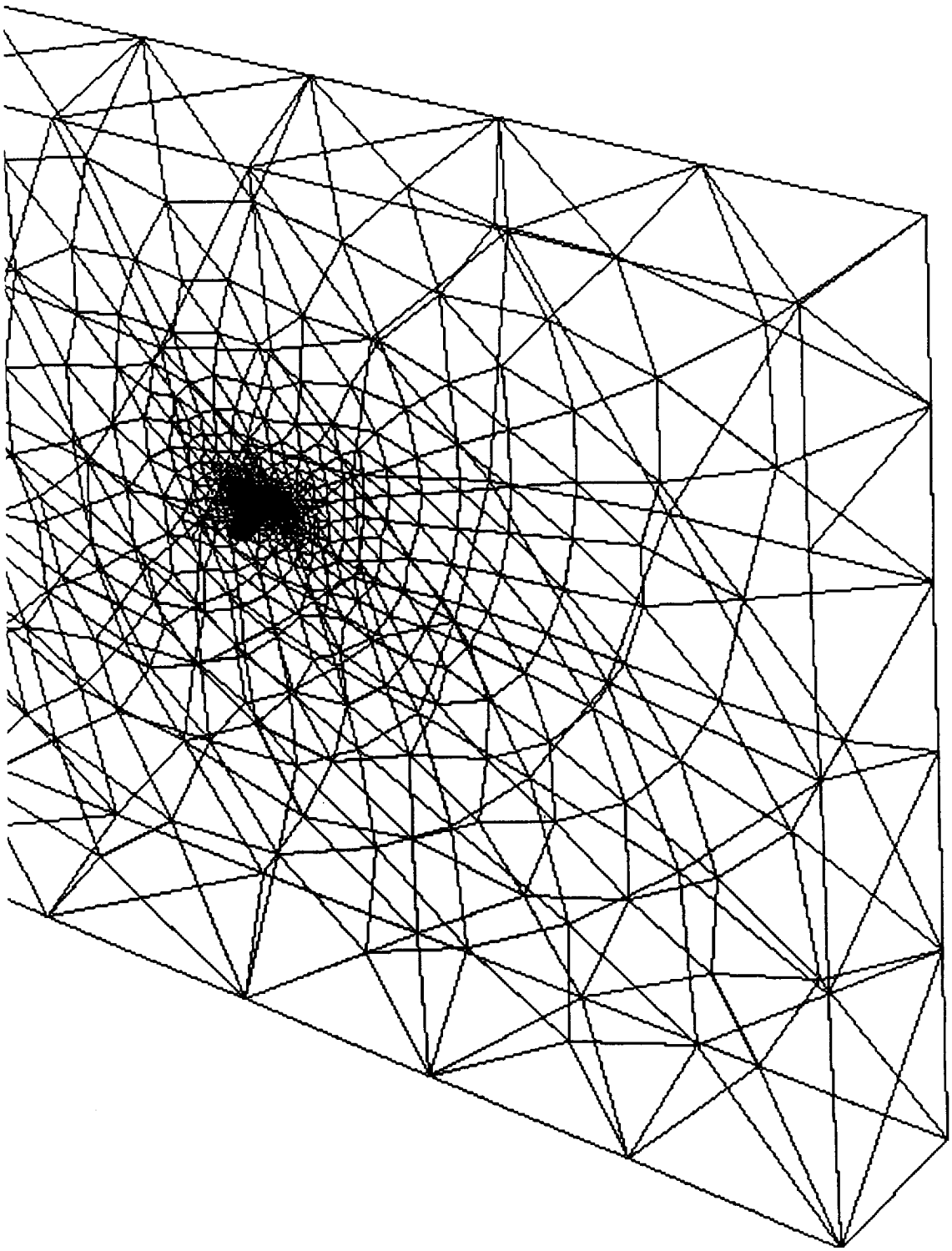


Fig. 4.13 Far Field boundary for the ONERA M6 wing.

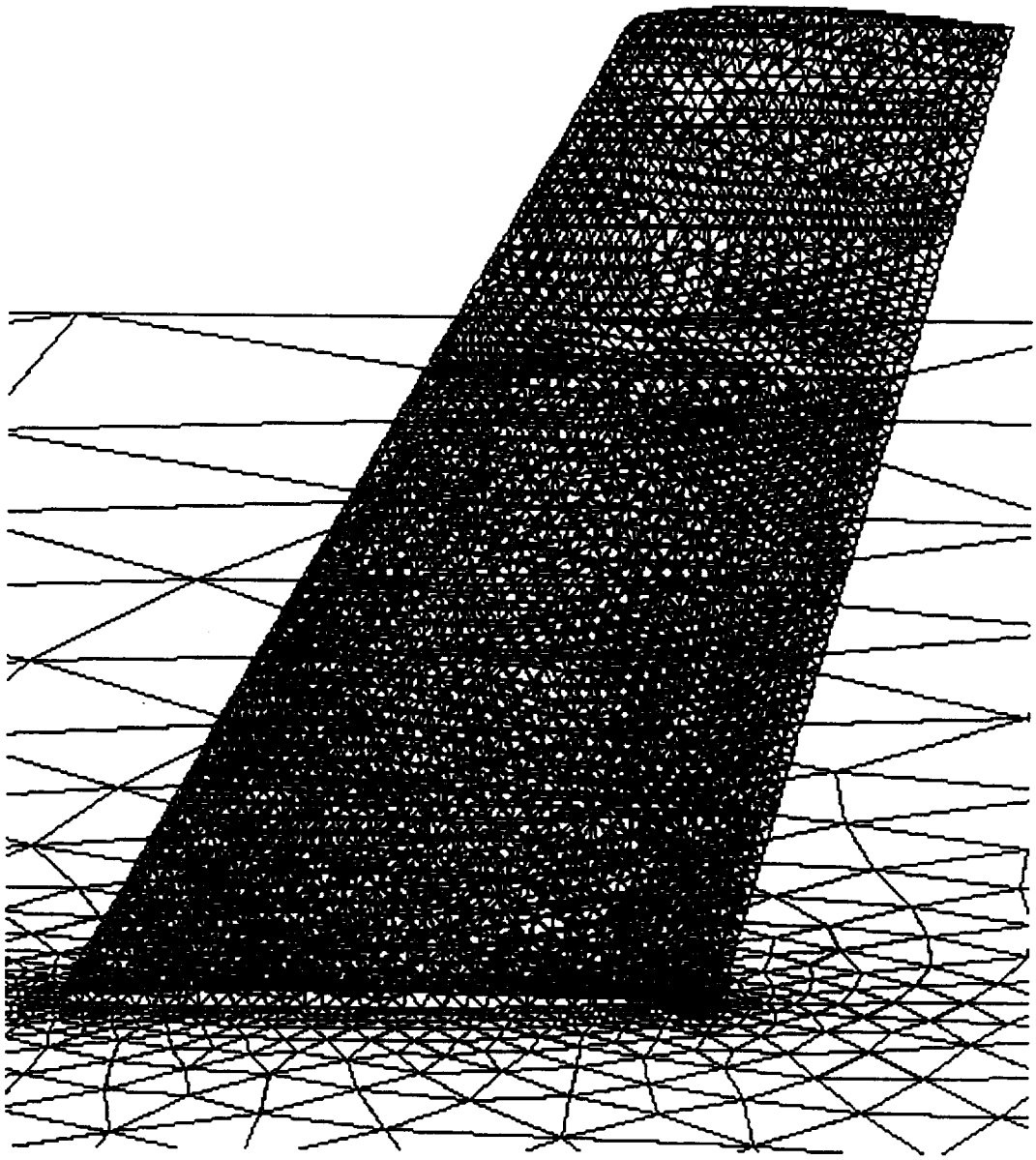


Fig. 4.14 Surface mesh on the ONERA M6 wing.

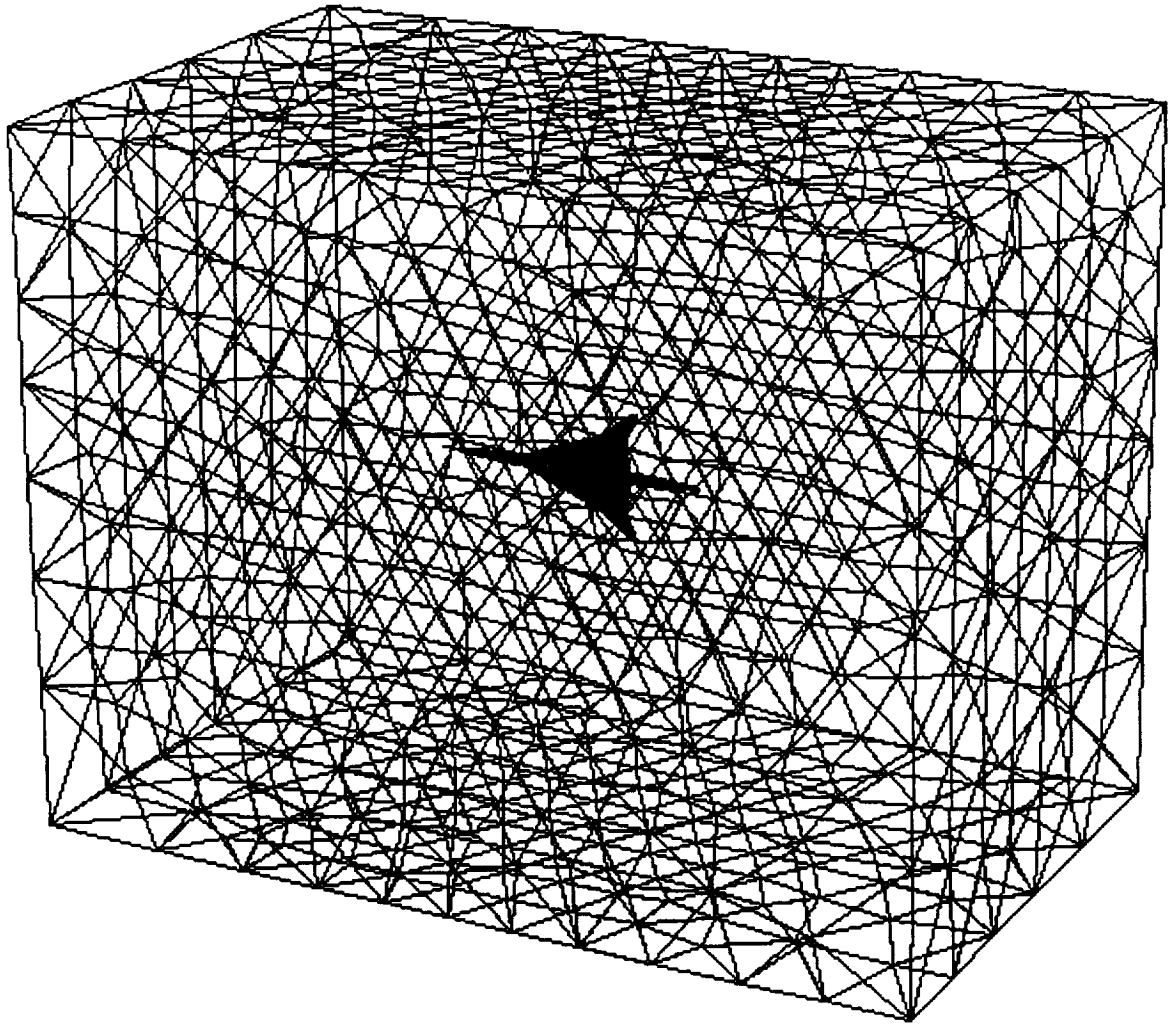


Fig. 4.15 Far field boundary for the HSCT type configuration.

