THREE-DIMENSIONAL FLOW SOLUTIONS FOR NON-LIFTING FLOWS USING FAST MULTIPOLE BOUNDARY ELEMENT METHOD

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

UĞUR KARBAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
AEROSPACE ENGINEERING

SEPTEMBER 2012

Approval of the thesis:

THREE-DIMENSIONAL FLOW SOLUTIONS FOR NON-LIFTING FLOWS USING FAST MULTIPOLE BOUNDARY ELEMENT METHOD

Submitted by **UĞUR KARBAN** in partial fulfillment of the requirements for the degree of **Master of Science in Aerospace Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan ÖZGEN _____
Dean, **Graduate School of Natural and Applied Sciences**

Prof. Dr. Ozan TEKİNALP _____
Head of Department, **Aerospace Engineering Dept.**

Assoc. Prof. Dr. Oğuz UZOL _____
Supervisor, **Aerospace Engineering Dept., METU**

Asst. Prof. Dr. Nilay SEZER UZOL _____
Co-supervisor, **Mechanical Engineering Dept., TOBB ETU**

Examining Committee Members:

Prof. Dr. İsmail H. TUNCER _____
Aerospace Engineering Dept., METU

Assoc. Prof. Dr. Oğuz UZOL _____
Aerospace Engineering Dept., METU

Prof. Dr. Yusuf ÖZYÖRÜK _____
Aerospace Engineering Dept., METU

Assoc. Prof. Dr. Dilek Funda KURTULUŞ _____
Aerospace Engineering Dept., METU

Asst. Prof. Dr. Cüneyt SERT _____
Mechanical Engineering Dept., METU

Date: 17/09/2012

I hereby declare that all the information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Uğur KARBAN

Signature :

ABSTRACT

THREE-DIMENSIONAL FLOW SOLUTIONS FOR NON-LIFTING FLOWS USING FAST MULTIPOLE BOUNDARY ELEMENT METHOD

Karban, Uğur
M.Sc., Department of Aerospace Engineering
Supervisor: Assoc. Prof. Dr. Oğuz Uzol
Co-Supervisor: Asst. Prof. Dr. Nilay SEZER UZOL

September 2012, 66 Pages

Driving aim of this study was to develop a solver which is accurate enough to be used in analysis and fast enough to be used in optimization purposes. As a first step, a three-dimensional potential flow solver is developed using Fast Multipole Boundary Element (FMBEM) for calculating the pressure distributions in non-lifting flows.

It is a steady state solver which uses planar triangular unstructured mesh. After the geometry is introduced, the program creates a prescribed wake surface attached to the trailing edge(s), obtains a solution using panel elements on which the doublet and source strengths vary linearly. The reason for using FMBEM instead of classical BEM is the availability of solutions of systems having DOFs up to several millions within a few hours using a standard computer which is impossible to accomplish with classical BEM. Solutions obtained for different test cases are compared with the analytical solution (if applicable), the experimental data or the results obtained by JavaFoil.

Keywords: Potential flow theory, 3D Panel Code, Fast Multipole Boundary Element Method.

ÖZ

KALDIRMA KUVVETİ YARATMAYAN AKIŞLARDA ÇOK-KUTUP SINIR ELEMANLARI YÖNTEMİYLE ÜÇ BOYUTLU AKIŞ ÇÖZÜMLERİ

Karban, Uğur
Yüksek Lisans, Havacılık ve Uzay Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Oğuz Uzol
Ortak Tez Yöneticisi: Yrd. Doç. Dr. Nilay SEZER UZOL

Eylül 2012, 66 Sayfa

Bu çalışmanın yürütülmesine neden olan ana fikir, hem analiz amaçlı kullanılabilecek kadar doğru sonuçlar veren, hem de eniyileştirme süreçlerinde kullanılabilecek kadar hızlı bir akış çözücü geliştirmektir. Bu kapsamda ilk aşama olarak kaldırma kuvveti yaratmayan akışlarda oluşan basınç dağılımlarını inceleyebilmek için "Hızlı Çok-Kutup Sınır Elemanları Yöntemi"ni temel alan bir kod geliştirilmiştir.

Çözücü, girdi olarak üçgen elemanlı düzensiz ağ şeklinde örülmüş geometriyi kullanmaktadır ve kararlı hal problemlerini çözmektedir. Kod, geometri tanıtıldıktan sonra firar kenar(lar)ına yapışık önceden tanımlı bir iz bölgesi oluşturmakta ve değerlerin doğrusal olarak değiştiği panel elemanları kullanarak çözüm elde etmektedir. Hızlı Çok-Kutup yönteminin klasik sınır elemanları yönteminin yerine tercih edilmesinin nedeni, bu yöntemin büyük çaplı sistemlerin çözümüne izin vermesidir. Klasik yöntemle, standart bir bilgisayarda on bin elemanlı bir problemin çözümü mümkün olmazken Hızlı Çok-Kutup yöntemi birkaç milyon elemanlı bir

problemi bile birkaç saat içerisinde çözebilmektedir. Farklı test durumları için çözümler (uygulanabilir olduğu durumlarda) analitik, deneysel ya da mevcut potansiyel kodlardan elde edilen sonuçlarla karşılaştırılmıştır.

Anahtar Kelimeler: Potansiye Akış Teorisi, 3 Boyutlu Panel Metod, Hızlı Çok-Kutup Sınır Elemanları Yöntemi.

To My Parents

And My Brother

ACKNOWLEDGEMENTS

My grateful thanks go to my supervisor Assoc. Prof. Dr. Oğuz UZOL together with my co-supervisor Assist. Prof. Dr. Nilay SEZER UZOL for their guidance and encouragements which made me finish my dissertation.

And my special thanks are for Ms. Özlem CEYHAN who had always become very eager to help me and had answered all my questions with a great kindness. Her advices and encouragements helped me a lot during this thesis study.

Also I would like to thank to the inhabitants in 172/4 for their admirable tolerance and patience towards me about being a bad co-worker throughout this thesis study.

And finally, my deepest thanks go to my parents and my brother who have always given their endless love and support all through my life. I dedicate this study to them.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF SYMBOLS

SYMBOLS

Φ	Total potential
ϕ	Perturbation Potential
σ	Source strength
μ	Doublet strength
n	Surface normal
r	Distance between the source and field points
S	Boundary of the domain
$R_{n,m}$, $S_{n,m}$	Solid Harmonic Function used in Multipole Expansions
P_n^m	Associated Legendre function
G	Fundamental solution of the potential equation
$M_{n,m}$	Moment expansion coefficient
$L_{n,m}$	Local expansion coefficient
U_∞	Free stream velocity
t^*	Artificial time parameter used in wake relaxation
ν	Dynamic viscosity
C_p	Pressure coefficient
A	System matrix used in GMRES solver
K_k	Krylov subspace defined for the system matrix
J	Objective function to be minimized in during GMRES process
V_k	Orthonormal basis for Krylov subspace
H_k	Hessenberg Matrix

CHAPTER 1

INTRODUCTION

Use of panel codes, which is a Boundary Element Method (BEM) application, in potential flow solutions is a long history starting with Hess & Smith [**1**], actually they were not calling it as "panel code", in 1962. Since then, a lot of people have studied it and numerous codes have been developed. The first panel code used constant singularity distribution with rectangular panel elements. And as more analysis performed on the issue, various codes such as the ones using linear / quadratic singularity distributions with flat / paraboloidal panel elements are developed. The major concern on the evolution of panel codes was to represent the surfaces of the geometry and potential distribution on these surfaces with least possible elements. This is because increasing the number of elements in BEM is quite expensive in terms of computation time. When the problem size increased with $O(N)$, the number of operations to be performed increase with $O(N^2)$ in BEM. If direct solvers such as Gauss Elimination Method are used for solving the linear system, the order gets even higher to $O(N^3)$.

A different approach in order to make large problems available for BEM is introduced by Rokhlin [**2**] and developed by Greengard [**3**] which is called Fast Multipole Boundary Element Method (FMBEM). Instead of trying to keep the number of elements at minimum, FMBEM keeps the order of increase in the number of operations at minimum. It is such that when the problem size increased with $O(N)$, the number of operations to be performed also increase with $O(N)$ when FMBEM approach is used. Although the coefficient in front of the order $O(N)$ is

quite big, as the problem size reaches to a few thousands of DoFs, classical BEM starts to cost more in terms of computation time when compared to FMBEM.

1.1 Literature Survey

The literature survey conducted for this thesis study starts from the potential theory which also forms the basis of the fast multipole method. As a complete reference for the potential theory investigated within the perspective of aerodynamics, Katz and Plotkin[**4**] investigated the potential problem together with all of its applications in the book they wrote. Although it is a quite large scaled work offering a good perception of low speed aerodynamics, the author focused on the parts explaining the establishment of the potential problem, singularity elements and the details of the panel method application. The book explains all the steps in order to construct a low order panel code clearly and provides a sample code also. And it helped the author gain a general perspective about the panel method and its synchronization with the boundary element methods.

An interesting application of a constant element panel method is conducted by Browne and Ashby [**5**] in which they used the panel method for estimation of the wind tunnel effect and correction of the wind tunnel measurements. As the method for the correction, they model the system to be investigated experimentally in the wind tunnel together with the wind tunnel and try to match the results obtained from the panel code with the experimental ones. After a match obtained, the walls in the numerical model are removed in order and a new solution is obtained in order to eliminate the wind tunnel effect. They present a tedious derivation for the analytical integration of the influence coefficients over constant panels.

Using panel methods for rotary wing or propeller calculations is also widespread. An example is the Ph.D thesis of Jin-Tae Lee [**6**]. He performed potential analysis on marine propellers using a constant element panel method and corrected the pressure distribution at the trailing edge using “Iterative Pressure Kutta Condition” (IPKC). He also gave the derivation of the equivalence of dipole and vorticity distribution for constant panel elements. A more general form of this equivalence is given in [**4**]

which includes a term accounting the gradient of the dipole distribution on the panel surface and hence it can be used for higher order elements. Another study applying low order panel method to marine propellers is conducted by Falcão de Campos [**7**] who also used IPKC for trailing edge but gave a more detailed description for it.

When higher order elements are selected for singularity distributions, obtaining analytical solutions to influence coefficient integrals requires a complicated analysis. One of the most popular codes of this type is the one written by Magnus and Epton [**8**] which is named as PAN AIR. It uses linear source and quadratic doublet distributions over flat panel elements. In PAN AIR, the source and doublet strengths are defined using first and second order polynomials respectively. And the coefficients of these polynomials are obtained from matrix equations obtained by dividing a panel into eight subpanels and also taking the relationship of each panel with its neighbors into consideration. Using these matrices defined on the subpanels and neighbors, the integrals in the influence coefficients are separated as regular and singular parts and only the regular parts are calculated since the singular parts cancel each other analytically. A similar approach in higher order panel methods is use of B-splines such that it is aimed to represent both the geometry and the singularity distributions using B-spline expansions. Application of B-splines to the potential flow problems is done by Maniar [**9**] in his Ph.D dissertation. He also used dividing the panels into subpanels taking the collocation points as reference and separating the regular and singular parts of the integrals in self influence coefficients.

Use of higher order methods in such a manner brings too much complexity in the foundation of the problem and the tools used in the solution. This is not preferred by the author hence integrating this approach with Fast Multipole algorithm is not easy and have not been done yet. Instead, a simpler formulation is used where linear singularity distributions are used and the numerical integration schemes are used for calculation of the influence coefficients. The problem with this approach is that numerical integration is not directly applicable since the integrand functions are becomes singular for self influence coefficients. In order to avoid this problem, the potential equation is desingularized so that numerical integration schemes become

available. A pioneering work on the issue is conducted by Liu and Rudolphi in [**10**] and [**11**]. They proposed a desingularization for both the strongly singular part in the doublet terms and the weakly singular part in the source terms. Another study on the issue is by Kouh and Suen [**12**] for potential problems with stationary boundaries. Later it is called "true desingularization" and an excellent explanation for desingularization of source and doublet terms in both equation and matrix forms by Klaseboer et al [**13**]. The non-singular equation obtained in [**13**] was for potential problems in infinite domain and with non-stationary boundaries. A corrected version for internal potential problems is given in [**14**] by Klaseboer et al again. This true desingularization works great for conventional boundary elements method. It is independent of the coefficient defined by the solid angle at where the collocation point is located. This brings an extra ease when applying the method for the geometries with sharp edges. Alternatively, as it is mentioned in [**4**], the weak singularity in the source term can be removed by cartesian to polar coordinate transformation. This method is adopted by the author for the source terms since it is not possible to use multipole expansions when true desingularization mentioned in [**13**] is applied.

While implementing the Fast Multipole algorithm to the solver, the author used the book by Liu [**15**] as the main reference. The book starts with introducing the mathematical tools required and then gives the details of the conventional BEM. It also explains the desingularization of the potential equation with smooth, i.e. having no sharp edges, and non-stationary boundaries. Afterwards, it introduces the FMBEM conceptually, defines and explains the tools required for FMBEM and finally gives the details of the Fast Multipole algorithm. All the equations are derived for 2D potential equation and the ones for 3D problems are given without providing any derivation. The book mainly focuses on non-adaptive Fast Multipole algorithm and only mentions about the adaptive method. It includes applications of FMBEM to various engineering problems which can be defined using the potential theory such as elastostatic problems, stokes flow problems and acoustic wave problems. Samples codes written in Fortran 77 for 2D classical BEM and FMBEM are also provided in the appendices. As the second reference, the author puts the Ph.D dissertation by

Yoshida [**16**]. He provided the details of the Adaptive Fast Multipole algorithm and focused on 3D problems only. He selected crack problems as the specific application area. He also provided all the derivations of the mathematical tools used in 3D FMBEM what made the author give special emphasis on this study. He also gave regularization of boundary integral equations for various potential problems. An interesting application of FMBEM is done by Lin and Liao [**17**]. They tried to calculate the added mass coefficient of a submarine by using FMBEM and compared the results with the experimental ones. As a test case, they also investigated uniform flow over a sphere which can be solved analytically. The same test case is investigated again with the same method but with a different name by Nabors et al [**18**]. They called the method as "Preconditioned, Adaptive, Multipole Accelerated" (PAMA) algorithm. They performed a detailed complexity analysis for the method they introduced which has actually no difference with FMBEM.

1.2 Objective and Scope

The motivating idea behind this study was to develop a 3D solver which couples the potential theory with the Integral Boundary Layer (IBL) theory which is accurate enough to be used in analysis and fast enough to be used in optimization purposes. And as a specific application area, performance and noise estimation over wind turbines was selected. However it is decided to limit this thesis work with developing a 3D BEM solver for non-lifting flows. Extending the study to lifting flows and coupling the inviscid solution with the viscous part through IBL equations and developing a noise estimation tool for wind turbines are set as future works of the study.

In order to obtain a solver capable of solving very large problems which are out of the range of conventional panel methods, it is decided to use the Fast Multipole Boundary Element Method. While developing the code, it is aimed that the code could handle any geometry which is meshed properly but of any size. For this, pointers and dynamic allocation of the memory are frequently used in various parts of the code.

This thesis study mainly focuses on application of FMBEM on potential flow solutions. Flow solutions obtained by a classical panel method compared with FMBEM. Both solvers use linear singularity distributions and the solution domain is discretized using unstructured grid with triangular panel elements. Triangular meshing allows generating grids on any geometry without facing skewness problems. And using linear elements both increases the accuracy of the solution and makes the velocity calculations quite easy. The collocation points are placed on the vertices of the panel elements. Normally this causes a stability problem in terms of numerical convergence but, in order to remove the instabilities, desingularization is applied to the total potential equation and fast multipole algorithm is written for the desingularized equation.

The results are tried to be verified through analytical and experimental comparisons. First a rectangular high aspect ratio wing generated from Van de Vooren airfoil is investigated and the results are compared with the analytical 2D solution of the airfoil. Secondly, the test cases investigated in the NASA report written by Caradonna [**19**] are repeated. And finally the pressure coefficient distributions obtained for NREL Phase VI test wind turbine are compared with the experimental ones.

1.3 Thesis Content

In “Theory” part, the equations for Potential Theory and FMBEM are given. A detailed derivation of the potential equation is presented first for making the problem more comprehensible. Subsequently the desingularization and discretization processes are explained in detail. After completing the potential theory, the concept how the potential problem is treated with Fast Multipole method is explained briefly. And the tools used in FMBEM are introduced together with the related equations. The Fast Multipole algorithm is explained in a general manner and the details left to the next section. No derivations are provided for the FMBEM equations, the author confined himself to give proper citation to the relevant references. This is done for

not losing concentration by complicated derivation processes but focusing on the implementation of the algorithm.