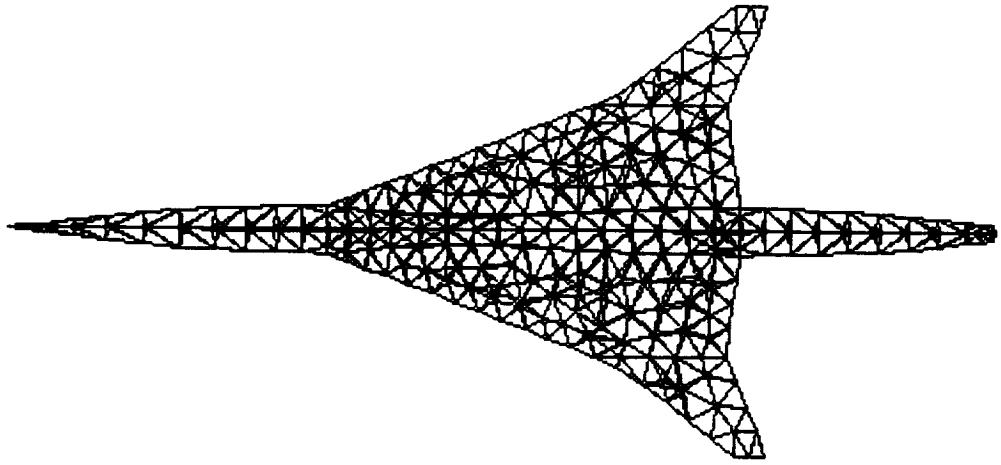


Fig. 4.16 PDE surface without horizontal and vertical tails.

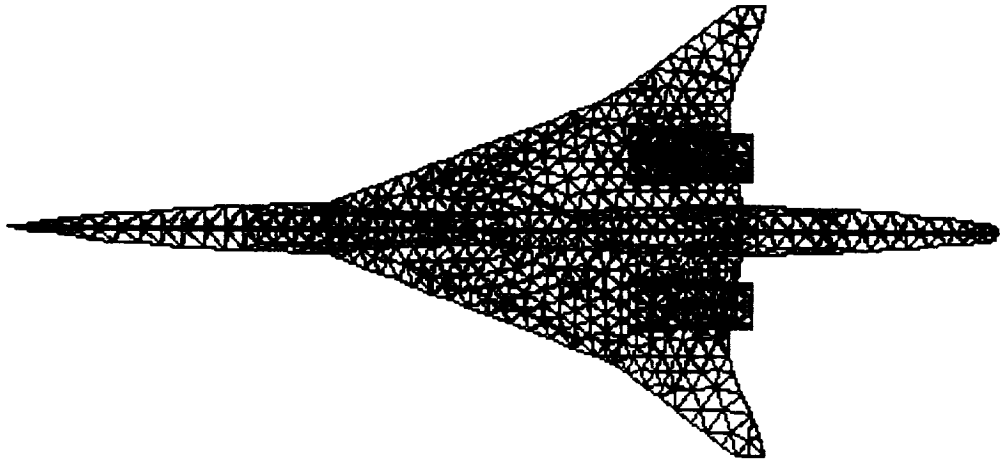


Fig. 4.17 PDE surface with engines mounted below the wing surface.

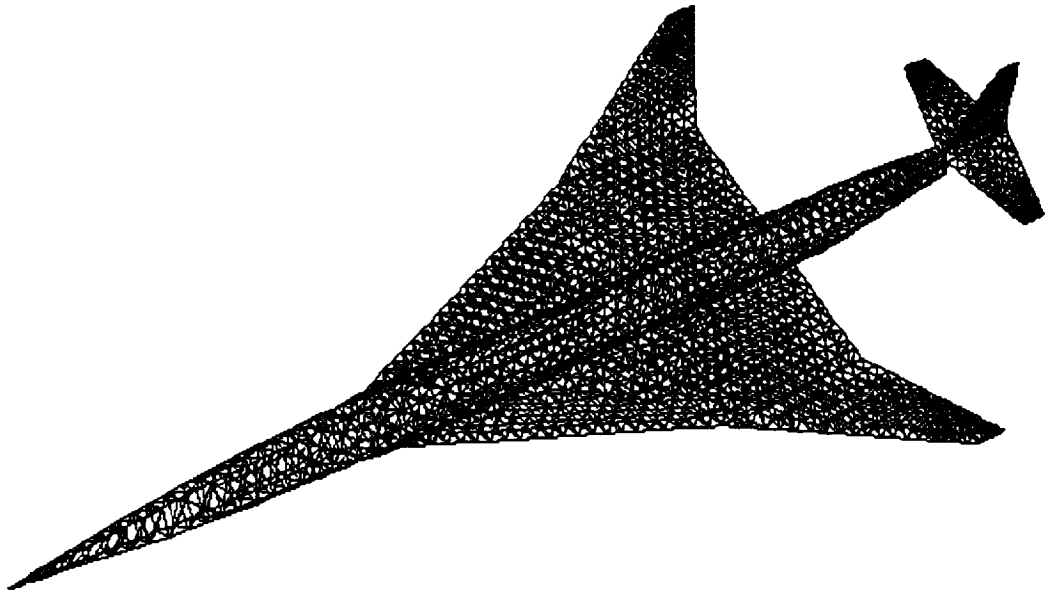


Fig. 4.18 PDE surface with horizontal and vertical tails.

Chapter 5

GOVERNING EQUATIONS FOR FLOW SOLUTIONS

5.1 Unstrurctured Grid Solution

The inviscid flow field is computed on the unstructured grids using USM3D, a three-dimensional upwind flow solver developed at NASA/LaRC [95]. The fluid motion is governed by the time dependent Euler equations for an ideal gas which express the conservation of mass, momentum, and energy for a compressible inviscid nonconducing fluid in the absence of external forces. The equations are given below in integral form, for a bounded domain Ω with a boundary $\partial\Omega$, are expressed as

$$\frac{\partial}{\partial t}\iiint_{\Omega} \mathbf{Q}\mathbf{dV} + \iint_{\partial\Omega} \mathbf{F}(\mathbf{Q}) \cdot \tilde{\mathbf{n}}\mathbf{ds} = \mathbf{0} \tag{5.1}$$

where

$$Q = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e_0 \end{Bmatrix}$$

and

$$\mathbf{F}(\mathbf{Q}) \cdot \tilde{\mathbf{n}} = (\mathbf{V} \cdot \tilde{\mathbf{n}}) \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e_0 + p \end{Bmatrix} + \mathbf{p} \begin{Bmatrix} 0 \\ \tilde{n_x} \\ \tilde{n_y} \\ \tilde{n_z} \\ 0 \end{Bmatrix}$$

The equations are nondimensionalized with reference density ρ_∞ and a speed of sound a_∞. Here $\tilde{n_x}, \tilde{n_y}$ and $\tilde{n_z}$ are the Cartesian components of the exterior surface unit normal $\tilde{n}$ on the boundary $\partial\Omega$. The Cartesian velocity components are u, v, and

w in the x, y and z directions, respectively. The term e_0 is the total energy per unit volume. With the ideal gas assumption, the pressure and total enthalpy can be expressed as

$$p = (\gamma - 1)\left(e_0 - \frac{1}{2}\rho\left(u^2 + v^2 + w^2\right)\right)$$

$$h_0 = \frac{\gamma}{\gamma - 1}\frac{p}{\rho} + \frac{1}{2}\left(u^2 + v^2 + w^2\right)$$

where γ is the ratio of specific heats and is prescribed as 1.4 for air.

The spatial discretization is accomplished with a cell centered finite volume formulation using the flux difference splitting procedure. The solution is advanced in time using a three-stage Runge-Kutta time stepping scheme. Local time stepping and implicit residual smoothing are used to accelerate the convergence of the solution to a steady state.

Boundary Conditions

For the solid boundaries such as the wing and centerplane, the flow tangency condition is imposed by setting the velocities on the boundary faces to their cell center values and then subtracting the component normal to the solid surface. Density and pressure boundary conditions are simply set to the cell-centered value. A condition of zero mass and energy flux through the surface is ensured by setting the left and right states of solid boundary faces equal to the boundary conditions prior to computing the fluxes with Roe's approximate Riemann solver.

Characteristic boundary conditions are applied to the far-field subsonic boundary using the fixed and extrapolated Riemann invariants corresponding to the incoming and outgoing waves. The incoming Riemann invariant is determined from the freestream flow and the outgoing invariant is extrapolated from the interior domain. At an outflow boundary, the two tangential velocity components and the entropy are extrapolated from the interior, while at the inflow boundary they are specified as having far-field values.

The unstructured grid generated around the NURBS M6 wing using the advancing front technique (shown in Fig. 4.13) is used as a test case. The solutions were started from freestream initial conditions with first-order scheme until the L_2-norm (RMS average of all residuals) decreased one order of magnitude, at which time the solver automatically switched to a higher order scheme. Converged solution is obtained for $M_\infty = 0.84$ and $\alpha = 3.06°$. The upper surface pressure contour with contour intervals of $\Delta\left(\frac{p}{p_{infty}}\right) = 0.02$ is shown in Fig. 5.1. The figure clearly shows a double shock wave on the upper surface and is in good agreement with the results obtained by Frink et al. [96].

Converged solutions are also obtained for HSCT type PDE surface shown in Figs. 4.16-18. Converged solutions are obtained for $M_\infty = 0.84$ and $\alpha = 5°$. Figure 5.2 shows the shaded Cp plot. Contours are plotted by taking a cutting plane at the mid section of the configuration. A shock wave is seen at the upper surface of the wing. A total lift of 0.33358 and a drag of 0.04301 were obtained. To simulate this HSCT configuration with engines and to study the performance features, two engines of tappered square cross-section are placed below the wing. The unstructured grid shown in Fig. 4.17 is used for this case. Figure 5.3 shows the shaded Cp plot and for this case. A total lift of 0.313434 and a drag of 0.05932 were obtained.

5.2 Potential Flow Solution

A low-order potential-flow panel code for modeling complex three-dimensional geometries is used to calculate surface pressure variations. The flow field is assumed to be inviscid, irrotational and incompressible. The velocity potential is given by the Laplace's equation:

$$\nabla^2 \Phi = 0. \tag{5.2}$$

The potential at any point P may be evaluated by Green's Theorem which results in

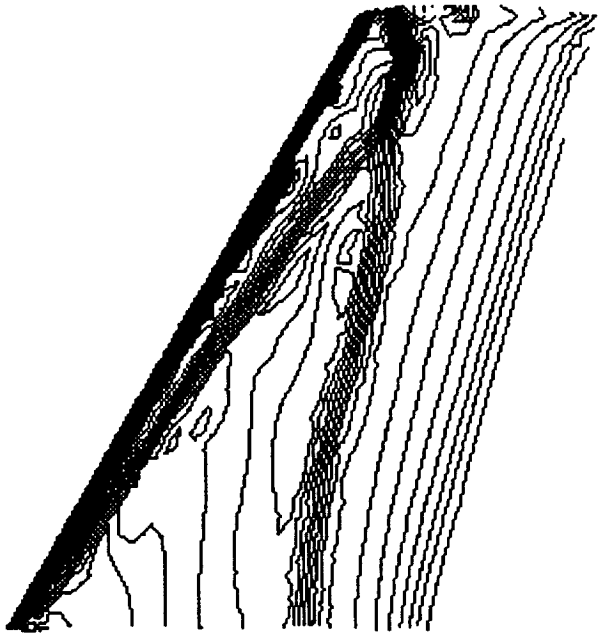


Fig. 5.1 Cp contours over the M6 wing.

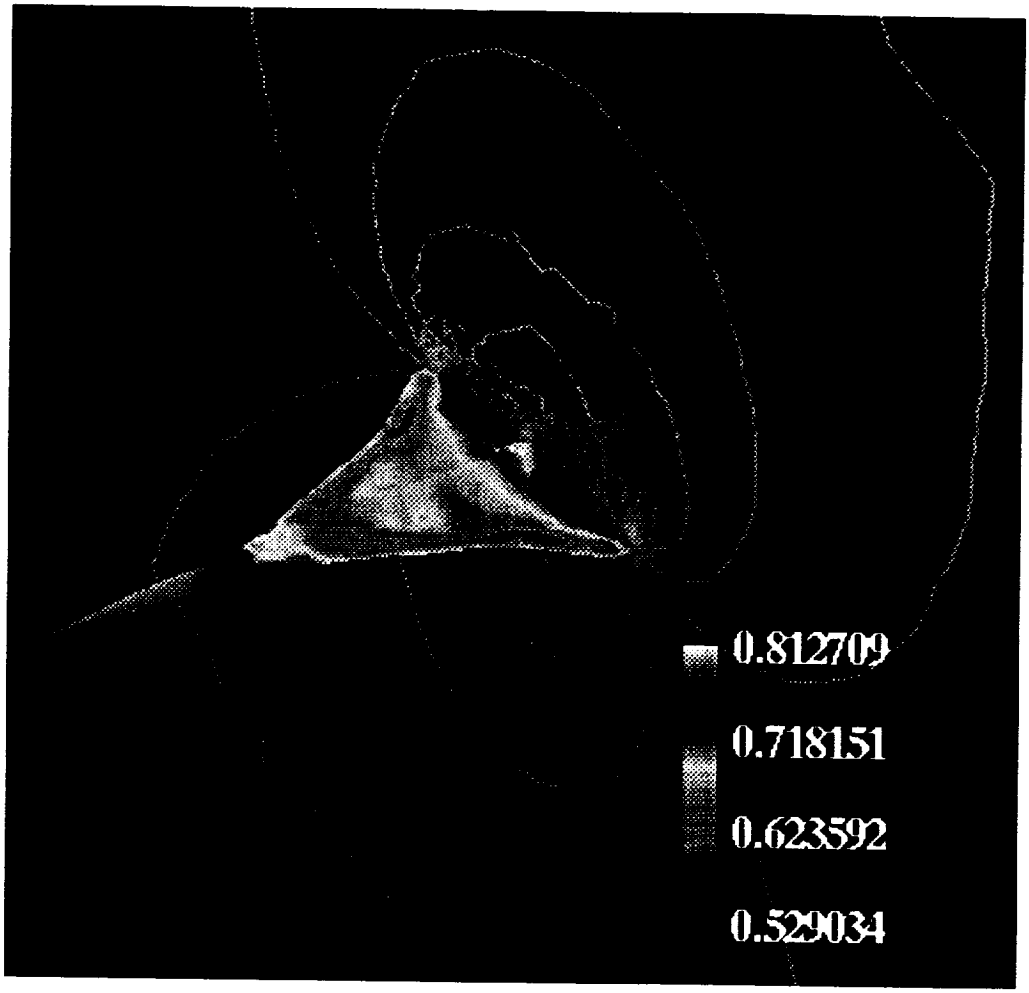


Fig. 5.2 Cp plot for the HSCT configuration without engines.

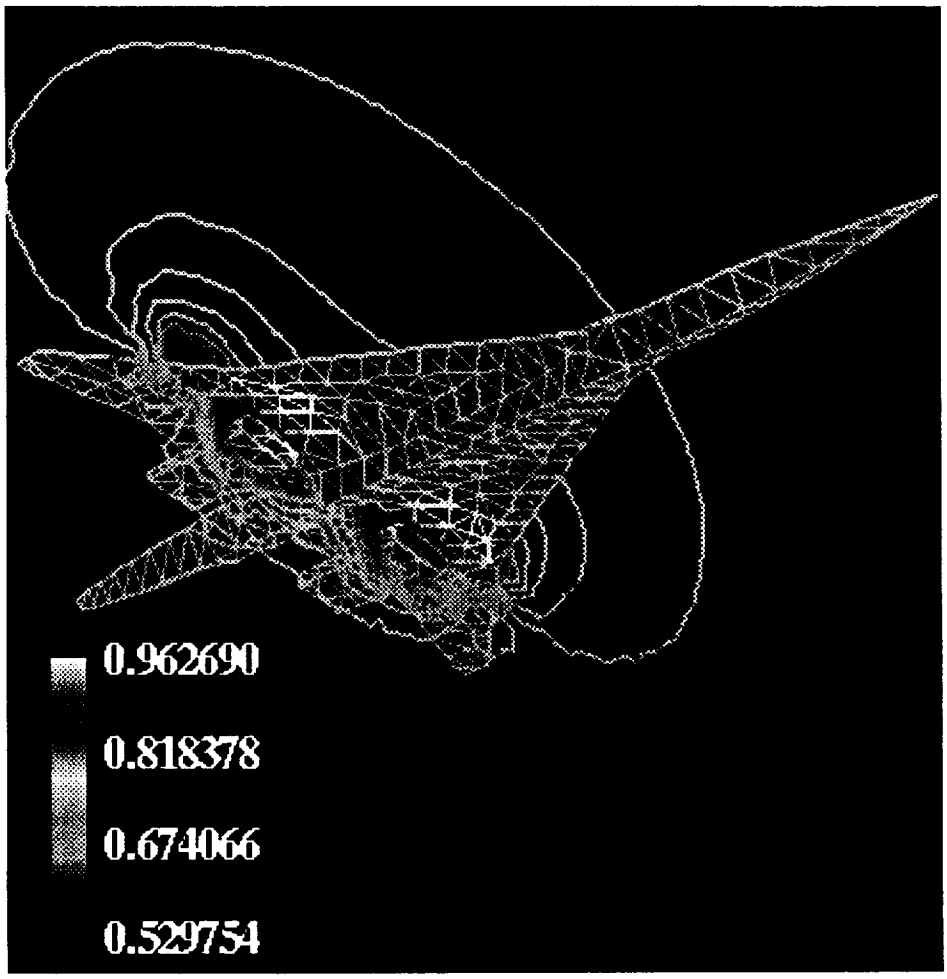


Fig. 5.3 Shaded Cp plot for the HSCT type configuration with engines.

the following integral equation

$$\Phi_P = \frac{1}{4\pi}\iint_{S+W+S_\infty}(\Phi-\Phi_i)\,\bar{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS - \frac{1}{4\pi}\iint_{S+W+S_\infty}\bar{n}\,(\nabla\Phi-\nabla\Phi_i)\,dS \quad (5.3)$$

It is assumed that the wake is thin and there is no entrainment, so the source term for the wake disappears and the jump in normal velocity across the wake is zero. Hence the simplified equation becomes

$$\Phi_P = \frac{1}{4\pi}\iint_{S}(\Phi-\Phi_i)\,\bar{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS - \frac{1}{4\pi}\iint_{S}\bar{n}\cdot(\nabla\Phi-\nabla\Phi_i)\,dS + \frac{1}{4\pi}\iint_{W}(\Phi_U-\Phi_L)\,\bar{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)\left(\frac{1}{\bar{r}}\right)dS + \phi_{\infty p} \quad (5.4)$$

The Dirichlet type boundary condition is used to solve Eq. (5.3). The total potential Φ can be viewed as being made up of an onset potential Φ_∞ and a perturbation potential $\phi = \Phi - \Phi_\infty$. The potential of the fictious flow is set equal to the onset potential, ϕ_∞. With this boundary condition, the singularities on the surface tend to be smaller than if the potential of the fictious flow is set to zero because the singularities only have to provide the perturbation potential instead of the total potential. The general equation for the potential at any point P can be written as

$$\Phi_P = \left[\iint_{S-P}\mu\bar{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS + K\mu_p\right] + \iint_{S}\left(\frac{\sigma}{\bar{r}}\right)dS + \iint_{W}\mu_W\bar{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS + \phi_{\infty p} \quad (5.5)$$

where $K = 0$ if P is not on the surafec, $K = 2\pi$ if P is on a smooth part of the outer surface, and $K = -2\pi$ if P is on a smooth part of the inner surface. If the surface is broken up into panels, Eq. (5.4) can be written in discretized form, breaking the integrals up into surface integrals over each panel. A constant strength source and doublet distribution is assumed over each panel and so the doublet and source strengths are factored out of the integrals. Taking point P to be at the centroid on the inside of one of the panels, the surface integrals over each panel are summed for all panels. For the panel containing point P, the surface integral is zero and only

the $-2\pi\mu_p$ term remains in the bracketed part of Eq. (5.4). For all other panels, the surface integral is used and the $-2\pi\mu_p$ term is zero since the pont P is not on the surface of any other panels. The process is repeated for point P at the centroid of every panel to yield a set of linear simultaneous equations to be solved for the unknown doublet strength on each panel. The surface integrals represent the velocity potential influence coefficients per unit singularity strength for panel K acting on the control point of panel J. Hence Eq. (5.4) becomes

$$\sum_{K=1}^{N_s} (\mu_K C_{JK}) + \sum_{K=1}^{N_s} (\sigma_K B_{JK}) + \sum_{K=1}^{N_w} (\mu_{WL} C_{JL}) = 0 \tag{5.6}$$

where

$$B_{JK} = \iint_K \frac{1}{\bar{r}} dS \tag{5.7}$$

and

$$C_{JK} = \iint_K \bar{n} \cdot \nabla \frac{1}{\bar{r}} dS \tag{5.8}$$

The coefficients C_{JK} and B_{JK} represent the velocity potential influence coefficients per unit singularity strength for panel K acting on the control point of panel J. Equations (5.6) and (5.7) are functions of geometry only and thus can be solved for all panels to form the influence coefficient matrix. Since the source values are known, they may be transferred to the right hand side of the matrix equation. Solutions for Eqs. (5.6) and (5.7) can be found in [97].