In the next section, the details of the implementation of the Fast Multipole algorithm to the solver are introduced. The process of importing and modifying the mesh data is explained first. And then the concept used in generating the oct-tree structure is introduced. Afterwards, the things to be done in Fast Multipole algorithm are given step by step.

After giving the details of the algorithms used in the solver, investigation of the test cases and comparisons with analytical / experimental data are conducted in the "Applications and Results" section. A high aspect ratio wing made by padding the Van de Vooren airfoil profile is investigated and the results are compared with 2D analytical solution of the airfoil. Besides solutions for various symmetric airfoils and non-lifting flow conditions are also presented together with the JavaFoil [**20**] solutions.

Comments on the thesis study and concluding remarks are presented in the "Conclusion" section. Also a general description of integrating this potential solver with IBL equations is given here.

CHAPTER 2

THEORY

In this part, theoretical background of this thesis study is presented. It starts with the derivation of the total potential equation. Following is the desingularization and the discretization of the potential equation. This completes the potential theory. After introducing the classical part, equations constructing FMBEM and the tools that FMBEM uses are given. As the last two parts of this section, details of wake relaxation and IPKC are introduced.

2.1 Incompressible, Potential Flow Equations

For an incompressible, potential flow the continuity equation can be simplified to the form;

$$\nabla^2 \Phi = 0 \tag{2.1}$$

which is nothing but the Laplace's equation. Here the flow is defined in a domain V (see Figure 1) which has the boundary surface S. In such a domain any vector can be treated using the divergence theorem. According to the divergence theorem, the following equality is valid where q being a vector defined everywhere in the domain V;

$$\int_S \boldsymbol{n} \cdot \boldsymbol{q}\, dS = \int_V \nabla \cdot \boldsymbol{q}\, dV \tag{2.2}$$

If one defines the vector q as a combination of two scalar functions of position ϕ_1 and ϕ_2 and their gradients such as;

$$\boldsymbol{q} = \phi_1 \nabla \phi_2 - \phi_2 \nabla \phi_1 \tag{2.3}$$

Then Green's second identity is obtained which forms the basis for the potential equation;

$$\int_S (\phi_1 \nabla \phi_2 - \phi_2 \nabla \phi_1) \cdot n\, dS = \int_V (\phi_1 \nabla^2 \phi_2 - \phi_2 \nabla^2 \phi_1)\, dV \tag{2.4}$$

Setting $\phi_1 = \frac{1}{r}$ and $\phi_2 = \Phi$ will let one to calculate the potential at a point P(x, y, z) where Φ is the potential of the flow in the domain V and r is the distance to P.

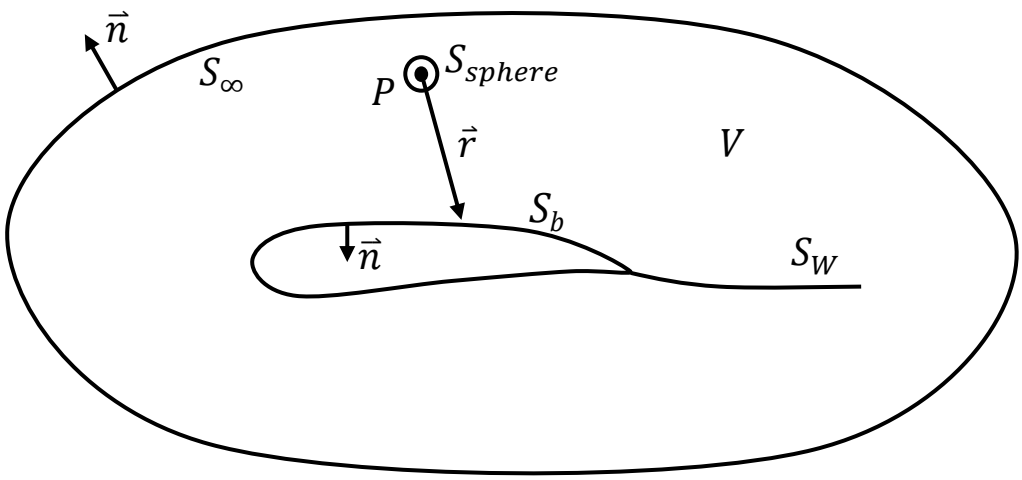

Figure 1 - Schematic Representation of the Domain and Its Boundaries

When P is outside V, then both ϕ_1 and ϕ_2 are continuous (and both satisfy the Laplace equation) hence, the Green's identity becomes;

$$\int_S \left(\frac{1}{r}\nabla\Phi - \Phi\nabla\frac{1}{r}\right)\cdot \boldsymbol{n}\, dS = 0 \tag{2.5}$$

Where the point P is inside the domain V, then a small sphere around P and having radius ϵ should be excluded from the domain to prevent singularity (since $r \to 0$, $\frac{1}{r} \to \infty$). Then the Green's identity becomes;

$$\int_{S+S_{sphere}} \left(\frac{1}{r}\nabla\Phi - \Phi\nabla\frac{1}{r}\right)\cdot \boldsymbol{n}\, dS = 0 \tag{2.6}$$

Treating of this integral equation around the sphere requires introducing a spherical coordinate system inside the sphere. Since n points towards the center of the sphere, it can be defined as $-e_r$ and then, then integral around the sphere becomes;

$$\int_{S_{sphere}} \left(\frac{1}{r}\nabla\Phi - \Phi\nabla\frac{1}{r}\right)\cdot \boldsymbol{n}\, dS = -\int_{S_{sphere}} \left(\frac{1}{r}\frac{\partial\Phi}{\partial r} + \frac{\Phi}{\mathrm{r}^2}\right)\cdot \boldsymbol{n}\, dS \tag{2.7}$$

Since r equals ϵ every where on the surface of the sphere and assuming that the potential will not vary too much inside the sphere the above equation can be rewritten as;

$$\begin{aligned}\int_{S_{sphere}} \left(\frac{1}{r}\frac{\partial\Phi}{\partial r} + \frac{\Phi}{\mathrm{r}^2}\right)\cdot \boldsymbol{n}\, dS &= -\left(\frac{\Phi(P)}{\epsilon^2}\right) \\ &= -\left(\frac{\Phi(P)}{\epsilon^2}\right)4\pi\epsilon^2\end{aligned} \tag{2.8}$$

Then the equation which yields the potential at the point P which is inside the domain V becomes as follows;

$$-4\pi\Phi(P) + \int_S \left(\frac{1}{r}\nabla\Phi - \Phi\nabla\frac{1}{r}\right)\cdot \boldsymbol{n}\, dS = 0 \tag{2.9}$$

$$\Phi(P) = \frac{1}{4\pi}\int_S \left(\frac{1}{r}\nabla\Phi - \Phi\nabla\frac{1}{r}\right)\cdot \boldsymbol{n}\, dS \qquad (2.10)$$

If the point P was taken on the boundary S_b, then a semi-sphere would be used to remove the singularity from the flow domain (here it is assumed that the surface S_b is smooth everywhere) and this would yield the equation;

$$\Phi(P) = \frac{1}{2\pi}\int_S \left(\frac{1}{r}\nabla\Phi - \Phi\nabla\frac{1}{r}\right)\cdot \boldsymbol{n}\, dS \qquad (2.11)$$

For a more genera case, namely, in the existence of sharp edges, the potential equation can be written as,

$$\mathrm{c(P)}\Phi(P) = \int_S \left(\frac{1}{r}\nabla\Phi - \Phi\nabla\frac{1}{r}\right)\cdot \boldsymbol{n}\, dS \qquad (2.12)$$

where c(P) is a coefficient showing the solid angle on the surface, as it is mentioned above it is zero for a point outside the domain, 4π for a point inside the domain and 2π for one on a smooth surface.

Adding the internal potential

If one assumes an artificial internal flow inside the domain S_b having a potential Φ_i, then for the point P taken inside V will be exterior to this new domain and hence the potential at point P due to this artificial flow will be zero which can be shown as;

$$0 = \frac{1}{4\pi}\int_S \left(\frac{1}{r}\nabla\Phi_\mathrm{i} - \Phi_\mathrm{i}\nabla\frac{1}{r}\right)\cdot \boldsymbol{n}\, dS \qquad (2.13)$$

Then, an alternative equation for the total potential at point P adding this artificial potential effect can be written as;

$$4\pi\ \Phi(P) = \int_{S_b} \left(\frac{1}{r}\nabla(\Phi - \Phi_{\mathrm{i}}) - (\Phi - \Phi_{\mathrm{i}})\nabla\frac{1}{r}\right) \cdot \boldsymbol{n}\, dS + \int_{S_w + S_\infty} \left(\frac{1}{r}\nabla\Phi - \Phi\nabla\frac{1}{r}\right) \cdot \boldsymbol{n}\, dS \tag{2.14}$$

One should note that the minus sign in front of the internal potential terms comes from the changing direction of the normal vector n in Φ_{i} which is actually towards outside S_b.

The contribution from the surface S_∞ can be written shortly as ϕ_∞. Also if the following to definitions are made;

$$-\mu = \Phi - \Phi_i \tag{2.15}$$

$$-\sigma = \frac{\partial\Phi}{\partial n} - \frac{\partial\Phi_i}{\partial n} \tag{2.16}$$

Where the term μ is called as a "doublet" and the term σ is called a "source".
The total potential is defined as the sum of the "perturbation potential" and the potential due to the free stream;

$$\Phi = \Phi_\infty + \phi \tag{2.17}$$

Then the potential equation may be given the following form;

$$4\pi\ \phi_P = -\int_{S_b} \left(\frac{1}{r}\sigma\right) dS + \int_{S_b} \left(\mu\frac{\partial r}{\partial n}\right) dS + \int_{S_w} \left(\mu\frac{\partial r}{\partial n}\right)\ dS \tag{2.18}$$

Here, ϕ_P is the perturbation potential at P which is defined as "source point". And any point collected on the boundaries for the integration is called a "field point".

2.1.1 Desingularization of the Potential Equation

As the source point is approached to the field point, where it occurs when the source points are used as collocation points, singularities arise in the integrands with the orders of $O(\frac{1}{r^2})$ in the doublet term and of $O(\frac{1}{r})$ in the source term. When the domain is constructed of constant elements, analytical solutions are available for the integration of both the doublet and source terms. But for the higher order elements, obtaining an analytical solution requires a complicated calculation process. And numerical integration methods such as "Gaussian Quadrature" integration scheme cannot be directly applied since the integrand functions are singular. In order to overcome this problem, "desingularization" is applied on the potential equation, so that numerical integration methods become available.

For a complete desingularization of the potential equation one can investigate Liu and Rudolphi's [**11**] work on the issue or the tedious study conducted by Klaseboer et al [**13**] where the details of the operations in the matrix form are also included. However, it is not possible to apply the multipole expansions to the fully desingularized form of the potential equation. Hence only the doublet terms which have $O(\frac{1}{r^2})$ singularity are removed. Actually for the source terms, $O(\frac{1}{r})$ singularity may be removed by making a Cartesian to polar transformation. For this, the collocation points should be placed at the vertices of the boundary element which is the case in this study.

As a first step of desingularization, the collocation point is located on internal side of the solid body. Since the total potential is constant in that region due to the Dirichlet boundary condition, the derivatives of the potential (namely, the source terms) vanishes from the potential equation. Then the following is obtained;

$$4\pi - c = \int_{S_b} \left(\frac{\partial r}{\partial n}\right) dS \tag{2.19}$$

where the term $4\pi - c$ indicates the solid angle on the internal side of the solid body. Also one should note the sign change due to change in the direction of the surface normal. Then the coefficient c can be replaced in the total potential equation as follows;

$$\begin{aligned} 4\pi\, \phi_P = &-\int_{S_b} \left(\frac{1}{r}\sigma\right) dS + \int_{S_b} \left[(\mu + \phi_P)\frac{\partial r}{\partial n}\right] dS \\ &+ \int_{S_w} \left[(\mu + \phi_P)\frac{\partial r}{\partial n}\right] dS \end{aligned} \tag{2.20}$$

With the given form, the above equation gets rid of the strong singularity. As mentioned before the weak singularity is treated by the coordinate transformation.

2.1.2 Boundary Conditions

Boundary conditions for potential flow problems in infinite domain may be defined using two different approaches. These are called "Neumann Type Boundary Condition" and "Dirichlet Type Boundary Condition". When potential flow problems are considered it can be said that Neumann BCs are more physical and Dirichlet type is actually derived from this one.

Neumann Type Boundary Condition

This condition states that the component of the flow velocity (gradient of the total potential) which is normal to the surface of the solid body has to be equal to zero. In other words, flow on the surface should always be tangential to the surface. It is expressed as follows;

$$\nabla\Phi_P \cdot \boldsymbol{n} = 0 \qquad where\ P \in S_b \tag{2.21}$$

Dirichlet Type Boundary Condition

This condition is derived from the Neumann Type BC. Since the flow is always tangent to the solid body surface, then the flux through surface is zero since the normal component of the velocity is zero. This fact brings the condition that the potential inside the solid body should be constant which is called the Dirichlet Boundary Condition. Letting Φ_i be the total internal potential, it is expressed as follows;

$$\nabla\Phi_{i_P} \cdot \boldsymbol{n} = 0 \tag{2.22}$$

For the outer boundary which is defined as S_∞, the only boundary condition is that;

$$\lim_{r\to\infty} \nabla\Phi = 0 \tag{2.23}$$

Within this thesis work, Dirichlet Type BC will be adopted when creating the linear system of equations since it is easier to combine with the FMBEM theory.

Kutta Condition

Despite of implementation of either Neumann or Dirichlet BCs, the problem remains unfixed for lifting flows which means the solution is not unique. In order to fix the problem one has to determine the strength of the vorticity around the lifting body (which is actually the strength of the wake) and the shape of the wake. There are various approaches for determining the wake strength. The most general and the simplest approach is to implement the "Morino's Kutta condition"[21] as it is done in 2D flows. It requires setting the trailing edge vorticity equal to zero which brings out the following equation;

$$\mu_U - \mu_L - \mu_W = 0 \tag{2.24}$$

Here the subscripts μ_U and μ_L correspond to the doublet strengths at the upper and lower surfaces on the trailing edge and μ_W is the wake strength.

2.1.3 Discretization of Potential Equation

For solving the potential problem, one has to reduce the problem to a set of linear equations. For this purpose, the boundaries are divided into small elements. These elements may be constant, linear or quadratic which affects the variation of the doublet/source strengths within the borders of these elements. If the element is constant, one needs to specify the doublet and source strengths at anywhere on the element (generally at the middle point of the element). For a linear element, one should specify those at the vertices of the element and for quadratic one, an extra point on each edge should be determined besides the vertices. In order to have a linearly independent system, it is required to have N equations for N points to be determined. These equations are obtained by applying the boundary condition on the collocation points selected on the domain. For this thesis study, linear elements are used for discretization. A schematic representation of the discretized geometry and the collocation points are shown in the following figure;

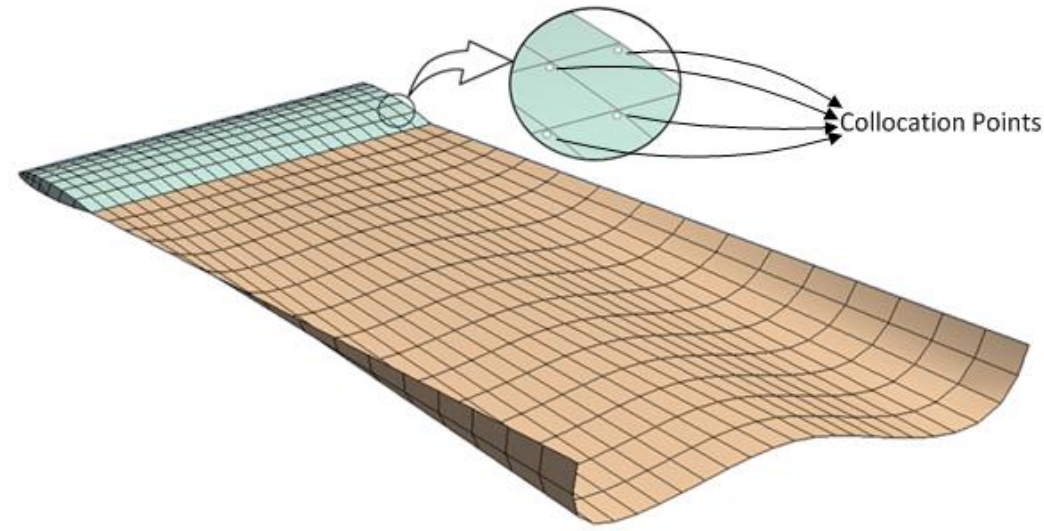

Figure 2 - Discretized Geometry and the Collocation Points

For N solid body elements and N_W wake elements the potential equation can be rewritten in discrete form as follows;

$$4\pi\phi_{\mathrm{P}} = -\sum_{i}^{N}\int_{i}\left(\frac{1}{r}\sigma\cdot n\right)dS + \sum_{i}^{N}\int_{i}\left[(\mu+\phi_P)\frac{\partial r}{\partial n}\right]dS + \sum_{k}^{N_W}\int_{k}\left[(\mu+\phi_P)\frac{\partial r}{\partial n}\right]dS \tag{2.25}$$

Where i and k represent the solid body and wake elements respectively. Before going further on discretization, it is better to reduce the "$\sigma\cdot n$" term into a simpler form. As it is mentioned before source strength is defined as;

$$-\sigma = \frac{\partial\Phi}{\partial n} - \frac{\partial\Phi_i}{\partial n} \tag{2.26}$$

Considering the definition of the total potential, the above equation can be rewritten as follows;

$$-\sigma = \frac{\partial(\phi+\phi_\infty)}{\partial n} - \frac{\partial(\phi_i+\phi_\infty)}{\partial n} \tag{2.27}$$

The internal perturbation potential ϕ_i is an artificial potential, hence it can be set arbitrarily. Here it will be set to zero for simplicity (this implies a ground-fixed observer and non-stationary boundary condition). When ϕ_i is equal to zero, $\frac{\partial\phi_i}{\partial n}$ is also zero. With these arrangements the source strength is set as;

$$-\sigma = \frac{\partial\phi}{\partial n} = \nabla\phi\cdot\boldsymbol{n} \tag{2.28}$$

which is nothing but the normal component of the perturbation velocity. As a result of Neumann BC, the total velocity (sum of perturbation and free stream velocities) should be tangential to the body surface, namely, its normal component should be equal to zero.

$$\nabla\phi \cdot \boldsymbol{n} + \nabla\phi_\infty \cdot \boldsymbol{n} = 0 \tag{2.29}$$

Using the above relation the source strength becomes;

$$\sigma = \nabla\phi_\infty \cdot \boldsymbol{n} = U_\infty \cdot \boldsymbol{n} \tag{2.30}$$

The term $\frac{\partial r}{\partial n}$ may also be written as follows;

$$\frac{\partial r}{\partial n} = -\frac{\vec{r} \cdot \boldsymbol{n}}{r^3} \tag{2.31}$$

One should also note that when the internal perturbation potential is set to zero, the potential ϕ becomes equal to the negative of the doublet strength $(-\mu)$. Regarding Hence the discrete equation for the perturbation potential is written in its final form as follows;

$$\begin{aligned} \phi(P) = &-\frac{1}{4\pi}\sum_{i}^{N}\int_i \left(\frac{1}{r}U_\infty \cdot \boldsymbol{n}\right) dS \\ &-\frac{1}{4\pi}\sum_{i}^{N}\int_i \left(\mu\frac{\vec{r} \cdot \boldsymbol{n}}{r^3}\right) dS \\ &-\frac{1}{4\pi}\sum_{k}^{N_W}\int_k \left(\mu\frac{\vec{r} \cdot \boldsymbol{n}}{r^3}\right) dS \end{aligned} \tag{2.32}$$

Here, one should note that the first integrand function at the right hand side of the above equation is used to construct the right hand side vector of the linear system while the remaining terms are added into the influence coefficient matrix. Integrations on these linear elements are made using the “Gaussian Quadrature” numerical integration scheme based on a local coordinate system on the panel. The

schematic representation of a single panel element and the source point with the local coordinate system placed on the element is depicted below;

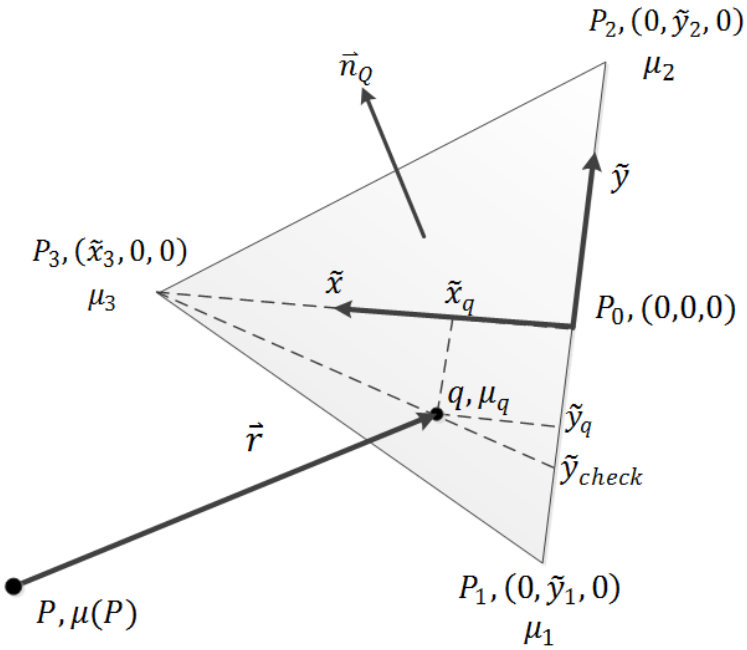

Figure 3 - Local Coordinate System Placed on a Panel Element

Here P is the source point and $\mu(P)$ is the doublet strength at the source point. P_0 is the center of the local coordinate system. "q" is one of the reference points that Gaussian Quadrature (GQ) use for approximating the integrand function. There are 16 points used for such an approximation in a fourth order GQ algorithm applied on a double integral (which is the case being investigated).

Since all the integrand functions are scalar in the potential equation, use of different coordinate systems will yield the same result in case each system is used consistently. Here in order to avoid complex calculations, all the points and vectors are defined with respect to the local coordinate system and then the value of the integrand function is calculated accordingly. Applying n^{th} order GQ scheme to one of the integrand functions using the local coordinate system, the following equation is obtained;

Let;

$$f = \mu \frac{\vec{r} \cdot \boldsymbol{n}}{r^3} \tag{2.33}$$

Then the double integral of "f" taken on the given element "Q" is written as follows;

$$\int_Q \left(\mu \frac{\vec{r} \cdot \boldsymbol{n}}{r^3} \right) dS = \frac{h_1}{2} \sum_{i=1}^{n} \left[c_{n,i}(d_1 - c_1) \sum_{j=1}^{n} c_{n,j} f(h^*, d^*) \right] \tag{2.34}$$

where,

$$h^* = h_1 r_{n,i} + h_2$$
$$d^* = \frac{(d_1 - c_1) r_{n,j} + d_1 + c_1}{2}$$

and the bounds of the integral domain are given as;

$$a = \tilde{x}_3; \;\; b = 0; \;\; c(x) = \left(1 - \frac{x}{x_3}\right) \tilde{y}_1; \;\; d(x) = \left(1 - \frac{x}{x_3}\right) \tilde{y}_2$$

And using the following definitions;

$$h_1 = \frac{b - a}{2}; \;\; h_2 = \frac{b + a}{2}$$
$$c_1 = c\left(h_1 r_{n,i} + h_2\right); \;\; d_1 = d(h_1 r_{n,i} + h_2)$$

Where $r_{n,i}$ corresponds to the reference points used in GQ and $c_{n,i}$ defines the weights corresponding to these reference points. For a fourth order GQ scheme the following values are used for these two;

Table 1 - Coefficients of 4th Order Gaussian Quadrature

Index	Reference Points	Weights
$1, 2$	$\pm\sqrt{(3-2\sqrt{6/5})/7}$	$\frac{18+\sqrt{30}}{36}$
$3, 4$	$\pm\sqrt{(3+2\sqrt{6/5})/7}$	$\frac{18-\sqrt{30}}{36}$

And the integrand function f in the GQ equation is expanded as follows;

$$f = \sum_{i=1}^{3} coef_i . \mu_i \frac{\vec{r} \cdot n}{r^3} \tag{2.35}$$

Where;

$$coef_1 = \frac{\tilde{y}_2 - \tilde{y}_{check}}{\tilde{y}_2 - \tilde{y}_1}\left(1 - \frac{\tilde{x}_q}{\tilde{x}_3}\right)$$

$$coef_2 = \frac{\tilde{y}_{check} - \tilde{y}_1}{\tilde{y}_2 - \tilde{y}_1}\left(1 - \frac{\tilde{x}_q}{\tilde{x}_3}\right)$$

$$coef_3 = \frac{\tilde{x}_q}{\tilde{x}_3}$$

$$\tilde{y}_{check} = \frac{(\tilde{x}_3 - \tilde{x}_q)}{\tilde{x}_3}\tilde{y}_q$$

2.2 Fast Multipole Boundary Element Method

When trying to solve a potential problem with BEM, it is very advantageous to use only the boundaries as an element source instead of the whole domain since the number of elements to be calculated drops dramatically this way. However though this advantage, BEM does not allow to solve large problems (i.e. having DoFs at the order of 10^5 or higher) since the number of operations to generate the linear system to be solved increase by $O(N^2)$ as the number of elements to be solved is increased by O(N). This problem arises from the fact that one needs to calculate the interaction

of each element with the other elements. If it is determined to use a direct solver to solve this linear system then the situation is even worse since the number of operations to solve a linear system with N DoF is defined by $O(N^3)$. The increase rate for the required memory is also defined by $O(N^2)$ in classical BEM. This means memory allocation problems arises for large sized potential problems even one has enough time to wait the solution.

Fast Multipole Method, introduced by Rokhlin [**2**] and developed by Greengard [**3**], substantially removed this problem and made the use of boundary element method possible for problems up to several millions of DoFs. They managed to keep both the number of operations and the required memory at O(N) as the system is increased by O(N). The method achieves this by grouping the elements in a tree structure and by adding the effect of far elements in a cumulative manner while keeping the direct integration for the close elements. The boundary element method which uses FMM to generate the coefficient matrix is called as Fast Multipole Boundary Element Method (FMBEM). A detailed and comprehensible explanation of FMBEM theory with its applications to various areas can be found in [**15**].

The key point in FMBEM is to eliminate the dependency of fundamental solution on both source and field points by writing it as a series expansion including two terms one is dependent on the source point and the other is on the field point. This can be expressed as follows;

$$G(P,Q) = \sum_{k} G_k^P(P,Q_c)\, G_k^Q(Q,Q_c) \qquad (2.36)$$

with $G(P,Q)$ being the fundamental solution to the Laplace's equation and Q_c the expansion center close to the field point Q. Writing the fundamental solution in this form, one has the chance to treat all the field points first and then transfer the results to the source points. The difference between the classical BEM approach and the FMBEM approach is explained schematically in Figure 4.

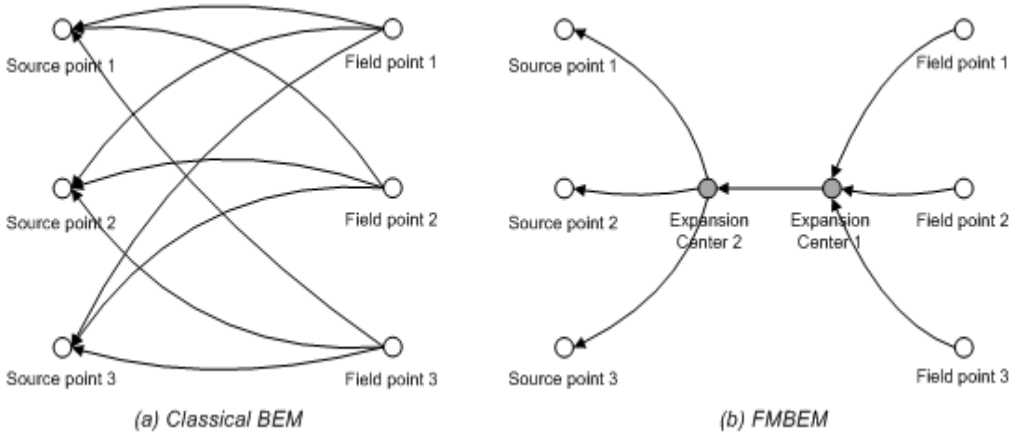

Figure 4 – Comparison between the classical BEM and FMBEM approach

Here in Figure 4-b, the movement from the "field points" to the first "expansion center" is called "moment expansion", the one from the first expansion center to the second is called "moment-to-local translation", and the one from the second expansion center to the source points is called "local expansion". Details of these expressions are given in the following sections.

2.2.1 Formulation for FMBEM

The detailed derivation of the formulation of FMBEM for 2D problems can be found in [**15**] and derivation for 3D problems together with the derivation of the solid harmonic functions are given in [**16**]. In order not to lose the concentration on the algorithm, only the resultant equations of FMBEM are given in this study with brief explanations.

For 3D problems, the fundamental solution of Laplace's equation can be written in the form of series of multipole expansion. For the expansion in 3D, solid harmonic functions are used;

$$G(P,Q) = \frac{1}{4\pi r} = \frac{1}{4\pi}\sum_{n=0}^{\infty}\sum_{m=-n}^{n} \overline{S_{n,m}}(P-Q_c)R_{n,m}(Q-Q_c) \qquad (2.37)$$
$$|Q-Q_c| < |P-Q_c|$$

The derivative of the fundamental solution with respect to n may also be written as;

$$\frac{\partial G(P,Q)}{\partial n} = \frac{1}{4\pi}\sum_{n=0}^{\infty}\sum_{m=-n}^{n} \overline{S_{n,m}}(P-Q_c)\frac{\partial R_{n,m}(Q-Q_c)}{\partial n} \tag{2.38}$$

$$|Q-Q_c| < |P-Q_c|$$

Here one should note that the field and source points are separated in the series expansion. In the above equation $R_{n,m}$ and $S_{n,m}$ are solid harmonic functions which are expanded as follows;

$$R_{n,m}(x) = \frac{1}{(n+m)!}P_n^m(cos\theta)e^{im\phi}r^n \tag{2.39}$$

$$S_{n,m}(x) = (n-m)!\,P_n^m(cos\theta)e^{im\phi}\frac{1}{r^{n+1}} \tag{2.40}$$

Where $x = (r,\theta,\phi)$ in spherical coordinates and P_n^m is the associated Legendre function which is given as follows;

$$P_n^m(x) = (1-x^2)^{\frac{m}{2}}\frac{d^m P_n(x)}{dx^m} \tag{2.41}$$

In order to calculate solid harmonic functions, recurrence formulae are used;

Recurrence Formulae for $R_{n,m}$

$$\begin{aligned} R_{0,0} &= 1 \\ R_{n+1,n+1} &= \frac{x_1+ix_2}{2(n+1)}R_{n,n} \\ R_{n,m} &= \frac{(2n-1)R_{n-1,m}-r^2R_{n-2,n}}{(n+m)(n-m)} \end{aligned} \tag{2.42}$$

Where

$$x = (x_1, x_2, x_3)$$

For negative "m" values;

$$R_{n,-m}(x) = (-1)^m \overline{R_{n,m}}(x) \tag{2.43}$$

Recurrence Formulae for $S_{n,m}$

$$\begin{aligned} S_{0,0} &= \frac{1}{r} \\ S_{n+1,n+1} &= \frac{(2n+1)(x_1 + ix_{2)}}{r^2} S_{n,n} \\ S_{n,m} &= \frac{(2n-1)x_3 S_{n-1,m}}{r^2} \\ &- \frac{-(n-1+m)(n-1-m)S_{n-2,m}}{r^2} \end{aligned} \tag{2.44}$$

For negative "m" values;

$$S_{n,-m}(x) = (-1)^m \overline{S_{n,m}}(x) \tag{2.45}$$

Derivatives of $R_{n,m}$ & $S_{n,m}$

For $R_{n,m}$;

$$\begin{aligned} \frac{\partial R_{n,m}}{\partial x_1} &= \frac{1}{2}\left(R_{n-1,m-1} - R_{n-1,m+1}\right) \\ \frac{\partial R_{n,m}}{\partial x_2} &= \frac{i}{2}\left(R_{n-1,m-1} + R_{n-1,m+1}\right) \\ \frac{\partial R_{n,m}}{\partial x_3} &= R_{n-1,m} \end{aligned} \tag{2.46}$$

For $S_{n,m}$;

$$\frac{\partial S_{n,m}}{\partial x_1} = \frac{1}{2}\left(S_{n+1,m-1} - S_{n+1,m+1}\right) \tag{2.47}$$

$$\frac{\partial S_{n,m}}{\partial x_2} = \frac{i}{2}\left(S_{n+1,m-1} - S_{n+1,m+1}\right)$$
$$\frac{\partial S_{n,m}}{\partial x_3} = -S_{n+1,m}$$