As a test case, the PDE surface shown in Fig. 2.12 is considered. Only half of the configuration was modeled in PMARC. The other half of the configuration was generated by reflecting the model across the plane of symmetry. The wing was represented with 300 panels: 15 divisions in the chordwise direction on the upper and lower surfaces of the wing with denser spacing near the leading and trailing edges, and 10 divisions in the spanwise direction with denser spacing near the root and tip

of the wing.The tip of the wing was closed off with a flat tip patch. The fuselage was represented with 320 panels. The wing/fuselage junction was modeled such that wing and fuselage panels matched up exactly. An initial wake was attached to the trailing edge of the wing and to the aft of the fuselage and carried downstream 20 chord length. Three time steps were specified to allow the wake start to roll up. The model was tested at an angle of attack of 4°, and the Cp plot is shown in Fig. 5.4.

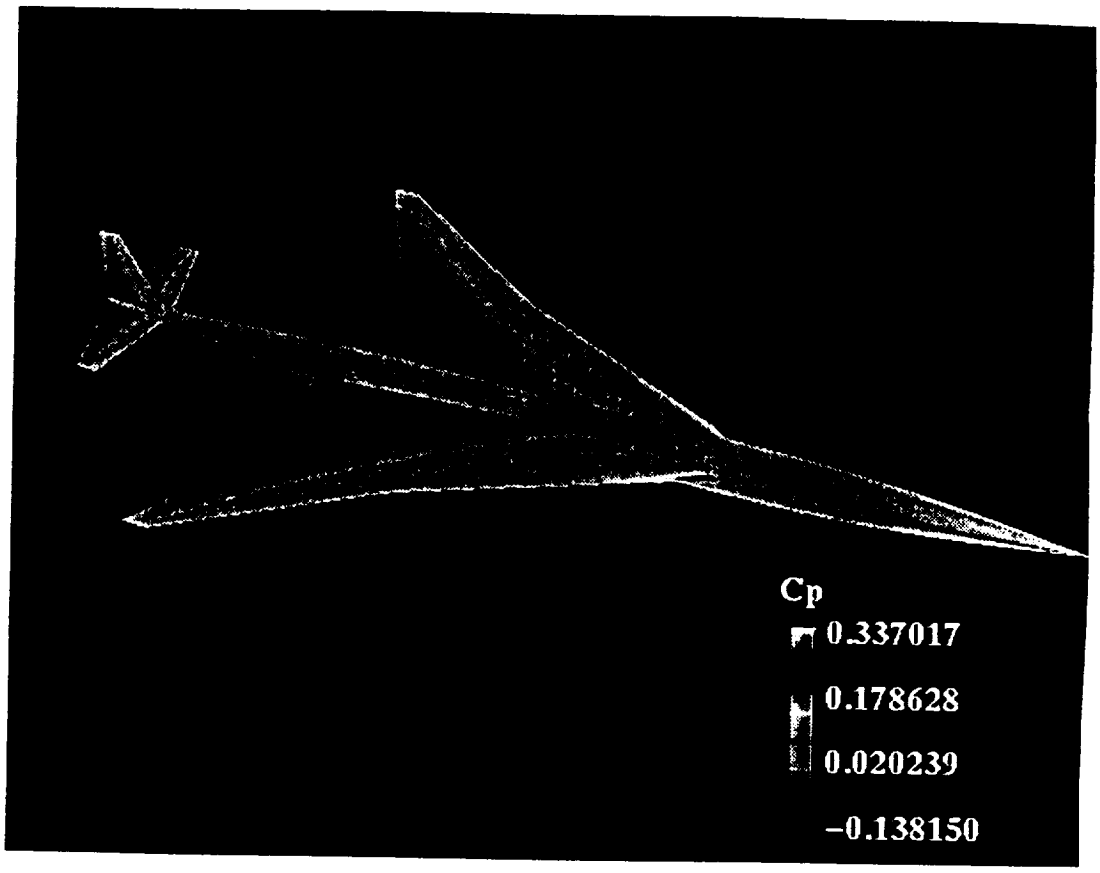


Fig. 5.4 Shaded Cp plot for the potential flow.

Chapter 6

METHOD OF SOLUTION FOR SENSITIVITY EQUATIONS

6.1 Introduction

Computational Fluid Dynamics (CFD) is now routinely applied to simulate flow about aerodynamic configurations. On current supercomputers, these simulations can require several hours per steady-state solution for viscous-compressible flow about airplane configurations. Such large amounts of computational time are acceptable for proof-of-concept studies and selective analysis. With the advent of the next generation of parallel supercomputers, airplane design and optimization using nonlinear CFD should become routine. An essential element in design and optimization is acquiring the sensitivity of functions of CFD solutions with respect to control parameters. For aerodynamic surfaces, the control parameters specify the surface shapes. This affects the surface grid and the field grid which, in turn, affects the flow field solution.

Sensitivity analysis (SA) provides a natural systematic means for both analyzing and predicting the behavior of physical approximations and computational systems or for identifying significant input parameters in a system. The system outputs are assumed to be functionally dependent upon the system inputs. The output changes in response to specified changes in the input; however, everything within the system is normally assumed to be fixed. Changes in the system outputs are related to the changes in the inputs through a sensitivity derivative (SD) matrix, or system

jacobian. The SD matrix may be used to control processes or designs that depend upon the system output.

Procedures for MDO of engineering systems have been addressed by Sobieski and others [97]. Sobieski proposes a unified system SA guided by system derivatives. Aerodynamics plays a central role which is connected to other disciplines. The objective and constraints are provided by the output functions of these several other disciplines. Each single discipline is then to supply not only the output functions for the constrained optimization process, but also the derivatives of all these output functions with respect to its input variables.

Numerous research efforts have examined the issue of efficient computation of SD for CFD. Typical techniques for computing sensitivities include by hand, by use of a symbolic expression differentiator and by approximation via divided differences. Unfortunately, none of these techniques can be used to deliver fast and reliable derivatives in a flexible and timely fashion for large computer codes. Hand coding of derivatives is impractical and symbolic approaches may require as much effort as hand coding. Divided differences may not be accurate and are obtained too slowly.

Automatic differentiation (AD) promises to address the need for a flexible and scalable technology capable of computing derivatives of large codes accurately, irrespective of the complexity of the model. In this study AD has been successfuly applied to obtain sensitivity derivatives required for an MDO procedure. Incremental iterative form, also known as the "delta" or "correction" form, is another successful approach for obtaining the solution state vector from the nonlinear governing flow. References [98], discuss the benefits of using this form to solve the large systems of linear equations needed to obtain SD. This incremental iterative formulation is very flexible and Korivi et al. [99] have demonstrated the use of this strategy to efficiently and accurately calculate quasianalytical sensitivity derivatives for a space-marching 3D Euler code with supersonic flow over a blended wing-body configuration.

6.2 Automatic Differentiation and ADIFOR

Automatic Differentiation (AD) is a collection of computer science techniques which permit one to automatically calculate the derivatives of information generated by a computer program with respect to any parameter intervening in its calculation. AD is essentially an automatic implementation of the chain rule of diffrentiation based on tracking the relationships between dependent and independent variables. Typically, to calculate the derivative of the output of a program with respect to its input, one modifies the original program by insertion of specialized instructions which identify relevant independent and dependent variables. The program is then modified automatically by a preprocessor which enhances it to calculate derivatives. The enhanced program is compiled conventionally, linked with special run-time libraries (if required) and executed to generate not only the original program's dependent variable but also their derivatives with respect to the independent variables.

There are two modes of AD. In the first, the forward mode, the chain rule is evaluated from the input to the output; in this mode, the computational cost increases with the number of outputs. In this mode, the chain rule is evaluated from the output to the input. While it can be much faster than the forward mode, this reverse mode can place enormous demands on computer storage and requires special memory handling. AD is distinct from finite difference or symbolic manipulation techniques. The former, based on perturbations of a program's input, generates approximate derivatives which can be affected by round-off and truncation errors [100]. While an exact technique, the later tends to generate very cumbersome expression for the derivatives.

The tool used in the present study is called ADIFOR [101] (automatic differentiation of Fortran). The tool is jointly developed by Argone National Laboratory and Rice University and it differentiates program written in Fortran 77. That is,

given a Fortran subroutine (or collection of subroutines) that describe a function and an indication of which variables in parameter lists or common blocks corresponds to "independent" and "dependent" variables with respect to differentiation, ADIFOR produces Fortran 77 code that allows the computation of the derivatives of the dependent variables with respect to the independent ones. ADIFOR employs a hybrid of the forward and reverse modes of AD. That is, for each assignment statement, code is generated for computing the partial derivatives of the result with respect to the variables on the right-hand side and then the partials are employed in the forward mode to propogate overall derivatives. The result is a significant decrease in complexity when compared to the forward mode of implementation. The ADIFOR tool produces portable Fortran 77 code and accepts almost all of Fortran 77, in particular, arbitrary calling sequences, nested subroutines, common blocks etc. ADIFOR-generated code can be used in various ways. Instead of simply producing code to compute the Jacobian J, ADIFOR produces code to compute J*S, where the "seed matrix" S is initialized by the user. Therefore, if S is the identity, ADIFOR computes the full Jacobian; whereas if S is just a vector, ADIFOR computes the product of the JACOBIAN by a vector. The running time and storage requirements of the ADIFOR-generated code are roughly proportional to the number of columns of S, so the computation of Jacobian-vector products and compressed Jacobians requires much less time and storage than does the generation of the full Jacobian matrix.

6.3 Theoretical Formulation

An implicit representation of a physical system can be modeled mathematically as

$$F(H, G(H)) = 0 \tag{6.1}$$

where G and H are dependent and independent variables, respectively. The function F can have algebraic, differential, integral or integral- differential characteristics.

The quantities G and H can be either scalar or vector depending on the nature of the physical model. The sensitivity of G with respect to H can be obtained by implicit differentiation of Eq. (5.1)

$$\left\{\frac{\partial F}{\partial H}\right\} = -\left[\frac{\partial F}{\partial G}\right]\left\{\frac{\partial G}{\partial H}\right\} \tag{6.2}$$

The coefficients, $\left\{\frac{\partial F}{\partial H}\right\}$ and $[\frac{\partial F}{\partial G}]$, can be obtained, provided that the solution to Eq.(5.1) is known. Equation (5.2), now a set of algebraic equations, can be easily solved for the sensitivity derivative, $\left\{\frac{\partial G}{\partial H}\right\}$. If $\left\{\frac{\partial F}{\partial H}\right\}$ and $[\frac{\partial F}{\partial G}]$ are not available, a finite difference approach can be adopted. The central difference approximation of $\left\{\frac{\partial G}{\partial H}\right\}$ can be devised as

$$\left\{\frac{\partial G}{\partial H}\right\} \approx \frac{G(H+\Delta H) - G(H-\Delta H)}{2\Delta H} \tag{6.3}$$

where ΔH is a small perturbation of a specified parameter. Although the implementation of the finite difference approach is comparatively easy, it has the disadvantage of being computationally expensive. Also, the choice of ΔH is crucial for accuracy of the derivative. A large values of ΔH may lead to inaccurate derivatives while a small value may result in round-off errors.

6.4 Application of ADIFOR to Potential Flow Code (PMARC)

Application of ADIFOR to advanced flow code has been done by Green et al.[114]. The results have been very encouraging and in the present study ADIFOR is applied to the potential flow code PMARC.

Figure 6.1 indicates an analysis system with input as S and D and output as Q. Both S, D and Q may be scalar, vector, or array quantities, and each may involve one or more variables. The input S and D for the system consist of several

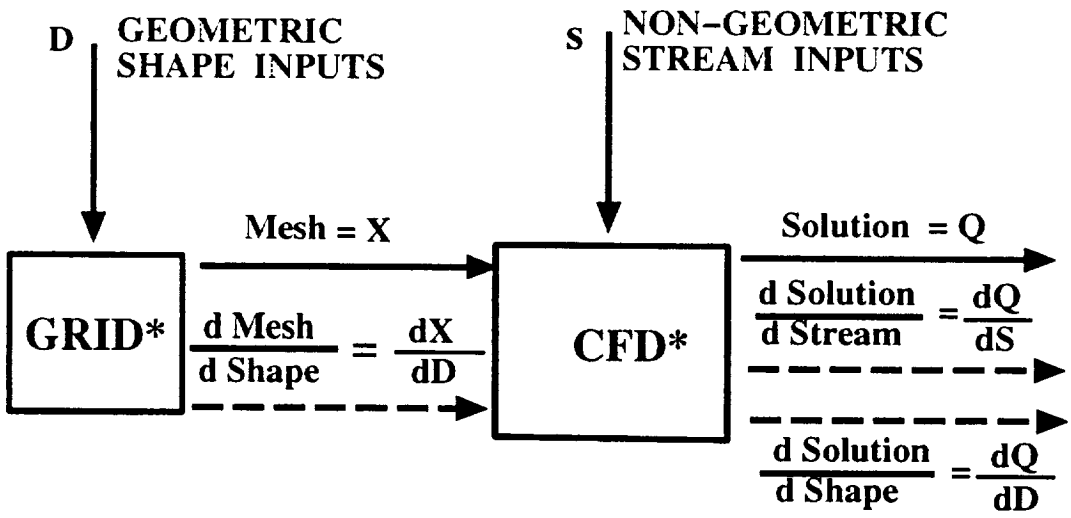


Fig. 6.1 Typical system with ADIFOR applied.

types of variables such as those that specify initial and boundary conditions, material properties, constraints, physical dimensions and approximations, design variables, and numerical solution parameters, etc. Similarly the output Q for the system my consist of local and global solution properties, accuracy measures, and performance indicators etc. When ADIFOR is applied to the above described system, the output consists of a combination of derivatives of the ouput system functions with respect to input system variables. These are indicated as dotted line in the figure. Application of ADIFOR to Fortran codes requires the specification of the independent and dependent variables to be used in forming the SD matrix.

In this study the computations of $C_L, C_D, and C_M$ have been used as the dependent variables. Application of ADIFOR to PMARC was performed in a very simple and straight forward manner. Minor changes to PMARC code was required for ADIFOR processing to be accomplished. The PMARC code was passed on to the ADIFOR as input. ADIFOR differentiated through the entire solution algorithm, the specified dependencies were traced and a new SD code was generated as required. The resulting SD modules were then assembled into a working code and the initial results were generated quickly. The code was run on an SGI Indigo machine and various test cases were examined, and comparisons with direct differentiation procedure have been made.

6.5 Application Of ADIFOR to Grid Generator (RAPID)

Unlike aerodynamic considerations, the grid sensitivity analysis has been used on structural design models for a number of years. In this context, grid sensitivity can be thought of as perturbation of structural loads, such as displacement or natural frequency, with respect to finite element grid point locations [102]. Two basic approaches have been cited for grid sensitivity derivatives. The first approach, known

as implicit differentiation, is based on implicit differentiation of discretized finite element system. The other, which is based on the variation of continuum equations, is known as variational or material derivative approach. Gradient based techniques applied to aerodynamic configurations optimization require the determination of grid sensitivity ($\frac{\partial \mathbf{X}_{vol}}{\partial P} = \frac{\partial \mathbf{X}_{vol}}{\partial \mathbf{X}_{surf}} \frac{\partial \mathbf{X}_{surf}}{\partial P}$). In the past, in order to evaluate such derivatives, each expression would have to be differentiated and chain ruled through out the mathematical system, either by hand or with the aid of a computer-aided algebraic manipulation system. The simplest way to obtain grid sensitivity is to vary the control parameters, one at a time, and finite difference the results. This, however, is proven to be computationally inefficient compared to analytical or semi-analytical differentiation of the grid equations. Also, the proper choice of a step size is not trivial and an improper choice might result in round-off error accumulation. The finite difference approach should only be used as the last resort when the extreme complexity of the grid equations dictates no other alternatives. For a less complicated grid equations, a semi-analytical approach would be more appropriate. The semi-analytical approach consists of analytical differentiation of the original function with respect to an intermediate function, the derivative of which is then evaluated numerically. It combines the efficiency of the analytical approach with the ease of implementation of the finite difference approach.

The analytical approach to the grid sensitivity problem is evaluation of the grid sensitivity coefficient by direct analytical differentiation of the grid equation. For most cases, the grid equation is not directly differentiable, although there are schemes that such differentiations are feasible. The algebraic grid generation scheme, such as *Two-Boundary Grid Generation* (TBGG), was successfully differentiated by analytical methods by Sadrehaghighi [108] and very accurate results were provided. The analytical approach has the advantage of being exact, thus, avoids the round-off errors associated with numerical approaches. Due to the time consuming nature and

tedious process of analytical approach, in this study grid sensitivity was obtained by applying ADIFOR software to the grid generator RAPID.

Geometric inputs to the RAPID software consists of wing planform and wing section definitions, fuselage defining parameters and grid spacing control parameters. These parameters are identified as independent variables in ADIFOR, and the output surface defining grid coordinates (x,y,z) as the dependent variables. ADIFOR successfuly differentiates the entire RAPID software by identifying the various dependencies and an enhanced code is generated. The enhanced ADIFOR grid generator $RAPID_{AD}$ not only generates the grid, but also generates the derivatives or grid sensitivities to the geometric parameters.

6.6 Optimization Problem

An objective of a multidisciplinary optimization of a vehicle design is to extremize a payoff function combining dependent parameters from several disciplines. Most optimization techniques require the sensitivity of the payoff function with respect to free parameters of the system. For a fixed grid and solution conditions, the only free parameters are the surface design parameters. Therefore, the sensitivity of the payoff function with respect to design parameters are needed.

The optimization problem is based on the method of *feasible directions* [104,105] and the generalized reduced gradient method. This method has the advantage of progressing rapidly to a near-optimum design with only gradient information of the objective and constrained functions required. The problem can be defined as finding the vector of design parameters $\mathbf{X}_D$, which will minimize the objective function $f(\mathbf{X}_D)$ subjected to constraints

$$g_j(\mathbf{X}_D) \leq 0 \qquad j = 1, m \tag{6.4}$$

and

$$\mathbf{X}_D^l \leq \mathbf{X}_D \leq \mathbf{X}_D^u \tag{6.5}$$

where superscripts denote the upper and lower bounds for each design parameter. The optimization process proceeds iteratively as

$$\mathbf{X}_D^n = \mathbf{X}_D^{n-1} + \gamma \bar{\mathbf{S}}^n \tag{6.6}$$

where n is the iteration number, $\bar{\mathbf{S}}^n$ the vector of search direction, and γ a scalar move parameter. The first step is to determine a feasible search direction $\bar{\mathbf{S}}^n$, and then perform a one-dimensional search in this direction to reduce the objective function as much as possible, subjected to the constraints.

The present optimization strategy is based on maximizing the lift coefficient, C_L, in response to surface perturbation, subject to pre-determined design constraints. Upper and lower bounds set for each design parameter and the sensitivity derivatives of the objective function, $\frac{\partial C_L}{\partial \mathbf{X}_D}$, and the constraint, $\frac{\partial C_D}{\partial \mathbf{X}_D}$, are obtained as previously described. Throughout the analysis, the drag coefficient, C_D, is to be no greater than the value of the initial design. The strategy, illustrated in Fig. 6.2, requires that the grid and grid sensitivity derivatives be provided dynamically during the automated optimization process.

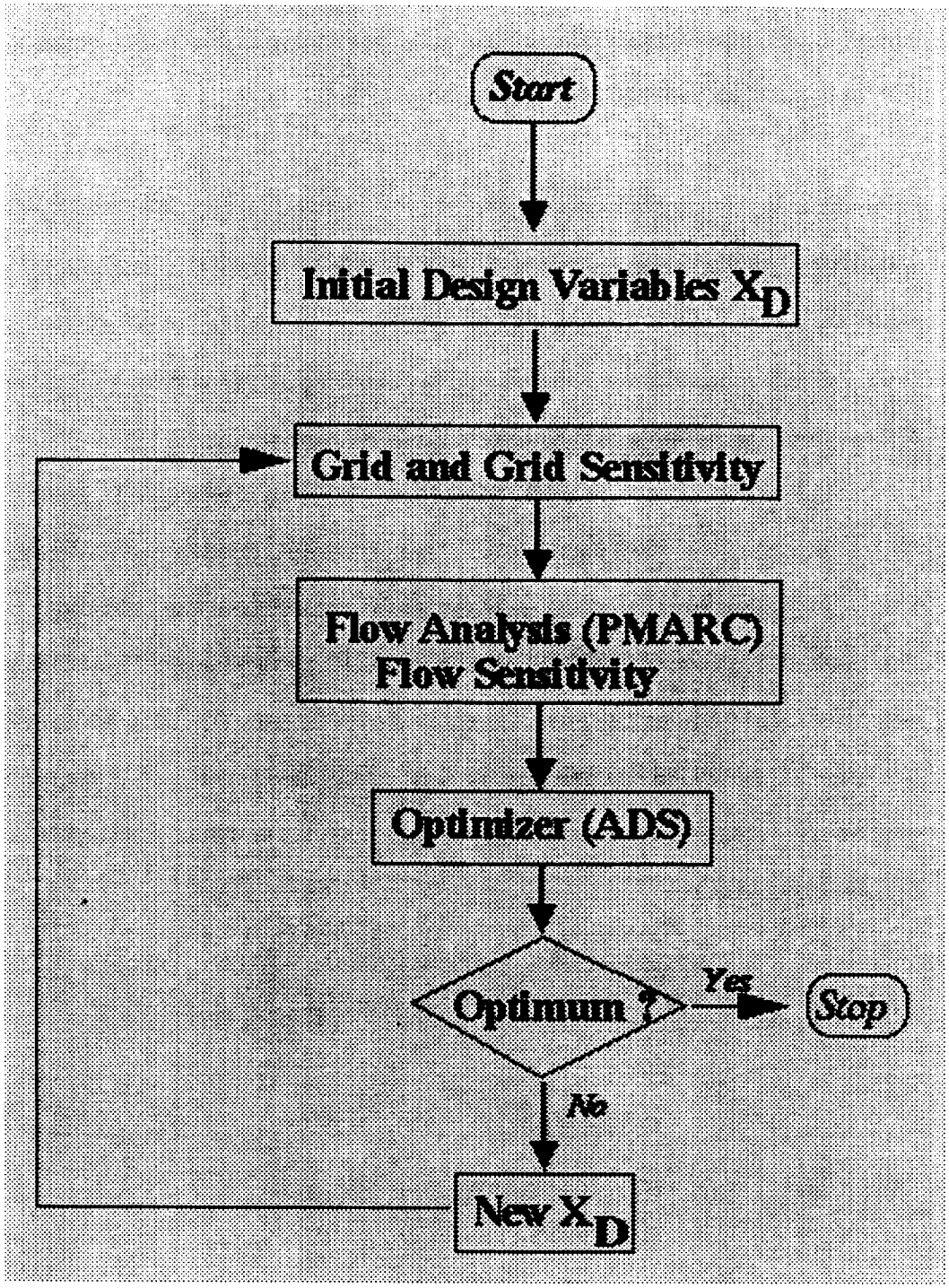


Fig. 6.2 Optimization strategy loop.