2.2.2 Expansion and Translation Equations

As stated before, separation of the source and the field points are achieved through multipole expansions around expansion centers and translation of these expansion centers. The first thing to investigate in this issue is "moment expansion" (ME). Implementing multipole expansion of the fundamental solution into the first integrand function in the potential equation, one obtains;

$$\int_{S'} G(P,Q)(\sigma \cdot n)\, dS = \frac{1}{4\pi}\sum_{n=0}^{\infty}\sum_{m=-n}^{n} \overline{S_{n,m}}(P - Q_c) M_{n,m}(Q_c) \qquad (2.48)$$
$$|Q - Q_c| < |P - Q_c|$$

Where S' corresponds to the portion of the domain boundary which satisfies the condition $|Q - Q_c| < |P - Q_c|$. And similarly the second term in the potential equation can be written as;

$$\int_{S'} \frac{\partial G(P,Q)}{\partial n}\mu\, dS = \frac{1}{4\pi}\sum_{n=0}^{\infty}\sum_{m=-n}^{n} \overline{S_{n,m}}(P - Q_c) \widetilde{M}_{n,m}(Q_c) \qquad (2.49)$$
$$|Q - Q_c| < |P - Q_c|$$

Where the terms $M_{n,m}$ and $\widetilde{M}_{n,m}$ are called "*multipole moments*" and are expanded as;

$$M_{n,m}(Q_c) = \int_{S'} R_{n,m}(Q - Q_c)(\sigma \cdot \boldsymbol{n}) dS \qquad (2.50)$$

$$\widetilde{M}_{n,m}(Q_c) = \int_{S'} \frac{\partial R_{n,m}(Q - Q_c)}{\partial \boldsymbol{n}} \mu dS \qquad (2.51)$$

Here the point Q_c is the expansion center. One can carry the expansion center to another point without re-calculating the moments. This process is called "*moment-to-moment translation*" (M2L) and the equation for the process is as follows;

$$M_{n,m}(Q_{c'}) = \sum_{n'=0}^{n} \sum_{m'=-n'}^{n'} R_{n',m'}(P - Q_{c'}) M_{n-n',m-m'}(Q_c) \qquad (2.52)$$

The same equation is also valid for $\widetilde{M}_{n,m}$. This completes the expansion process near the field points. The next step is to obtain a similar expansion near the source points using the moment expansions calculated. This expansion is called "*local expansion*" (LE) and formulated as follows;

$$\int_{S'} G(P,Q)(\sigma \cdot \boldsymbol{n})\, dS = \frac{1}{4\pi} \sum_{n=0}^{\infty} \sum_{m=-n}^{n} R_{n,m}(P - P_L) L_{n,m}(P_L) \qquad (2.53)$$

Where the "*local expansion coefficient*" $L_{n,m}$ is given as;

$$L_{n,m}(Q_L) = (-1)^n \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \overline{S_{n+n',m+m'}}(P_L - Q_c) M_{n',m'}(Q_c) \qquad (2.54)$$
$$|P - P_L| < |Q_c - P_L|$$

where the process is called "*moment-to-local* translation" (M2L). Similar to the moment expansion case, the expansion center can also be moved to another point in local expansion process. The equation for this "*local-to-local translation*" (L2L) is given as follows;

$$L_{n,m}(Q_{L'}) = \sum_{n'=n}^{\infty} \sum_{m=-n'}^{n'} R_{n'-n,m'-m}(P_{L'} - P_L) L_{n',m'}(P_L) \qquad (2.55)$$

It is possible to write the same equations for the second integrand function. The basic idea of the FMBEM theory is given in Figure 5 with two separated group of source and field points. However before going further to the analysis of a discretized

domain like in Figure 2, some definitions about the grouping of the elements in the domain should be given.

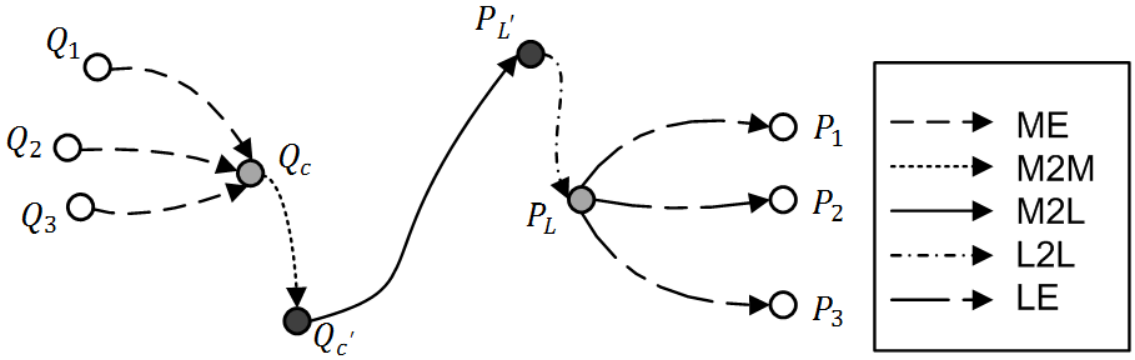

Figure 5 - Procedure for FMBEM Approach

2.3 GMRES Algorithm

The author followed the steps defined in the paper written by Saad and Schultz [**22**] in a straightforward manner for constructing the GMRES algorithm. Without giving all the details, the main parts of the algorithm repeated here. One may see the original paper for the complete process.

GMRES algorithm can be separated into two parts

- Obtaining an orthogonal basis using the Krylov subspace K_k where;

$$K_k \equiv span\{v_1, Av_1, \cdots, A^{k-1}v_1\} \tag{2.56}$$

  for the coefficient matrix A to be used on a linear system $Ax_k = f$.

- Finding a solution vector which minimizes the objective function defined as

$$J \equiv min\|f - Ax_k\| \tag{2.57}$$

which is nothing but the norm of the residual. For minimizing the objective function, "*QR factorization*" is applied to the Hessenberg Matrix which is equal to;

$$H_k \equiv V_k^T A V_k \tag{2.58}$$

with V_k being a matrix whose columns form an orthonormal basis $\{v_1, v_2, \cdots, v_k\}$ for the Krylov subspace.

The solution vector x_k is written in the form;

$$x_k = x_0 + z \tag{2.59}$$

with x_0 being the initial estimate for the solution vector and $z \equiv V_k y_K$. Since the initial estimate is constant and V_k depends on the Krylov subspace only, it is aimed to find vector y_k which minimizes the objective function. The orthogonalization algorithm is give as follows;

1. Selecting the initial estimate, finding the first residual vector and defining the first vector of the orthonormal basis using this residual vector

$$r_0 = f - Ax_0 \;\; ; \quad v_1 = r_0 / \|r_0\|$$

2. Doing the iterations in order to complete the orthonormal basis as follows. For $j = 1, 2, \cdots, k$

$$\begin{aligned} h_{i,j} &= \left(Av_j, v_i\right), \;\; i = 1, 2, \cdots, j \\ \hat{v}_{j+1} &= Av_j - \sum_{i=1}^{j} h_{i,j} v_j \end{aligned}$$

$$
\begin{aligned}
h_{j+1,j} &= \|\hat{v}_{j+1}\| \\
v_{j+1} &= \hat{v}_{j+1}/h_{j+1,j}
\end{aligned}
$$

3. Obtaining the solution using the following equation;

$$x_k = x_0 + V_k y_k \tag{2.60}$$

As mentioned before, "*QR* factorization" is used for minimizing the objective function. An important relation used in the process is as follows;

$$AV_k = V_{k+1}\widehat{H}_k \tag{2.61}$$

where $\widehat{H}_k$ is the same with the matrix H_k with an extra row is added to the end whose all the elements are zero but $h_{k+1,k}$. After introducing the above equation, the following process is conducted for minimizing the residual norm;

$$
\begin{aligned}
J(y_k) &= \|f - A(x_0 + V_k y_k)\| \\
&= \|r_0 - AV_k y_k\| \\
&= \|\beta v_1 - V_{k+1}\widehat{H}_k y_k\|
\end{aligned}
$$

where $\beta = \|r_0\|$. One should note that the first term in the above equation can be written as $\beta v_1 = V_{k+1}\beta e$ where e is the first column of an identity matrix with dimensions $(k+1) \times (k+1)$. Then it can be written that;

$$J(y_k) = \|V_{k+1}(\beta e - \widehat{H}_k y_k)\|$$

Since V_{k+1} is positive definite, the objective function reduces to;

$$J(y_k) = \|\beta e - \widehat{H}_k y_k\| \tag{2.62}$$

At this point, "QR factorization" for the matrix $\widehat{H}_k$ so that the following equation is obtained;

$$Q_k \widehat{H}_k = R_k \tag{2.63}$$

where Q_k is a $(k+1) \times (k+1)$ unitary matrix and R_k is an upper triangular matrix with size $(k+1) \times k$ and a last row equal to zero. The matrices Q_k and R_k are obtained using Givens transformations and the details can be found in [**22**]. Since Q_k is unitary, the objective function can be written in the following form;

$$\begin{aligned} J(y_k) &= \left\| Q_k \beta e - Q_k \widehat{H}_k y_k \right\| \\ &= \left\| Q_k \beta e - Q_k \widehat{H}_k y_k \right\| \end{aligned} \tag{2.64}$$

where $g_k = Q_k \beta e$. Since the last row of R_k matrix is zero, the last element of $R_k y_k$ cannot be affected changing the vector y_k. However, the objective function can be minimized by making the remaining elements zero in $R_k y_k$. Let $\widehat{R}_k$ be a $k \times k$ matrix equal to the first k elements of R_k. Then the result of the following equation yields a y_k vector which minimizes the objective function;

$$\widehat{R}_k y_k = 0$$

Since $\widehat{R}_k$ is an upper triangular matrix, y_k can be calculated easily. Afterwards the solution is obtained via the following equation;

$$x_k = x_0 + V_k y_k$$

For GMRES algorithm with incomplete orthogonalization, the process is repeated up to "*m*" terms. A solution is obtained as explained above according to the first *m* vectors. Then the process is started again using the solution obtained as the first estimate. And it goes on until the residual becomes smaller than the convergence limit.

CHAPTER 3

DETAILS OF THE SOLVER

In this section, the algorithms established for implementing the FMBEM to the solver are explained. First a general outline of the code is given through a flow chart. Afterwards, the parts mentioned in the flow chart are investigated in detail.

3.1 Code Outline

When writing a code, there is always a compromise between speed and the memory allocation. As a general approach, the author gave more weight to the former while writing this code, since it is aimed to obtain a solver which converges to the solution as fast as possible. Another reason for this approach is that the allocated memory does not exceed several hundreds of megabytes even the domain is composed of hundreds of thousands of elements. The allocated memory, on the other hand, is tried to be kept at minimum by not storing the empty blocks in any array used throughout the solution. The outline of the code is given in Figure 6. The wake relaxation process is started after a certain convergence is reached. This measure is taken in order to prevent unrealistic deformations on the wake surface. Together with the wake relaxation, IPKC scheme is started to be applied also.

3.2 Meshing Geometry and Reading Geometry Data

The author used the GAMBIT software for meshing the geometry. And its generic output format (.neu) is sent to the solver as the input file. The .neu file format first lists the node coordinates and then there comes the elements together with the

connectivity data. The details of the .neu format are given in [**23**]. Together with the mesh data, the trailing edge (TE) data should be introduced to the code since it is not easy for the code to detect TE by itself. As GAMBIT creates nodes on edges in sequence, it is enough to give the indices of the first and the last node on the edge together with the starting and ending indices of the nodes between them.

3.3 Algorithm for Oct-tree Structure

For applying FMBEM algorithm on the calculations related to far elements, it is required to identify the spatial relationship between the elements first. To achieve this, an oct-tree structure is constructed in the 3D domain.

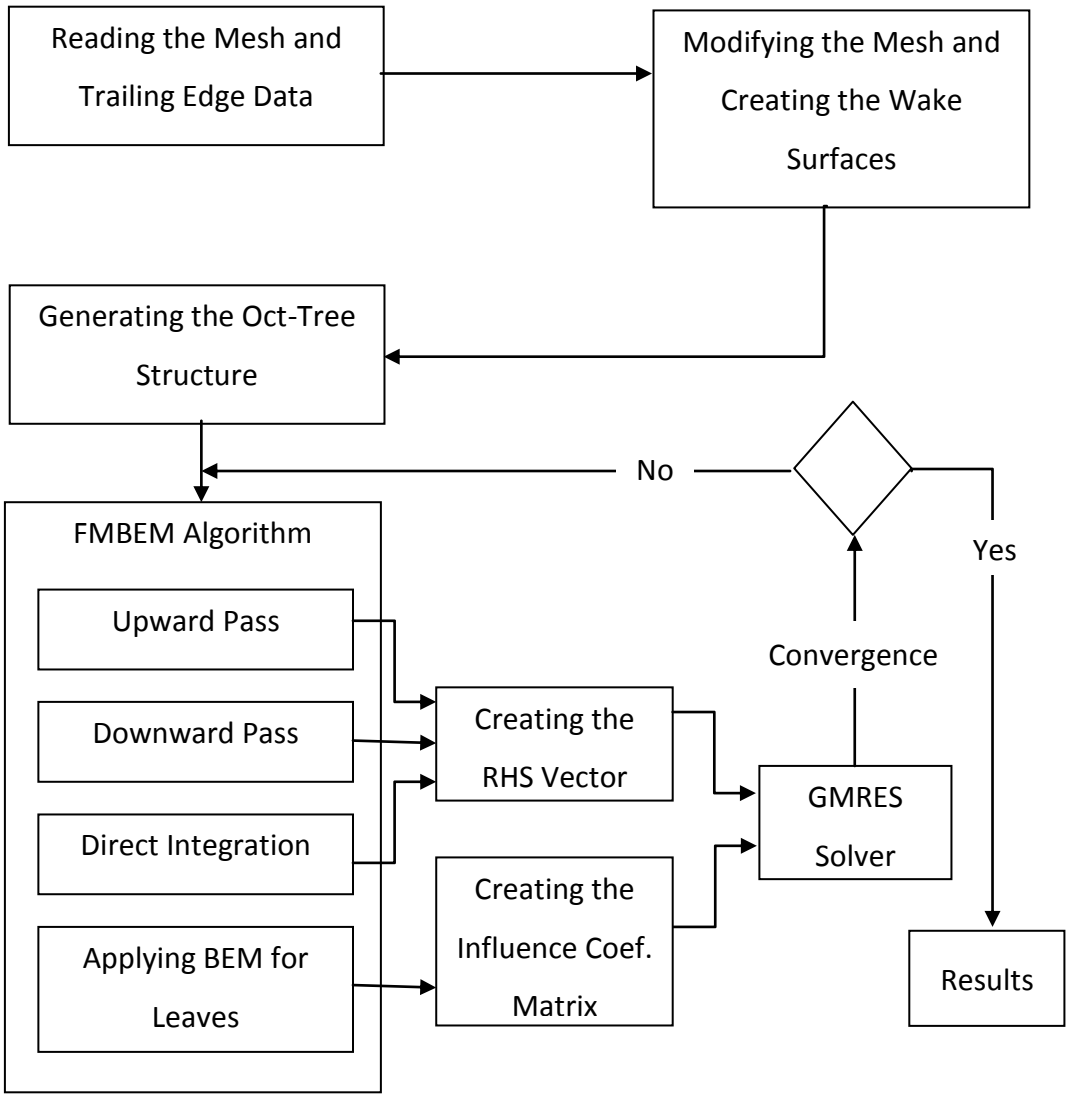

Figure 6 - Flow Chart Showing the Code Outline

The idea is basically to divide a volume (can be defined as a “cell”) which contains enough elements (greater than a pre-defined limit) into 8 children and then to divide each child in a similar manner again if the number of elements it borders is greater than the limit specified and to go on this procedure until there exists no cells containing elements more than the limit. A schematic representation for the 2D case (quad-tree structure) where the cell limit is specified as 5 can be seen in Figure 7 and Figure 8.

Each division is associated with a level, such that the big cell is thought to be at level zero, the cells in the upper side of Figure 8 are at level 1 and the cells which are magnified aside in the lower side of Figure 8 are at level 4.

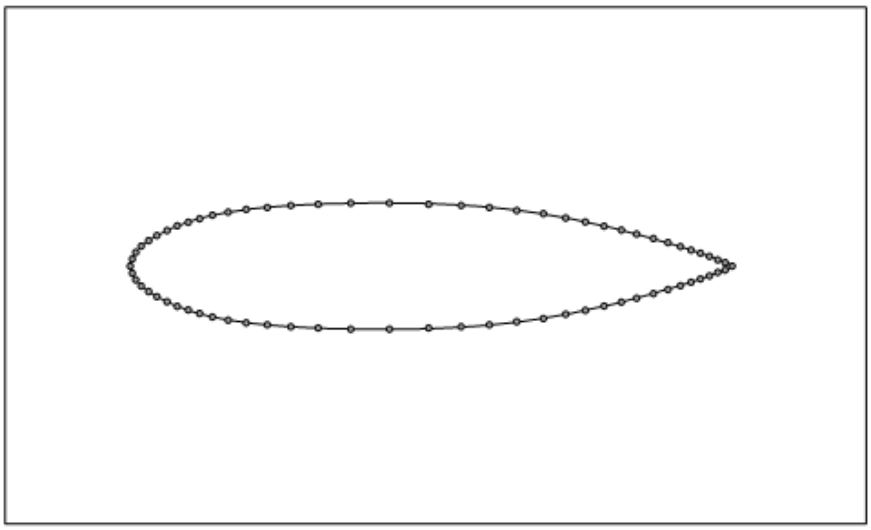

Figure 7 - The Discretized Domain and the Big Cell Containing the Whole Domain

Use of Pointers for the Construction of Oct-tree Structure

The total number of leaf cells in an oct-tree structure cannot be predicted without knowing about the homogeneity of the elements in the domain. Hence the memory required to store the oct-tree structure cannot be pre-allocated if the domain is discretized randomly. In order to jump over this problem, a method which allocates required amount of memory when it is required during the construction process should be applied. For this purpose the properties named “pointer” and “derived data type” in FORTRAN 95 are used.

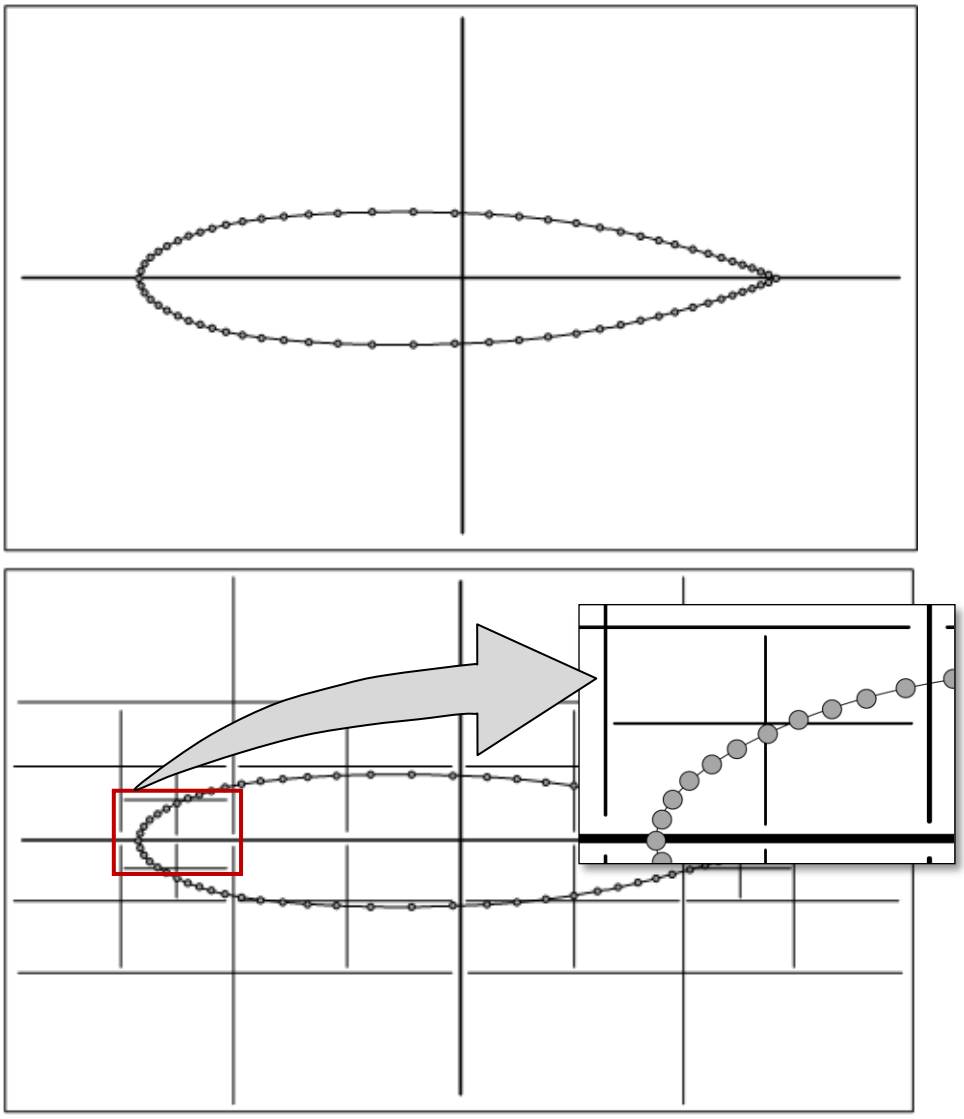

Figure 8 - Division of Cells to Construct the Tree Structure (The above depicts the first division of the big cell and the below shows the finalized structure with cell limit equal to 5)

A pointer is, as the name implies, a scalar (it may also be an array if defined so) that points a location in the memory when it is allocated. The memory block pointed by a pointer can be deallocated or the pointer can be set to point another block in the run-time. Also a pointer may be linked to another pointer so that they both point the same memory block.

The derived data type property is used to create a structure which may contain components of different types in it. To explain better, it is good to use an example which also summarizes the oct-tree construction. First a type named "Tree" is defined with the components "elements", "isLeaf", "child" as follows;

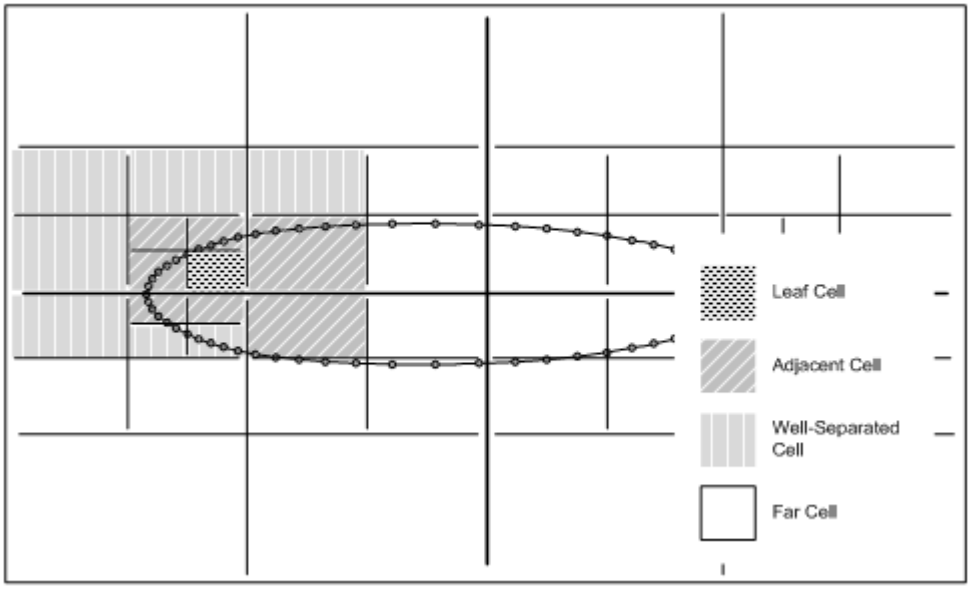

Figure 9 - Spatial Relationship between the Cells

```
Type Tree
     Integer                :: elements
     Logical                :: isLeaf
     Type(tree), pointer    :: Child
End type tree
```

Now when a variable named “Cell” is set as;

```
Type(Tree), pointer   :: Cell
```

it has the scalar component “elements”, let us say showing the total number of elements in the cell, and the logical component “isLeaf”, defining whether the cell is a leaf cell or not. Since the variable “Cell” is defined as a pointer, it has to be allocated before being used. After allocation, its components may be assigned using “%” as a separator;

```
allocate(Cell)
Cell%elements    = 15
Cell%isLeaf      = .False.
```

Coming to the critical part, structure “Cell” is also allowed to have a component of a derived type. In this example, the pointer named “Child” of type tree is also a component the pointer cell which is also of type tree. And the component “Child” has, by definition, all the components defined in the type “tree”. This property builds up the key point in the use of pointers for constructing the oct-tree structure. This allows a child cell to be allocated as it is required. Again after being allocated once, the component “Child” may be assigned similarly;

```
allocate(Cell%Child)
Cell%Child%elements   = 5
Cell%Child%isLeaf     = .True.
```

Here “Child” is defined as scalar but it could also be assigned as an array of pointers. Then each element of that pointer array would contain all the components defined in the derived type “Tree”. For the oct-tree structure, the rank of the element “Child” should be set as “8” since each cell contains eight children in oct-tree structure. If one wants to assign the components of the “i^{th}” cell, the following has to be done;

```
allocate(Cell%Child(8))
Cell%Child(i)%elements     = 5
Cell%Child(i)%isLeaf  = .True.
```

In order to make the cells reachable when it is needed, each cell is given an id. The id of the big cell containing the whole domain is set to “0”. Any child created at any level is given a sub-id from “1” to “8”. Accordingly, the equation used for finding the id of a newly created cell is;

$$ChildCell\_ID = FatherCell\_ID \times 10 \; + \; ChildCell\_subID$$

For instance, the id of the first child of the big cell is “1” ($0 \times 10 + 1 = 1$) while the id of the third child of the first child is “13” ($1 \times 10 + 3 = 13$).

For reaching a cell having an id “38582”, the integer id is converted to an integer array (it is named in the code as “”*Digits_Array*”) of rank “5” such that each element in the array corresponds to a digit in the id sequentially. Then using a pointer pointing the top of the tree (named as “*Top*”) and a dummy pointer to be carried down along the tree (named “*Cell_Dummy*”), the cell is reached through the following loop;

```
Cell_Dummy => Top
Rank = size(Digits_Array)
Do i = 1, Rank
     Cell_Dummy  => Cell_Dummy%Child(Digits_Array(i))
End do
```

In the above code the symbol “=>” is used for making the first pointer to point the same thing with the second one.

In order to store the data indicating which element is contained by which cell, the following method is used. The element ids are kept in a list together with the corresponding connectivity data named as “*ElmntList*”. The distribution of the elements into the cells starts from level zero. The elements in the “*ElmntList*” are redistributed in order to form blocks such that each block stays within the borders of a cell at level 1. In other words, investigating Figure 10, the elements are redistributed such that the first “n” elements belong to the cell with id 1, the ones between the “$(n+1)^{th}$” and the “m^{th}” indices belong to the cell with id 2, and it goes on like that.

After creating these blocks, the starting and ending indices of each block are recorded in a component named “*i_StartEnd*” (which is a 1D integer array of rank 2, the first element for the starting index, and the second for the ending index) under the structure “cell” of type “tree”. This way the component “*i_StartEnd*” shows the first and the last element (hence the ones among the first and the last) contained by the cell. After redistributing all the elements in the “*ElmntList*” and recording the start-end indices of each block, construction of the tree structure for the zeroth level is

completed. For the next level, a similar process is conducted but instead of treating the whole "*ElmntList*", a single block corresponding to a child at level 1 is taken into consideration. This time, the elements in the selected block are redistributed to form smaller blocks. And start-end indices of each small block are recorded in the "*i_StartEnd*" components of the newly created children of the cell being investigated. Figure 10 illustrates this process schematically for a quad tree structure in 2D.

Since above figure describes a case in 2D, each cell has four children. Accordingly, "*i_StartEnd*" components for the pointer "Top" and its first child are set as follows;

```
Top%Child(1)%i_StartEnd = (/1, n/)
Top%Child(2)%i_StartEnd = (/n+1, m/)
Top%Child(3)%i_StartEnd = (/m+1, k/)
Top%Child(4)%i_StartEnd = (/k+1, N/)
```

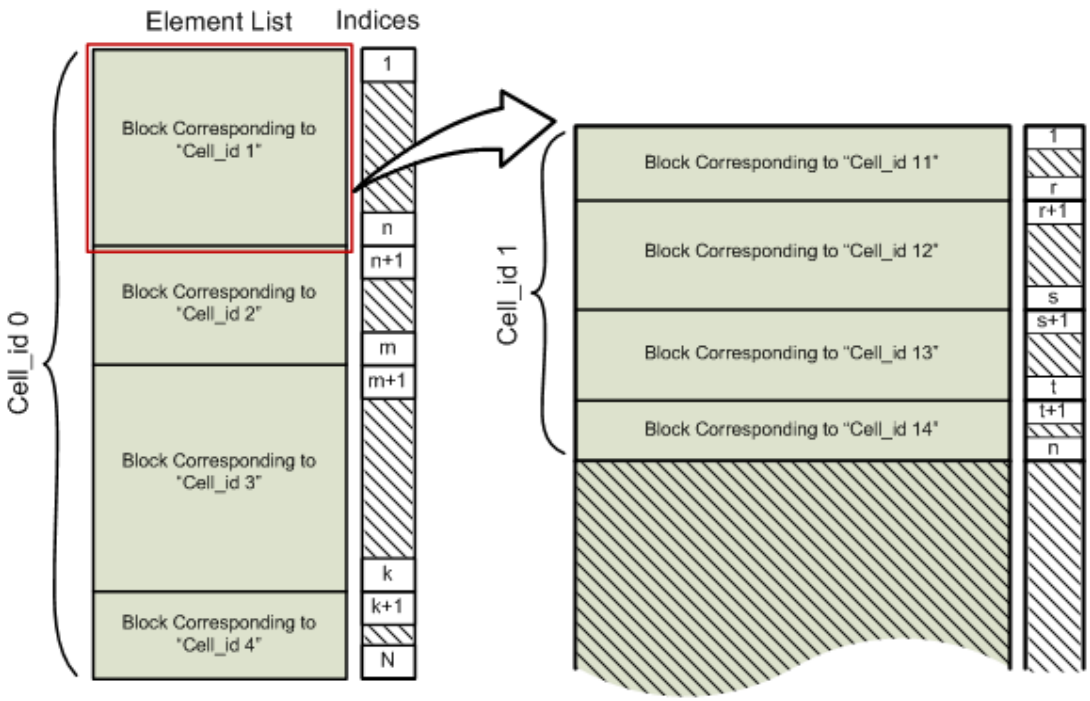

Figure 10 - Redistribution of the Elements in the Element List

```
Top%Child(1)%Child(1)%i_StartEnd = (/1, p/)
Top%Child(1)%Child(2)%i_StartEnd = (/p+1, r/)
Top%Child(1)%Child(3)%i_StartEnd = (/r+1, s/)
Top%Child(1)%Child(4)%i_StartEnd = (/s+1, n/)
```

This process goes on until there exist no cell containing elements more than the limit specified and afterwards the construction process of the oct-tree structure is completed. In the type “tree”, it is assigned the components for keeping the data of coordinates of the cell center, the starting / ending indices of the first / last element contained in the cell, the ids of the adjacent cells, the moments and local expansion coefficients belonging to the cell, the flag indicating whether the cell is a leaf or not. Also the components “Child” and “Father” of type tree are contained for indicating the children and the father of the cell.

3.4 FMBEM Algorithm

After introducing the tools that FMBEM uses now the rules and procedures of FMBEM can be explained. The items to be performed sequentially are written below;

1. Construct the Oct-Tree structure regarding the limit to the number of elements that a leaf cell can contain.
2. Then find the adjacent, well separated and far cells for that leaf since each group of cells is treated differently. Adjacents of a cell is found by using the following method. First the possible center for an adjacent cell is calculated by taking the center of the cell being investigated as reference. A dummy pointer is set to the top of the tree structure and moved towards the possible adjacent cell center level by level by, let us call it as such, “stepping” onto the existing cell centers. Each move corresponds to a digit in the adjacent cell id. This moving down in the tree structure is repeated until the adjacent cell center is reached. If an empty cell is encountered during this movement, the process is stopped since that means such an adjacent does not exist on the tree.