Chapter 7

RESULTS AND DISCUSSION

Two test cases are considered to demonstrate the feasibility of current procedure. For each case, the grid and flow sensitivity coefficients of the field have been obtained. The sensitivities of the total forces (i.e., Lift and Drag coefficients) are tabulated for optimization purposes. The first test case, a symmetrical generic airplane with 14 surface defining parameters (Fig. 7.1), has been used mainly to exhibit the accuracy of grid sensitivity coefficients with those obtained using finite difference approach before proceeding to a realistic configuration. The second test case, a HSCT type configuration (Fig. 7.2), has been used to extend the analysis to do a three dimensional optimization. An optimization module has been integrated into the overall procedure to optimize the geometry using the resultant sensitivity coefficients. The improved design is used for the Euler study where an Euler type two block volume grid is constructed using GRIDGEN software and solutions obtained using the TLNS3D code. The CSCMDO[106] software is also used to transform the grid from the original geometry to the new optimized geometry.

7.1 Grid Sensitivity

Grids obtained from RAPID software, shown in Fig. (3.1), is considered for grid sensitivity analysis. Grid sensitivity study was performed on the HSCT type configuration shown in Fig. (7.2) with fourteen surface defining design variables. The surface grid sensitivity with respect to the vector of design parameters, X_D, is obtained from the ADIFOR differentiated code RAPID.

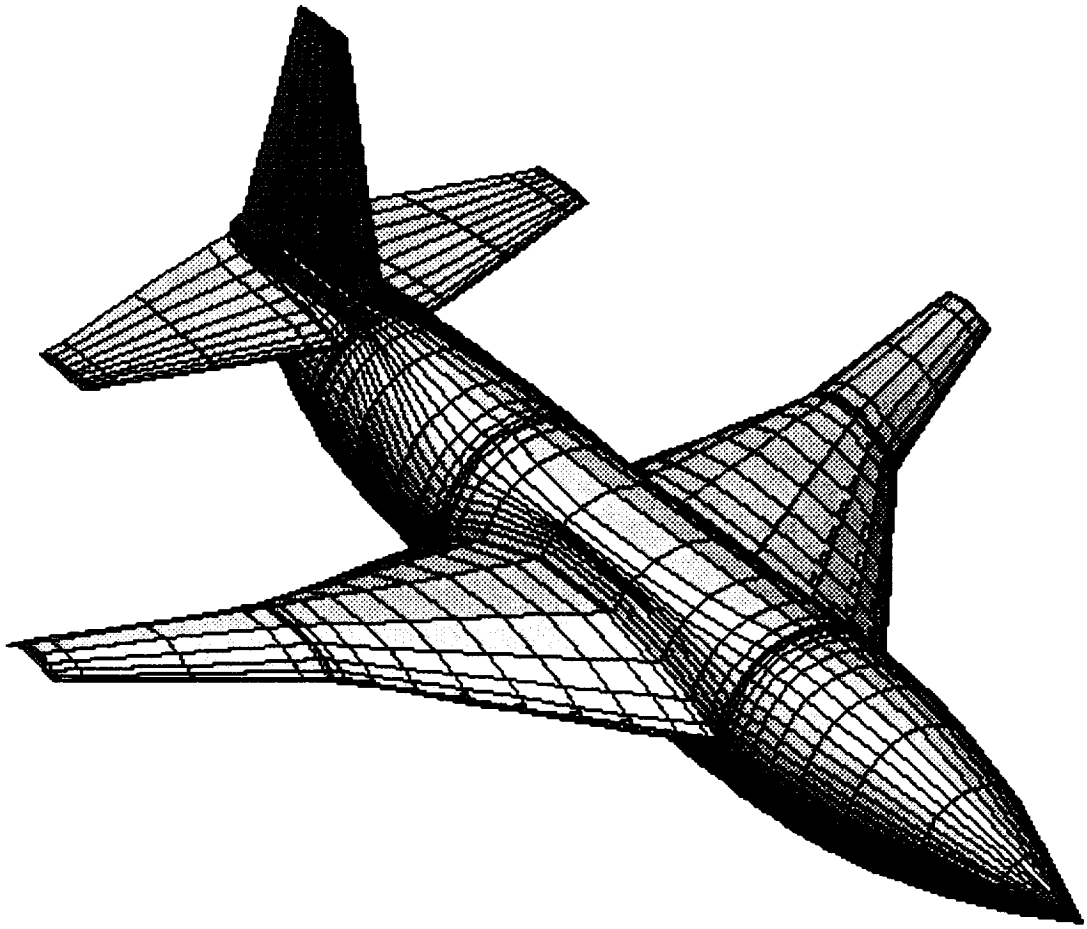


Fig. 7.1 Generic airplane for case 1.

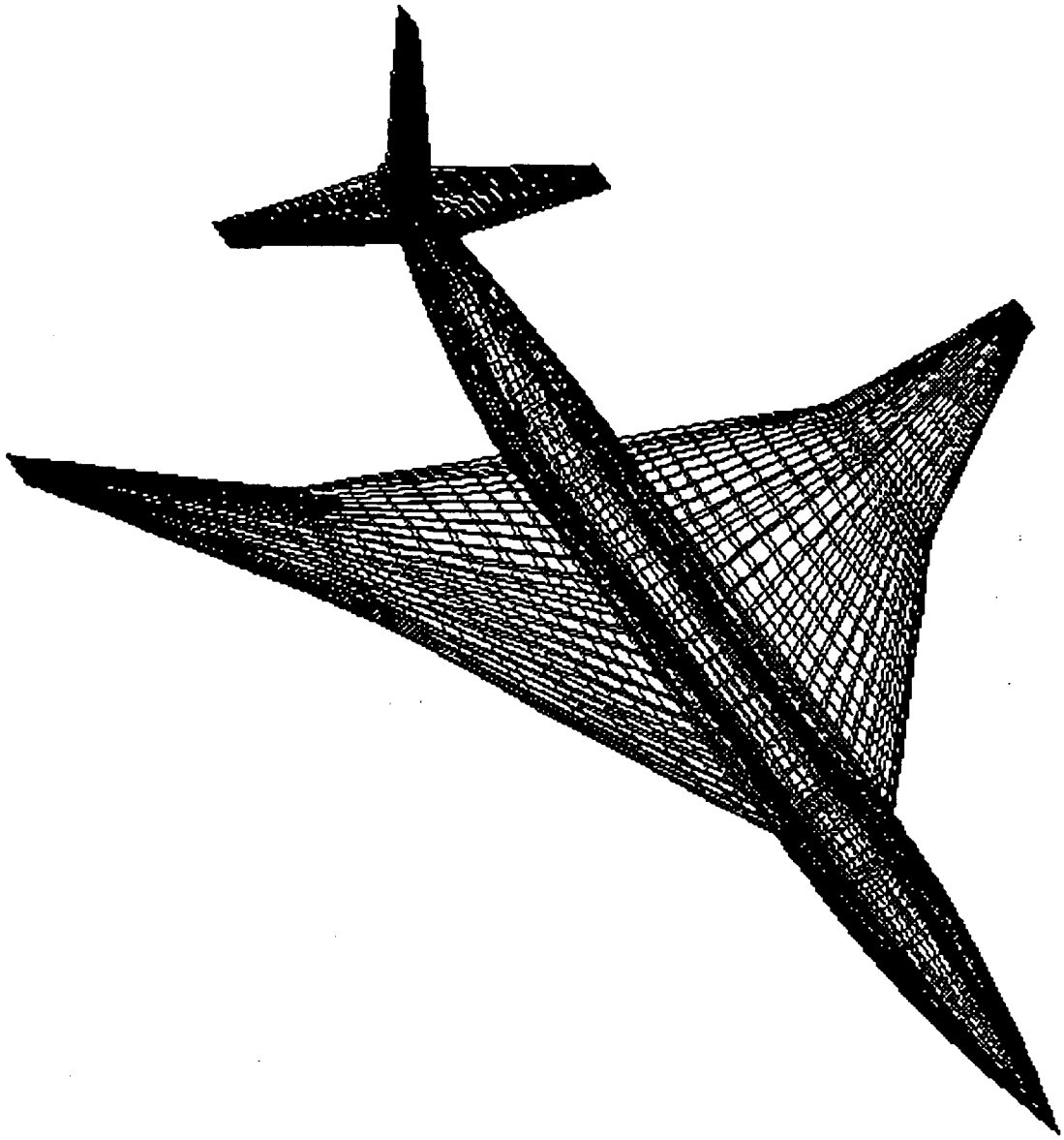


Fig. 7.2 Symmetrical HSCT type configuration.

Figure 7.3 shows the x-coordinate sensitivity with respect to chamber. The highest contour levels are, understandably, located at the point of maximum chamber. Since the wing sections are defined as a NACA four-digit wing sections, this is positioned at about 0.3 of the chord length from the leading edge[107]. The positive and negative contour levels correspond to the upper and lower surfaces. The sensitivity levels decrease when moving away from the location of maximum chamber.

Typical CFD calculations are performed on a computational mesh that is "body-oriented". Changes in the geometric shape results in the movement of grid points throughout the entire mesh. The benchmark for comparison of these grid sensitivity terms is by performing finite difference. If forward difference approximations are selected, for example, the mesh generation code is used to produce one additional perturbed grid for a slightly perturbed value of geometric shape design variable which in this case is the camber. Finite difference at each of the grid coordinate is calculated and the result is shown in Fig. 7.4. It is seen that the results obtained by ADIFOR and the finite difference are in very good agreement, thus confirming the accurate results obtained by ADIFOR.