3. As mentioned before, FMBEM process requires an iterative solver to solve the linear system created. Within this thesis work, a GMRES solver written by the author is used for this purpose. Since GMRES method requires a preconditioned solution vector, the collocation points in the domain should be preconditioned. This is achieved by treating each leaf separately (as if the leaf itself constitutes a stand-alone system) and obtaining a solution using conventional BEM together with "*LU Decomposition*" method for solving the linear set.
4. Calculate multipole moments at each leaf taking the leaf center as the expansion point.
5. Translate the moments calculated at the leaf centers to the centers of the father cells up to level "2" using the M2M translations. According to this, for a leaf cell having the id "436235", the centers of the following cells are used as expansion centers with the given order; "43623", "4362", "436", "43" (the author prefers to call these cells as the "*root cells*" of the given leaf). This process is called as "*upward pass*" and shown schematically in Figure 11.
6. Take a leaf cell into consideration and create the influence coefficient matrix with the elements inside the cell using conventional BEM (similar to the preconditioning process).
7. The total moment of a far cell at level "n" is translated to the root of the leaf at level "n" through M2L translation and then carried to the leaf through L2L translations.
8. For the moments of the cells in the interaction list (i.e. well-separated cells), M2L translations are performed directly to the leaf.
9. After the items "7" and "8" are completed, all the moments (namely, the total potential effect of all the cells excluding the adjacent ones) are collected at the leaf center. This process is called as "*downward pass*" (See Figure 12). Following is the expansion of these moments to the elements in the leaf through "*local expansions*" (just the opposite of "*moment expansions*") and construction of the right hand side vector of the linear system.
10. Finally the effect of the elements in the adjacent cells is added to the right hand side vector through direct integration using conventional BEM.

11. In order to obtain a solution for the whole domain the process explained between the steps “6” and “10” is performed for each leaf in the oct-tree structure.
12. This way construction of the linear system is completed and GMRES solver is used in order to obtain solution.
13. For the next iteration, the steps between 4 and 12 are repeated. The process goes on until the convergence limit is reached.

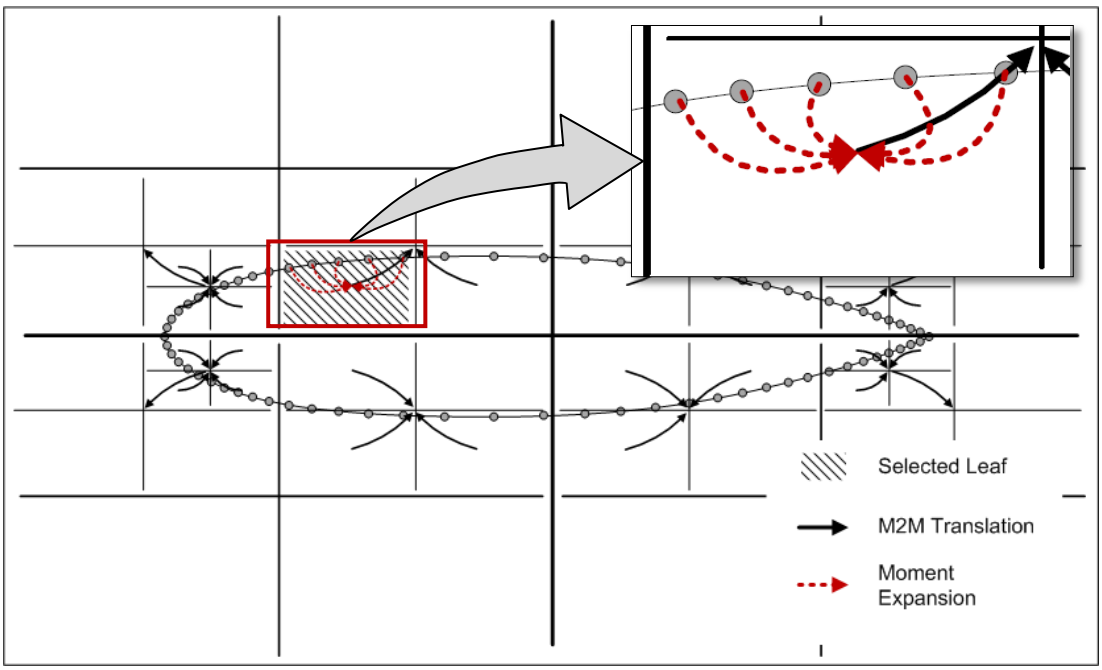

Figure 11 – Upward Pass Process

3.5 Calculating the Velocities from the Doublet Distribution

After obtaining the doublet distribution over the surface, it is required to calculate the velocities on the collocation points in order to obtain a pressure distribution over the boundaries of the domain. For calculating the velocity through the doublet distributions, the following equation is used;

$$V = \nabla\mu \tag{3.1}$$

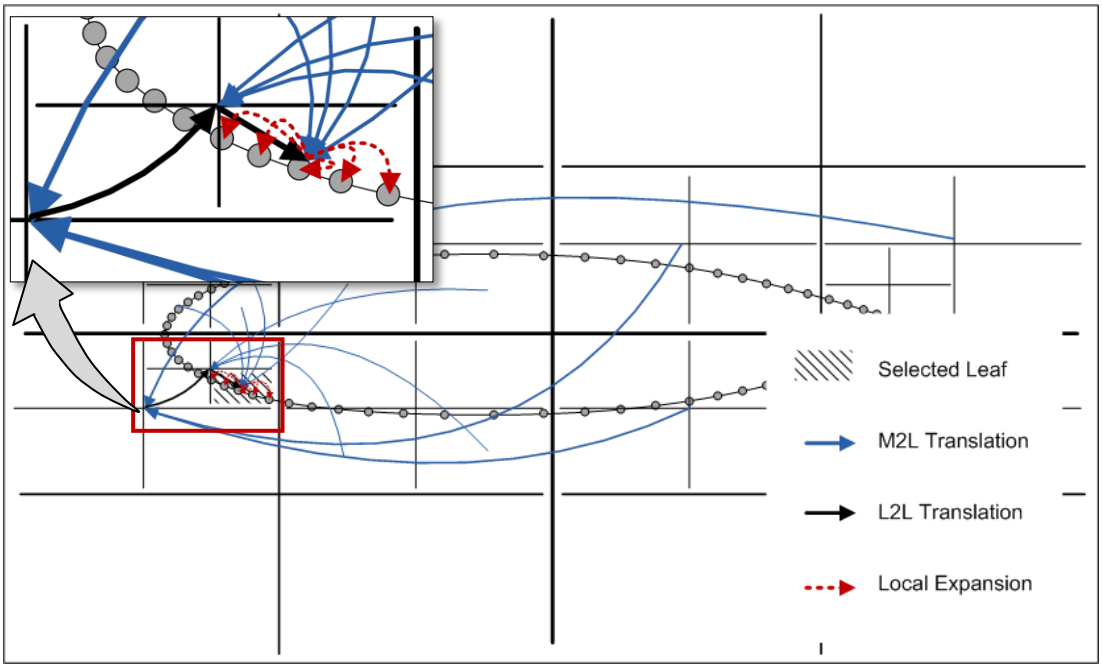

Figure 12 - Downward Pass Process

When using constant singularity distribution with a structured mesh, the velocity on an element is calculated through numerical differentiation which is done by investigating the change in the doublet strengths within the neighboring elements in spanwise and chordwise directions. However such an approach is not applicable when an unstructured mesh is used to represent the domain. This is because the mesh is, by definition, unstructured which means the neighboring elements are not aligned on the chordwise or the spanwise the directions (actually this is impossible when triangular elements are used for meshing) and the ".neu" format does not store the neighboring data for elements. Actually this difficulty in calculating the velocities is the main reason driving the author to use a linear strength distribution instead of a constant one. With a linear doublet strength distribution supplied, the author pursued the following method in order to calculate the velocities.

First the data that which node is used by how many elements is obtained and kept in a list through a routine. Then, an element is selected for investigation and each of its vertices (which are used as the collocation points) is treated separately. For the first vertex, a local coordinate system is attached on the element as seen in Figure 13. The unit vectors $\tilde{x}$ and $\tilde{y}$ which are defined in the global coordinate system, constitute the

principle axes of the local coordinate system together with the normal vector $\vec{n}$. The vectors $\vec{e}_1$ and $\vec{e}_2$ are used to define the two edges using the vertex being investigated. And $\vec{V}_{e_1}$, $\vec{V}_{e_2}$, $\vec{V}_x$ and $\vec{V}_y$ define the velocities along $\vec{e}_1$, $\vec{e}_2$, $\tilde{x}$ and $\tilde{y}$ directions respectively.

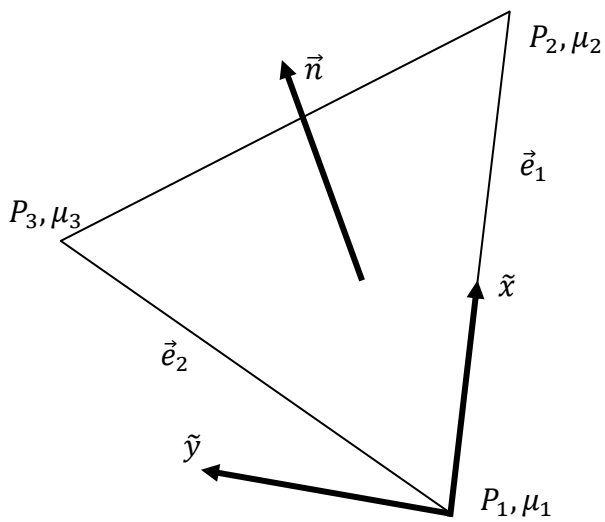

Figure 13 - Local Coordinate System Attached to the First Vertex

Regarding the above definitions;

$$
\begin{aligned}
\vec{V}_{e_1} &= \frac{(\mu_2 - \mu_1)}{|\vec{e}_1|}\vec{e}_1 \\
\vec{V}_{e_2} &= \frac{(\mu_3 - \mu_1)}{|\vec{e}_2|}\vec{e}_2 \\
\vec{V}_x &= \vec{V}_{e_1} \\
\vec{V}_y &= \frac{\vec{V}_{e_2} - \vec{V}_x(\vec{e}_2, \tilde{x})}{(\vec{e}_2, \tilde{y})} \\
\vec{V} &= (\vec{V}_x, \tilde{x})\tilde{x} + (\vec{V}_y, \tilde{y})\tilde{y}
\end{aligned}
$$

This result gives the velocity contribution of the selected element to the selected node. The same analysis is performed for all the elements using this node and the resulting velocities are averaged in order to find the velocity on the node. Actually, this method applied can be called as central differencing with treating $\tilde{x}$ and $\tilde{y}$ components of the local velocities together.

CHAPTER 4

RESULTS AND DISCUSSIONS

The verification of the solver is done first with comparing the results obtained for a rectangular wing using Van de Vooren (VDV) airfoil as profile with the analytical solution of the airfoil in 2D. The results for various symmetric airfoils at non-lifting flow conditions are also obtained compared with JavaFoil results. However, for lifting cases, it is not achieved to obtain a correct solution neither in conventional BEM, nor in FMBEM due to an error in implementation of the Kutta condition

Besides the verification of the FMBEM solver, some performance tests comparing the conventional BEM with FMBEM are also conducted. The comparison is made in terms of both the time required for the converged solution and the memory allocation. Details of all these applications are given in the following subsections.

All the analysis is performed on a laptop having an Intel i5 2.8 GHz processor, 3 MB L2 cache and 2GB RAM on it.

4.1 Case 1: Sphere

As the first test case, a sphere translating with a constant velocity in an infinite fluid is selected. The grid distribution over which the mesh converged results are obtained are depicted in Figure 14. And the mesh converged results of FMBEM and conventional BEM algorithms are compared with the analytical solutions of the potential flow equation over a sphere in Figure 15.

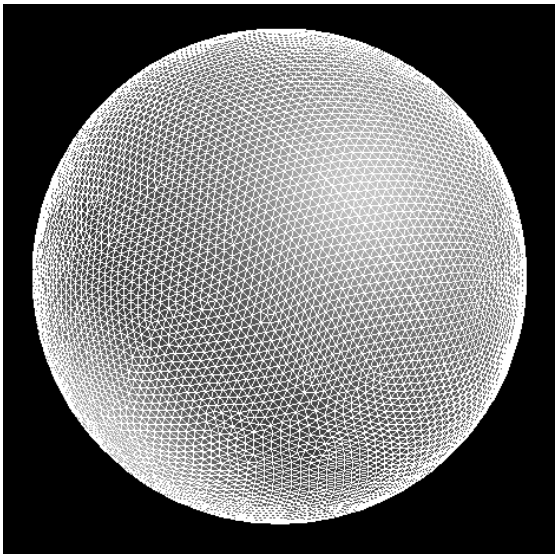

Figure 14 - Grid Distribution on Sphere

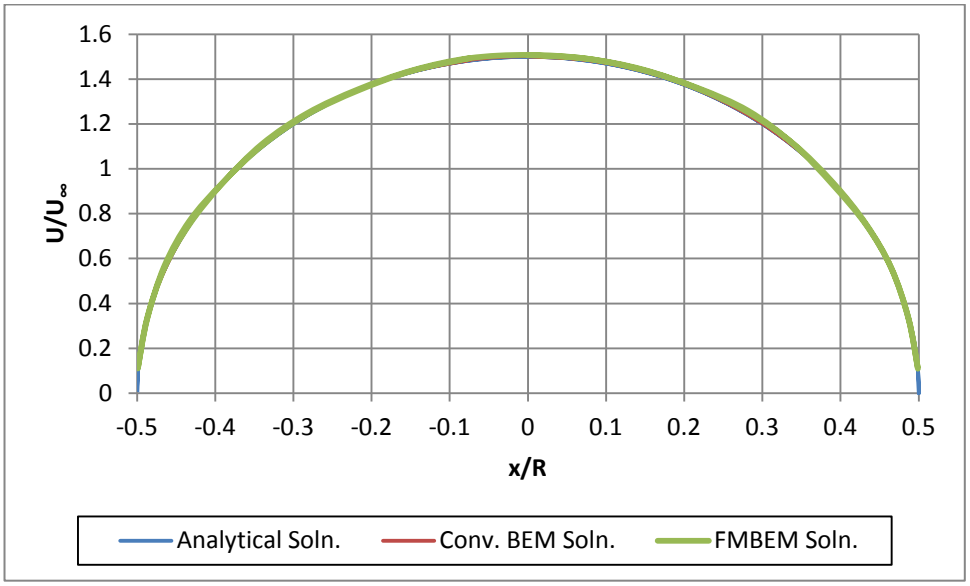

Figure 15 - Velocity Distribution over Sphere

As seen on the above plot, an almost perfect match is observed in both conventional BEM and FMBEM algorithms. The change in the total RMS error as the number of elements used in the solution varies can be investigated in Figure 16.

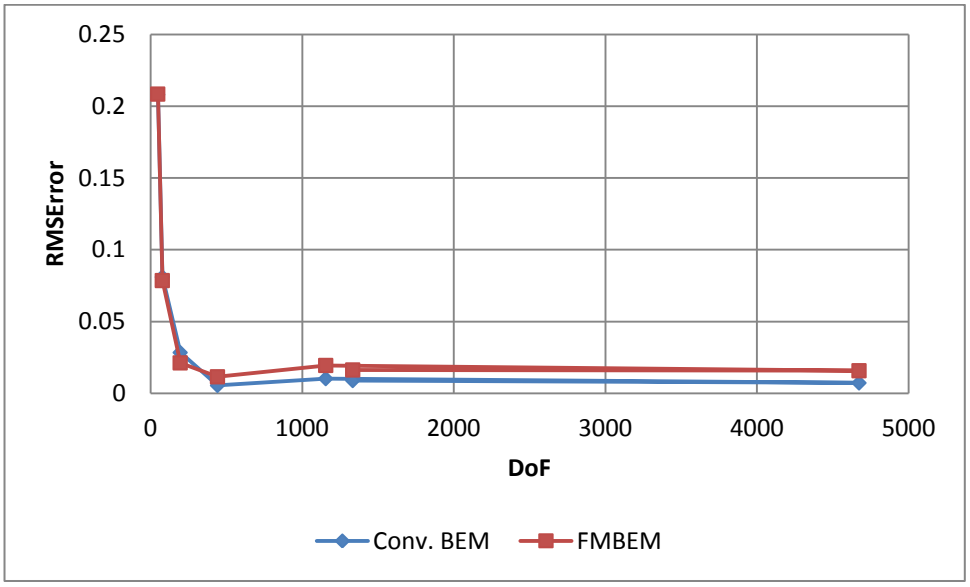

Figure 16 – Change in RMS Error with DoFs

One can see that conventional BEM yields better results with the same number of elements when compared to FMBEM. It is clear that FMBEM encounters loss of accuracy due to the omitted terms in the multipole expansion series.

4.2 Case 2: Rectangular Wing

The geometrical specifications of the wing used for this test are as follows;

- Profile : Van de Vooren airfoil
- Chord Length : 1m
- Span : 10 m
- Angle of Attack : 0 deg

In Figure 17, the grid distribution on which the mesh converged solutions for the Test Case 2 are obtained is shown. The velocity distributions are calculated on the symmetry axis. In Figure 18, the mesh converged results obtained from the conventional BEM solver and FMBEM solver are compared with the analytical solution of the airfoil in 2D. Both solvers use linear strength distributions on triangular unstructured grid elements.

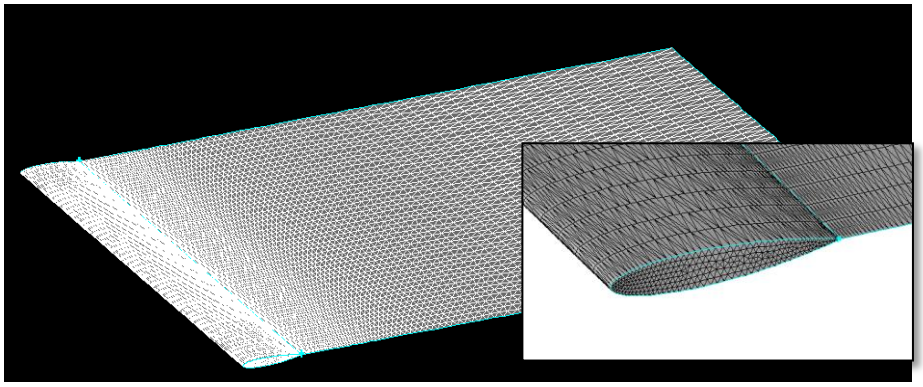

Figure 17 - Grid Distribution on Rectangular Wing

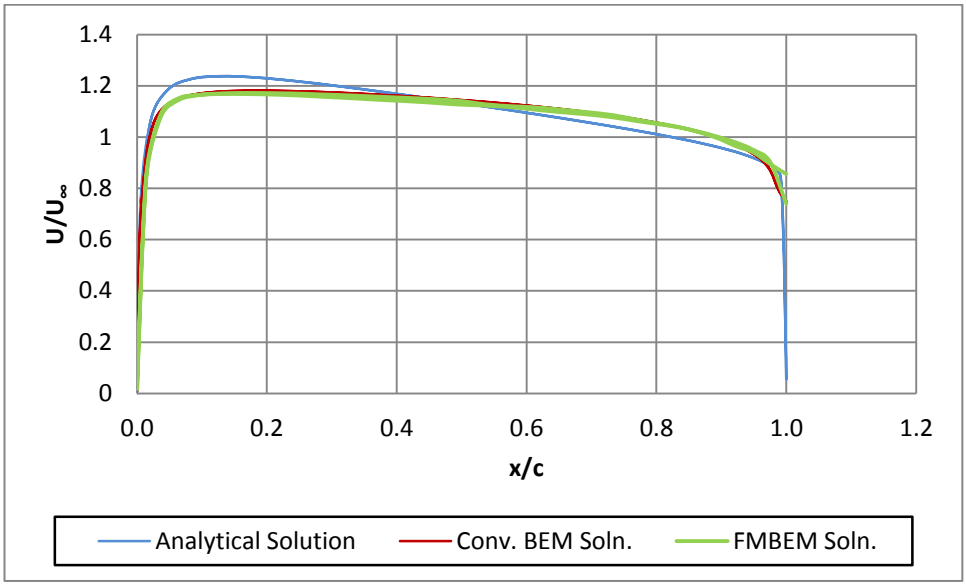

Figure 18 - Velocity Distribution on the Symmetry Axis for Van de Vooren Airfoil

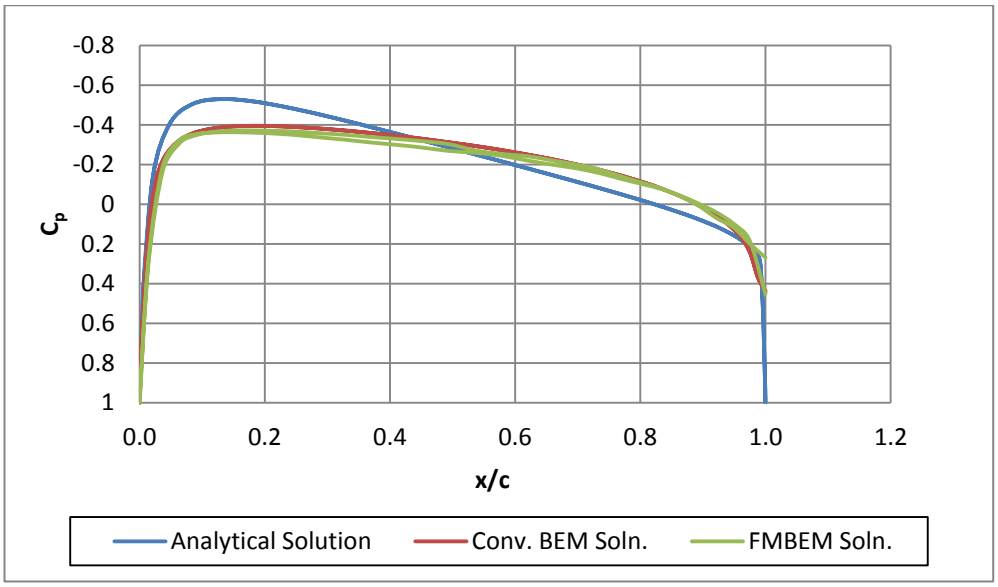

Figure 19 - Pressure Coefficient Distribution on the Symmetry Axis for Van de Vooren Airfoil

As it can be seen on the above graph, the solutions obtained from the conventional BEM and FMBEM are almost the same. Both BEM and FMBEM tends to estimate the peak velocity lower than the 2D solution.

Similar analysis is performed for NACA 0012 and Newton Circular airfoils and the results are depicted in the following figures.

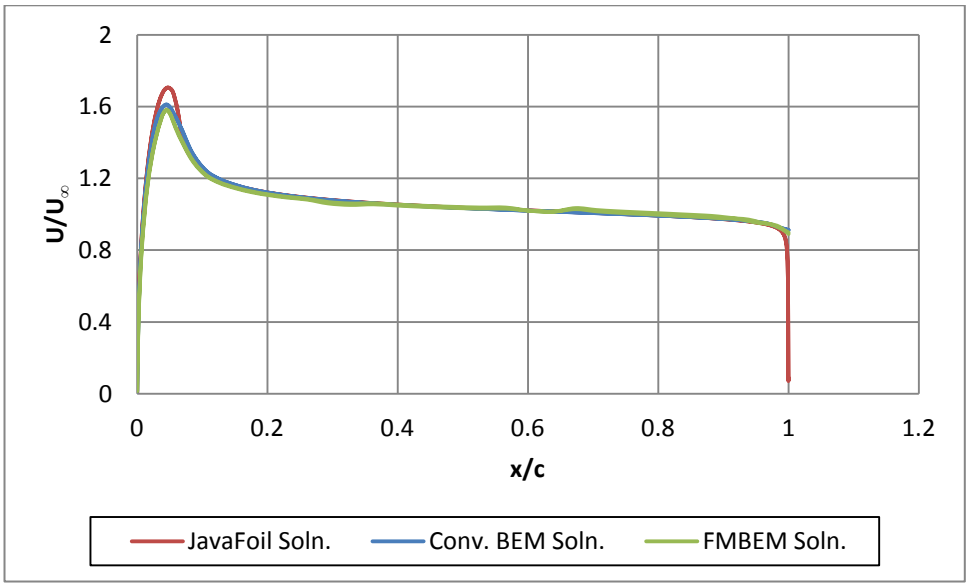

Figure 20 - Velocity Distribution on the Symmetry Axis for Newton Circular Airfoil

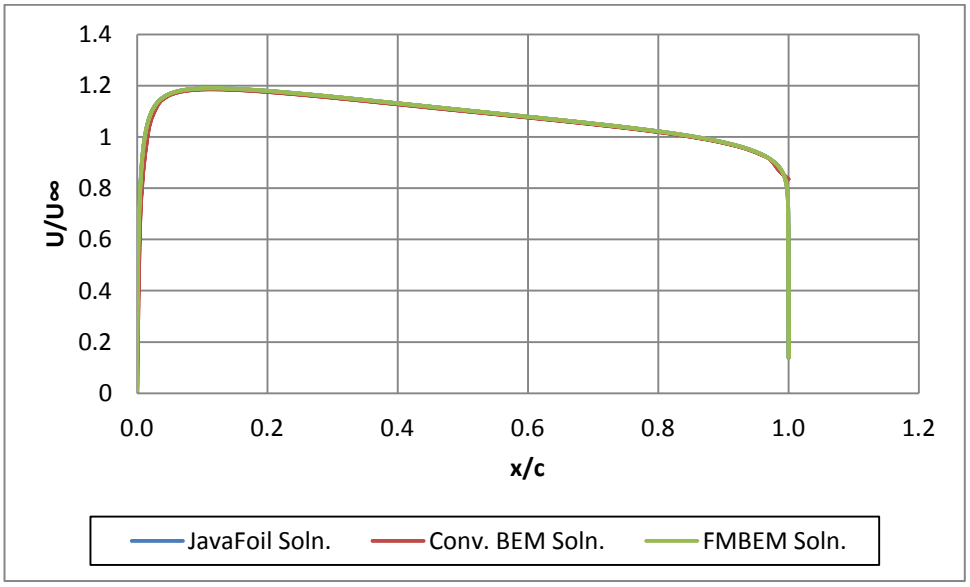

Figure 21 - Velocity Distribution on the Symmetry Axis for NACA0012

It is seen that a tendency to estimate the peak velocity lower than the 2D solution exists again for Newton Circular airfoil and a perfect match occurred for NACA0012. This assures that FMBEM solver estimates the potential distribution for non-lifting flows with a high accuracy.

4.3 Performance Comparison between Conventional BEM and FMBEM

As it is mentioned in the previous sections, for a BEM solver using constant singularity distributions, the number of operations to be done in order to obtain a converged solution is proportional to $O(N^3)$ with a direct solver such as Gauss Elimination method is used and $O(N)$ with an iterative solver such as GMRES solver. The same situation is valid in terms of the amount of memory required during the solution process. When a linear singularity distribution is used the ratio slightly differs since the number of elements is not equal to the number of unknowns. If N_U corresponds to the number of unknowns and N_E corresponds to the number of elements then the number of operations to create the influence coefficient matrix is defined with $O(N_U \times N_E)$ and one to solve this matrix is defined with $O(N_U^2)$. Fast Multipole Method claims to keep all these ratios at $O(N)$ and this is depicted via the following comparison. The solvers compared within each other are;

- Conventional BEM with a direct solver using *LU Decomposition* method
- Conventional BEM with GMRES solver
- FMBEM

Not all the cases were run with all three solvers since the amount of the required memory for BEM solvers tended to exceed the memory limit of test computer as the problem size increased up to a certain limit. The remaining cases were tested using the FMBEM solver only. The details of the test cases are given in Table 2 and the performance comparison which bases these test cases is given in Table 2. The comparison is made in terms of both the CPU time and the amount of memory required for a converged solution.

Table 2 – List of Test Cases Used in Performance Comparison

	Number of Unknowns	Number of Elements
Test Case 1	1036	3968
Test Case 2	2586	8928
Test Case 3	5528	13852
Test Case 4	9156	23056
Test Case 5	12750	33696
Test Case 6	17550	48536
Test Case 7	36750	112496

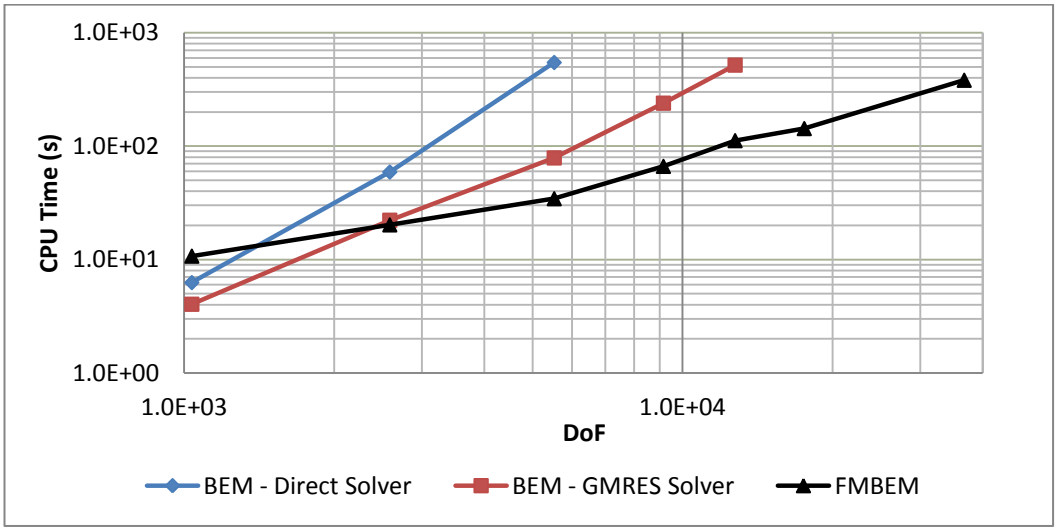

Figure 22 - CPU Time Comparisons for BEM and FMBEM

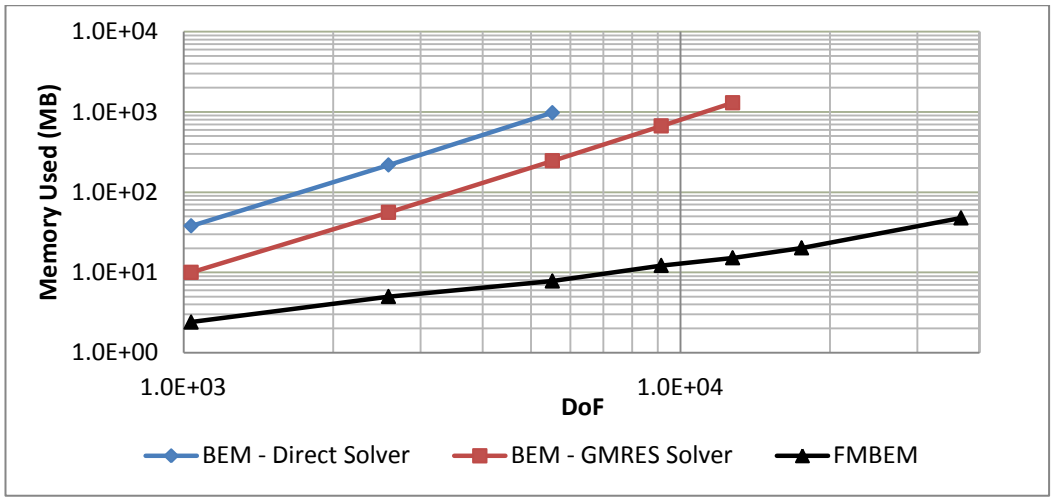

Figure 23 - Memory Requirement Comparison for BEM and FMBEM

Each test case corresponds to a point on the above plots. Only the first three test cases could have been applied to the conventional BEM with direct solver and the first five could have been applied to the one with GMRES solver due to the machine limits.

4.4 Convergence for the Number of Terms Used in Multipole Expansion

The number of terms used in the multipole expansions affects the number of operations performed in translation and expansion operations with $O(p^4)$ where p is the number of expansion terms. It is important to determine the number of terms to be used since unnecessary terms costs too much in terms of computation time while not affecting the accuracy significantly. In order to make the issue more clear, a study analyzing the change of the error and the computation time as p increases is conducted. The results of the analysis are given in Figure 24 and Figure 25.