Having confirmed the results obtained from ADIFOR, and to further evaluate the results, next the design variable chord is choosen as the independent parameter. Figure 7.5 shows the x-coordinate sensitivity with respect to the chord. The concentration of contours are near the leading edge, in the x-direction of the wing fuselage intersection thus confirming the maximum effect at that point. The increase in chord moves the tip of the wing fuselage airfoil section towards the nose of the airplane configuration while the trailing edge is kept fixed. Figure 7.6

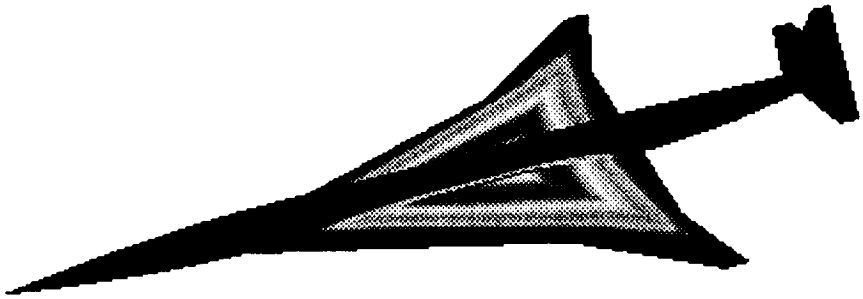


Fig. 7.3 X–coordinate sensitivity with respect to camber.

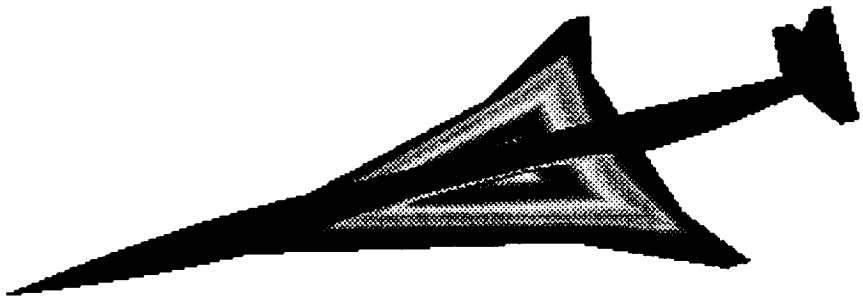


Fig. 7.4 Finite difference X–coordinate sensitivity with respect to camber.

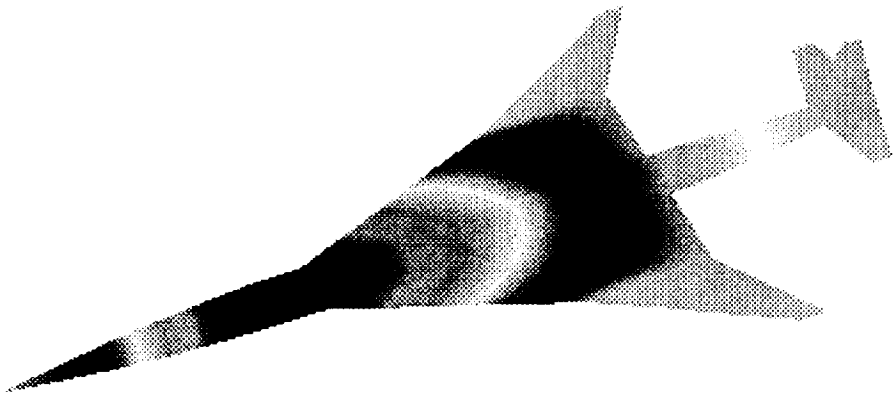


Fig. 7.5 X–coordinate sensitivity with respect to chord.

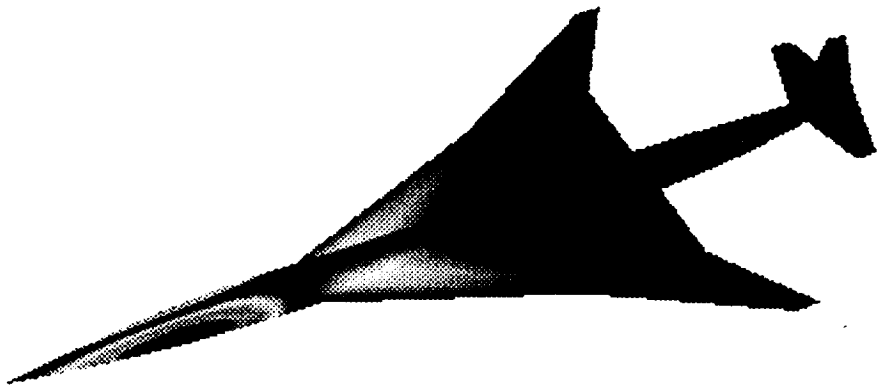


Fig. 7.6 Y–coordinate sensitivity with respect to chord.

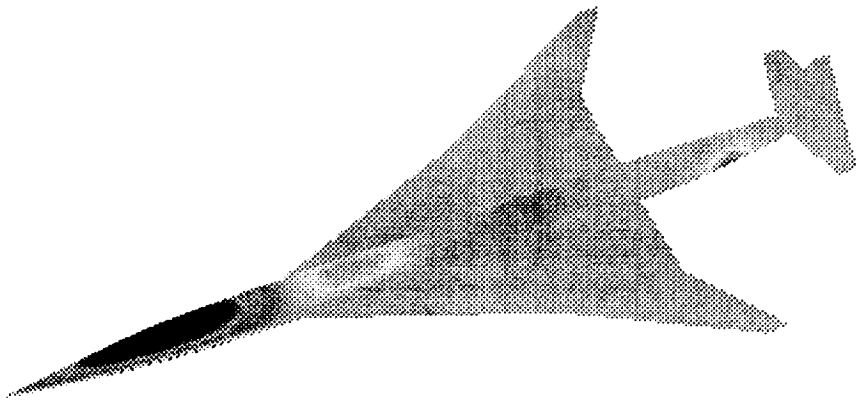


Fig. 7.7 Z–coordinate sensitivity with respect to chord.

shows the y-coordinate sensitivity with respect to chord. Unlike the x coordinate sensitivity the contour levels this time is concentrated in the y-direction indicating a maximum change. Figure 7.7 shows the z coordinate sensitivity with respect to chord.

7.2 Flow Sensitivity

When AD is applied directly to the potential flow code PMARC, the resulting AD-enhanced code calculates the required sensitivity derivatives through an iterative process. The ADIFOR procedure generates a new version of the potential flow code that has the capability to calculate the derivatives of lift, drag, and pitching moment with respect to a wide variety of different types of input parameters (including parameters related to the geometric design).

Both geometric and non-geometric design variables are considered to evaluate the accuracy of ADIFOR enhanced PMARC ($PMARC_{AD}$). Angle of attack (α) is considered as the non-geometric design variable and both sensitivities of lift and drag are computed. For the geometric design variable, wing thickness and fuselage diameter are considered. The values are compared with the finite difference results and are tabulated in Table 7.1. It is seen that the values obtained by ADIFOR are in good agreement with the finite difference values, thus confirming the successful differentiation of the PMARC code.

7.3 Optimization Problem

An objective of multidisciplinary optimization of a vehicle design is to extremize a payoff function combining dependent parameters from several disciplines. Most optimization techniques require the sensitivity of the payoff function with respect to free parameters of the system. For a fixed grid and solution conditions,

Table 7.1. Comparison Of ADIFOR results with finite difference for geometric and nongeometric design variables.

	ADIFOR		Finite Diff.	
	$\frac{\delta C_L}{P}$	$\frac{\delta C_D}{\delta P}$	$\frac{\delta C_L}{\delta P}$	$\frac{\delta C_D}{\delta P}$
Angle Of Attack	0.19981	-0.21192	0.19974	-0.2127
Wing Thickness	0.171446	-0.07568	0.17146	-0.07547
Fuselage Diameter	-2.29020	-0.09421	-2.29031	-0.09420

the only free parameters are the surface design parameters. Therefore, the sensitivity of the payoff function with respect to design parameters are needed.

The present optimization strategy is based on maximizing the lift coefficient, C_L, in response to surface perturbation, subject to pre-determined design constraints. Upper and lower bounds are set for each design parameter and the sensitivity derivatives of the objective function, $\frac{\partial C_L}{\partial X_D}$, and the constraint, $\frac{\partial C_D}{\partial X_D}$, are obtained as previously described. Throughout the analysis, the drag coefficient, C_D , is to be no greater than the value of the initial design. The strategy, illustrated in Fig. 6.2, requires that the grid and grid sensitivity derivatives be provided dynamically during the automated optimization process.

Optimization of the HSCT type configuration shown in Fig. 7.2 was carried out on SGI machine with memory capacity of 512 MB. Sixteen design variables were selected for the optimization process. A total of twelve design optimization cycles were performed and each iteration took approximately 7.5 min of cpu time. It was noted that the lift which was initially 0.01712 became 0.0748. The initial and final shapes with shaded Cp plots are shown in Figs. 7.8 and 7.9. The comparison of the two shapes, before and after the optimization cycle is shown in Fig. 7.10. A considerable increase in the length of the inner and outer wings with an increase in the wing planform area is seen. A decrease in the wing and fuselage thickness with an increase in camber is also noted.

7.4 Euler Flow Solutions

To verify the results obtained from the optimizer, it was suggested to perform Euler calculations over the initial and final shapes of the HSCT configuration. A semidiscrete, cell-centered finite volume algorithm TLNS3D , based on a Runge-Kutta time-stepping scheme, is used to obtain the Euler solutions around the HSCT

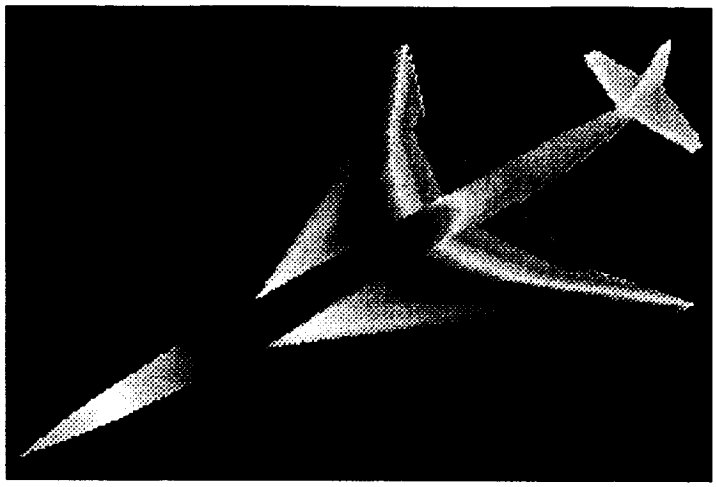


Fig. 7.8 Cp over the initial HSCT configuration.

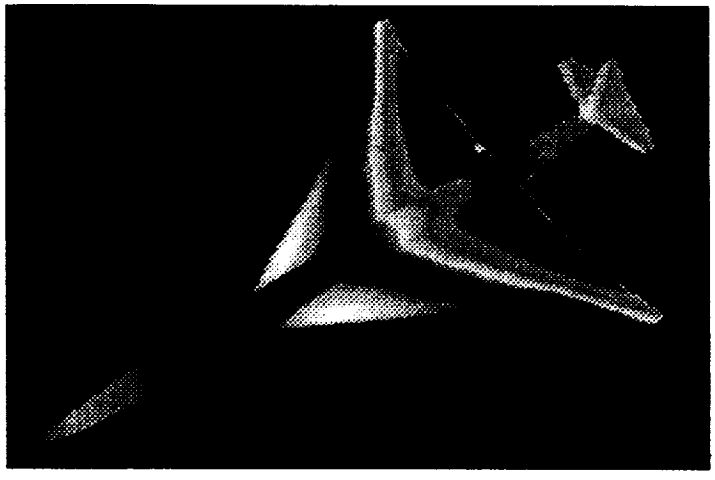


Fig. 7.9 Cp over the final HSCT configuration.

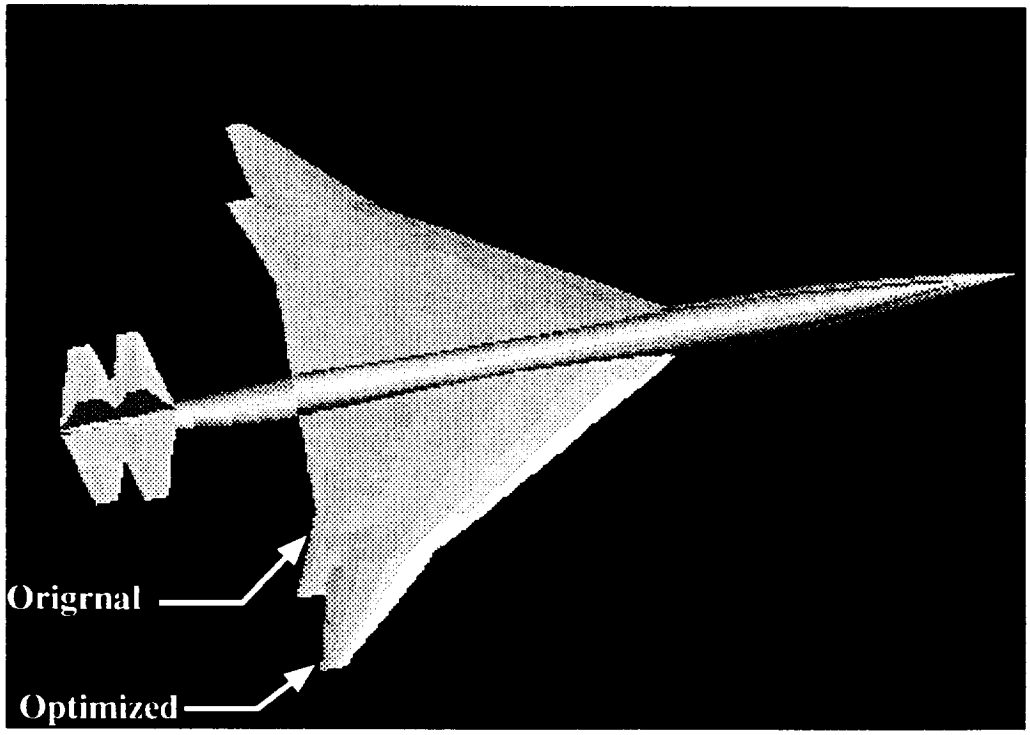


Fig. 7.10 Comparison of initial and final shapes.

type configuration. The efficiency of the numerical scheme is greatly enhanced by taking advantage of the multigrid acceleration technique. A two block, C-O mesh with 142x82x42 grid, shown in Fig. 4.12, is used to obtain converged solution at Mach number of 2.4. Figure 7.11 shows the Cp plot over the original geometry. A lift of 0.01748 was obtained, and it compared very well with the potential flow case.

Next a volume grid was generated over the optimized HSCT surface shown in Fig. 7.9. In this case the CSCMDO software was used to interpolate points from the volume grid generated over the original geometry, thus saving time by not generating the grid from scratch. Figure 7.12 shows the shaded Cp plot over the optimized configuration. Figures 7.13 and 7.14 show the line plot of Cp at the crank of the wing. It is noted that the distribution of pressure is very smooth and well behaved over the surface of the optimized configuration, thus confirming the trend of the results obtained from the optimizer.

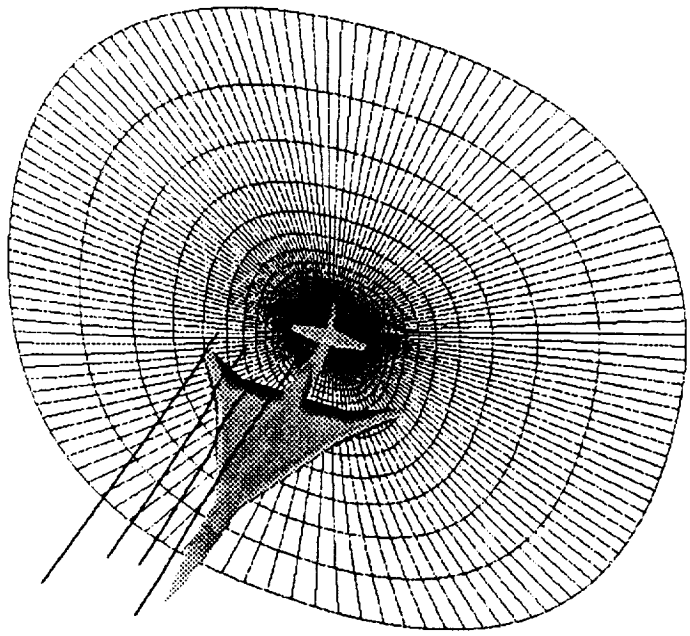


Fig. 7.11 Euler flow solution on the original configuration.

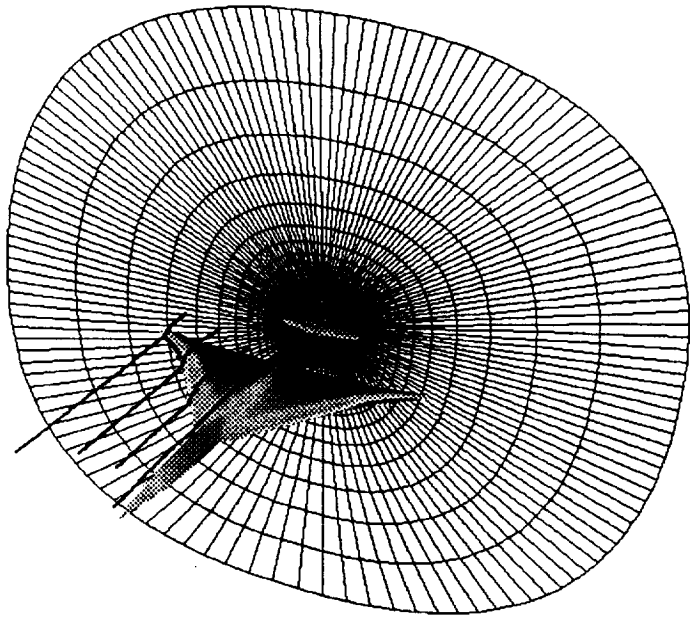


Fig. 7.12 Euler flow on the optimized configuration derived from potential flow optimization.

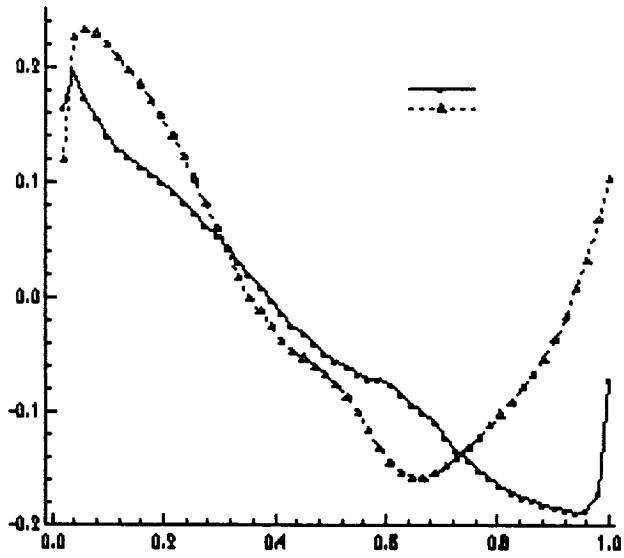


Fig. 7.13 Cp plot at the crank for the original configuration.

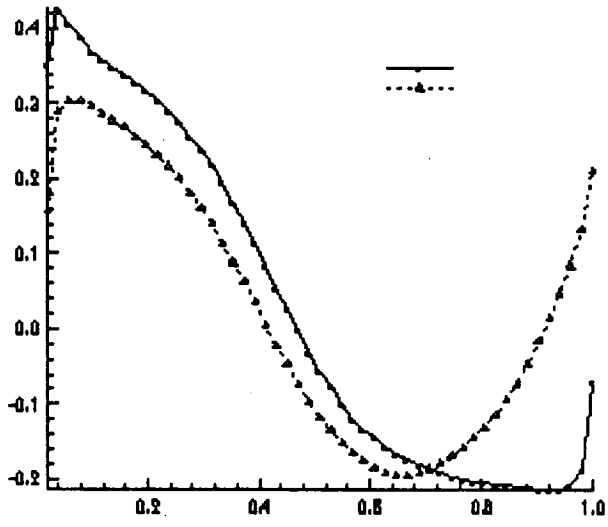


Fig. 7.14 Cp plot at the crank for the optimized configuration.

Chapter 8

CONCLUSIONS AND RECOMMENDATIONS

An algorithm is developed to define the surfaces of aerodynamic configurations for design optimization. The two schemes investigated are Non-Uniform Rational B-Splines (NURBS) and Partial Differential Equation (PDE). NURBS parametrization defines the surface by a set of ordered control points. These control points act as a set of design parameters which is used in an optimization process. The PDE technique offers a unique type of parametrization where a surface is defined by a fourth order partial differential equation. This procedure generates a blend surface between two sets of curves. The design parameters in this case is the constants used in the definition of the two curves.

The PDE technique is used towards the parametrization of a HSCT type configuration. Inclusive in this definition are surface grids, volume grids, and grid sensitivity. The design variables are incorporated into the boundary conditions, and the solution is expressed as a Fourier series. The fuselage has circular cross section, and the radius is an algebraic function of four design parameters and an independent computational variable. Volume grids are obtained through an application of the Control Point Form method.

A graphic interface software is developed to represent the PDE surface. The software has the capability to dynamically change the surface with the change in input design variables. Various options and features are provided to enhance the quality of the software which gives a competitive outlook.

Grid sensitivity with respect to surface design parameters and aerodynamic

sensitivity coefficients based on potential flow is obtained using an Automatic Differentiation precompiler software tool ADIFOR. Aerodynamic shape optimization of the complete aircraft with twenty four design variables is performed. Unstructured and structured volume grids and Euler solutions are obtained to demonstrate the feasibility of the new surface definition.

Future investigations should include the implementation of present approach using larger grid dimensions, adequate to resolve full physics of viscous flow analysis. A grid optimization mechanism based on grid sensitivity coefficients with respect to grid parameters should be included in the overall optimization process. An optimized grid applied to present geometry, should increase the quality and convergence rate of flow analysis within optimization cycles. Other directions could be establishing a link between the graphic software and the optimization procedure, such that the changes in the design variables in each optimization cycle is visible to the user. Another contribution would be the extension of the current algorithm to represent complex configurations. A hybrid approach can be selected where certain sections or skeletal parts of a surface are specified analytically and interpolation formulas are used for intermediate surfaces.