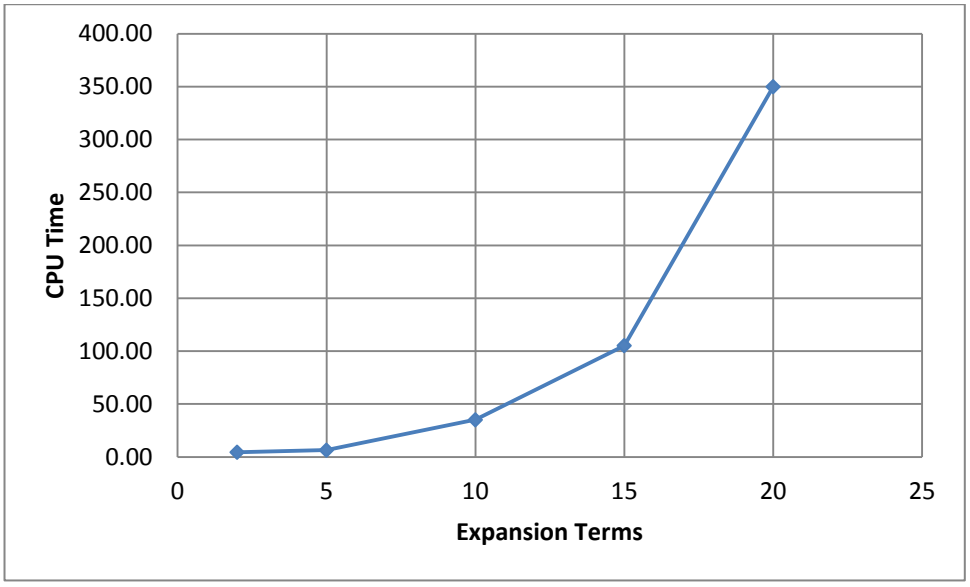

Figure 24 - Change of Computation Time with the Number of Expansion Terms

4.5 Effect of the Limit for Number of Elements Contained in a Cell

The limit for the number of elements contained in a cell is also an important parameter that affects the performance of the FMBEM solver. One should be careful when selecting it since use of extreme numbers may cause decrease in performance.

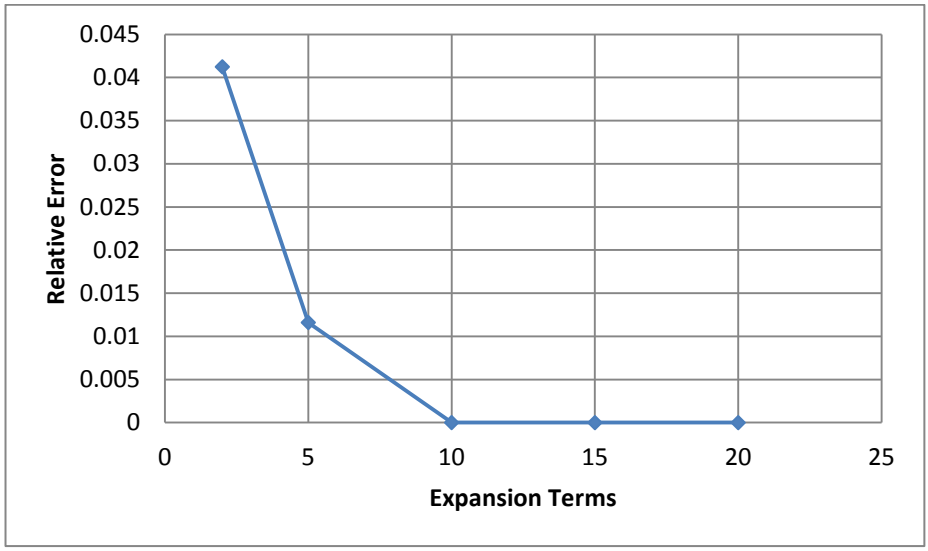

Figure 25 - Change in the Relative Error with the Number of Expansion Terms

Selecting the limit too high increases the number of elements contained in the leaf cells. And this situation causes increase in the computation time since the number of elements that are directly related to each other also increases. In other words, the solver tends to approach to the conventional BEM. However selection of a limit which is too small may lead to loss of accuracy or even instability. This affect can be seen in Figure 26 where the sudden change on the accuracy for low limit values indicates an instability in the solution. Hence it is required to find a reasonable limit for the number of elements contained in a leaf cell.

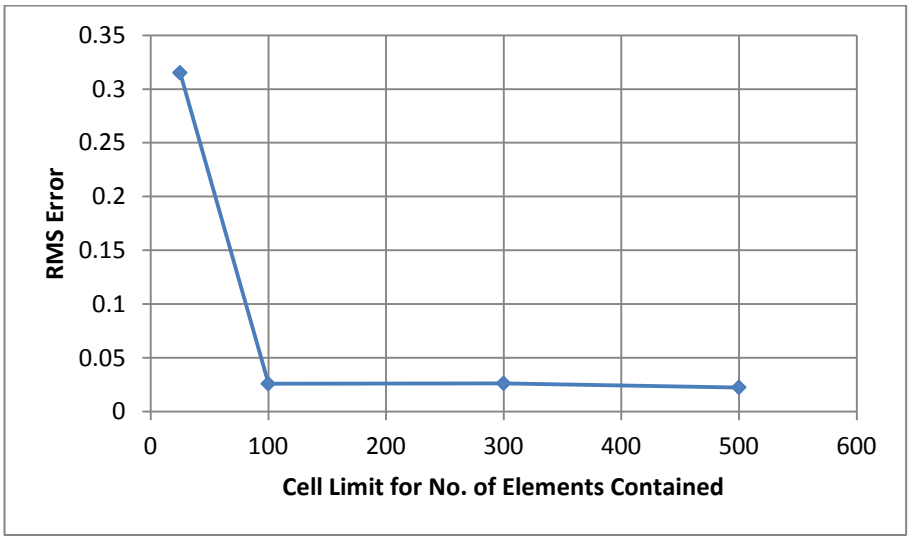

Figure 26 - Effect of Cell Limit on the Accuracy

4.6 Disadvantages of FMBEM Algorithm

Although use of FMBEM is advantageous in terms of computation time and memory issues for problems of large size, it has also some drawbacks. First of all, the author observed that if the total number of elements in the domain is about 5000 or less, the conventional BEM is better in terms of performance related issues when compared to FMBEM.

Besides FMBEM algorithm needs a homogenous mesh distribution which may be infeasible sometimes. One needs to keep successive ratio among the grid elements as small as possible since sudden changes in the mesh size cause instability. However keeping the successive ratio too small excessively and unnecessarily increases the total number of elements used to represent the geometry. And one may use a coarser grid distribution on the region where no sudden changes in the potentials are expected in conventional BEM while it may lead erroneous results in FMBEM. Use of “Adaptive FMBEM” may help to overcome this problem.

CHAPTER 5

CONCLUSIONS

It is aimed to develop an accurate and fast potential solver for non-lifting flows within this thesis study. The theoretical background related to the issue, the details of the solver and the results obtained using the solver are presented.

Besides FMBEM being a complex algorithm and hard to implement into panel method application, it has been seen that use of FMBEM is quite advantageous for large problems when considering the memory requirement and CPU time for converged solution. In order to make use of the solver practical, the author used a commercial mesh generator instead of developing one. And triangular grid elements are selected for being capable of meshing any geometry although the solver is not tested with complex geometries yet.

CHAPTER 6

FUTURE WORK

The inspiring idea behind this thesis work was to develop an analyzer to be used in performance and noise optimization of wind turbines. As a first step, it is aimed to developed an inviscid solver. The current solver can be applied for non-lifting flows only. In order to make lifting flows solvable an “Iterative Pressure Kutta Condition” (IPKC) is to be applied on the system. Moreover a wake relaxation scheme over a prescribed geometry is also aimed to be added. The details of these two schemes can be found in the next to subsections.

6.1 Shedding of Wake from a Prescribed Geometry

As it is mentioned in Potential Flow Theory, in order to make the solution of the potential equation unique, the doublet strengths on the wake lines and the shape of the wake surface has to be determined. The solution to the former is obtained through application of an iterative 3D version of the Kutta condition which tries to keep the pressure difference zero along the trailing edge. Coming to the latter, again an iterative method is applied to obtain the wake geometry.

Since the wake surface is infinitely thin, no pressure jump is allowed along it while there may occur a potential jump. This brings out the condition that the wake surface becomes tangential to the flow velocity behind the solid body. In order to satisfy this condition on the wake surface, a prescribed wake geometry is determined first and

then this geometry is deformed so that the wake lines are aligned with the velocities on the wake nodes. Deformation is done as follows;

- Flow velocities on the wake nodes are calculated.
- The components of the flow velocities which are normal to the wake lines are calculated. This is achieved by taking the projection of the velocities onto the wake lines and then subtracting these projections from the velocities.
- Wake nodes are moved along the normal components throughout an artificial time parameter. This artificial time parameter is calculated as follows;

$$t^* = \frac{l}{U_\infty} \tag{6.1}$$

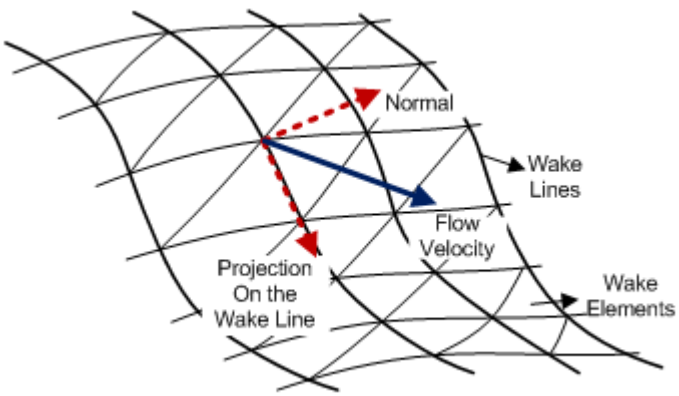

Figure 27 - Decomposition of flow velocities on the wake surface

The velocities on the wake nodes are calculated through the Biot-Savart law. However, if two nodes get too close to each other, the induced velocity tends to infinity unrealistically. In order to avoid these jumps in the velocities, a wake model is introduced to the system which limits the induced velocity when the distance is smaller than specified limit which is called the "*core radius*". "*Rankine model*" is selected for wake modeling and accordingly, the induced velocity distribution near a wake is given as follows;

The core radius "r" is determined through the equation;

$$r = \sqrt{4\nu t^*} \tag{6.2}$$

where ν is the kinematic viscosity. For details of vortex models and derivation of formulae, one may see [**24**], [**25**] and [**26**].

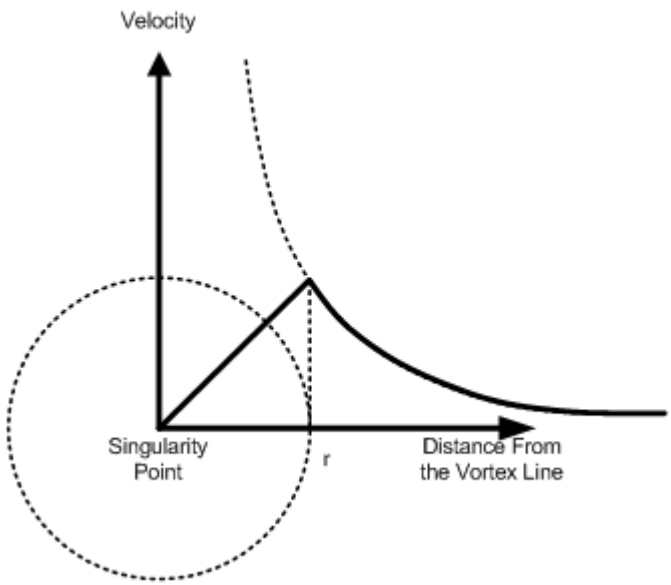

Figure 28 - The Rankine Vortex Model

Using Fast Multipole Method Approach for Wake Calculations

In order to make induced velocity calculations for wake elements possible using Fast Multipole Method (FMM) approach, a similar process is pursued. The intended method is summarized as follows.

First, the spatial relationship between the wake elements are defined just as it is done on the solid boundary elements. While calculating the induced velocity on a wake collocation point, the effect of closer elements is added using the Biot-Savart law.

For calculating the induced velocity due to the far elements, FMM approach is used. First the induced potentials of the far elements are calculated on the wake surface. Using these induced potentials the induced velocity component tangential to the wake surface is calculated. Afterwards, the collocation points on the wake surface are offset in both directions along the surface normals. For these upper and lower offset nodes, the induced potentials due to the far cells are calculated via local expansions. Taking the potential values on these offset nodes into consideration, the induced velocity component normal to the wake surface is calculated by numerical differentiation, namely central differencing.

Finally, the two velocities due to the far and close elements are added on to calculate the total velocity on the selected wake collocation point.

6.2 Iterative Pressure Kutta Condition

Morino's Kutta condition assumes a 2D flow along the chordwise direction which is not always the case for 3D problems, i.e. flow near the tip of a wing. Additionally the flow at the trailing edge may have a spanwise component also. This neglected component causes non-zero pressure differences at the trailing edge. In order to solve this problem, Kinnas et al. [**27**] applied scheme in order to set the pressure difference to zero numerically. An additional term is added to the equation used in the 2D version to add the effect of the neglected component of the flow velocity.

As it is mentioned before, Morino's Kutta condition assumes the following;

$$\mu^W = \mu^U - \mu^L \tag{6.3}$$

Since this equation is valid along the trailing edge, the following representation will be used;

$$\mu_j^W = \mu_j^U - \mu_j^L = \Delta\mu_j, \quad j = 1, 2, \dots, N_{TE} \tag{6.4}$$

where N_{TE} is the total number of nodes on the trailing edge. The condition for IPKC is that pressure difference between the upper and lower surfaces on the trailing edge is set to zero;

$$\left(\Delta C_p\right)_j = \left(C_p^U\right)_j - \left(C_p^L\right)_j = 0, \;\; j = 1,2, \dots, N_{TE} \tag{6.5}$$

In order to satisfy this condition, a multidimensional Newton-Raphson (N-R) scheme is applied. According to N-R, the equation giving the values for μ_j^W at (k+1)th iteration may be written as follows;

$$\sum_{j=1}^{N_{TE}} P_{i,j}\left(\Delta\mu_j^{(k+1)} - \Delta\mu_j^{(k)}\right) = -\left(\Delta C_p\right)_i^{(k)} \qquad i = 1,2, \dots, N_{TE} \tag{6.6}$$

where;

$$P_{i,j} = \frac{\partial\left(\Delta C_p\right)_i}{\partial\Delta\mu_j} \tag{6.7}$$

$\Delta\mu_j^{(1)}$ is obtained from the Morino's Kutta condition. The above equation may be rearranged in the matrix form as;

$$\left[P^{(k)}\right]\left[\delta(\Delta\mu)^{(k+1)}\right] = -\left[\left(\Delta C_p\right)^{(k)}\right]$$

where;

$$\delta\left(\Delta\mu_j\right)^{(k+1)} = \Delta\mu_j^{(k+1)} - \Delta\mu_j^{(k)} \tag{6.8}$$

After the above linear system is solved and the change in the wake strength is calculated, the whole domain is investigated again with these updated values and the doublet strengths are recalculated. Accordingly, the pressure difference at the trailing edge is also recalculated to obtain $\left(\Delta C_p\right)^{(k+1)}$. Then a new iteration begins with the RHS vector is updated.

Calculating the Jacobian matrix $P_{i,j}$ is not very straightforward. Hence a modified N-R scheme is implemented to find it, assuming the elements of the matrix do not vary with the propagating iterations. Then $P_{i,j}$ becomes;

$$P_{i,j}^{(k)} \cong P_{i,j}^{(\beta)} = \frac{\left(\Delta C_p\right)_i^{(\beta)} - \left(\Delta C_p\right)_i^{(0)}}{\Delta\mu_j^{(\beta)} - \Delta\mu_j^{(0)}} \tag{6.9}$$

Where β is a small number and $\left(\Delta C_p\right)^{(\beta)}$is the pressure difference at the trailing edge which is obtained for the wake strength $\Delta\mu_j^{(\beta)} = (1+\beta)\Delta\mu_j^{(0)}$. Then the above equation may be rewritten as;

$$P_{i,j}^{(\beta)} = \frac{\left(\Delta C_p\right)_i^{(\beta)} - \left(\Delta C_p\right)_i^{(0)}}{\beta\Delta\mu_j^{(0)}} \tag{6.10}$$

The Jacobian may be refined after some iterations using $\Delta\mu_j^{(\beta)} = (1+\beta)\Delta\mu_j^{(n)}$ and the following equation;

$$P_{i,j}^{(n)(\beta)} = \frac{\left(\Delta C_p\right)_i^{(n)(\beta)} - \left(\Delta C_p\right)_i^{(n)}}{\beta \Delta \mu_j^{(0)}} \tag{6.11}$$

6.3 Coupling with Boundary Layer Equations

After completing the inviscid part, the next step is integrating this solver with integral boundary layer (IBL) theory via quasi-simultaneous coupling method. The method requires a separate potential solver to supply the inviscid solution to the boundary layer equations. Hence it is assumed that FMBEM solver can be directly integrated to this system. After obtaining the boundary layer parameters, it is intended to perform noise estimations for wind turbines through empirical relations giving the sound pressure level of various noise mechanisms defined for wind turbines.

REFERENCES

[1] J. L. Hess and A. M. O. Smith, "Calculation of Non-Lifting Potential Flow About Arbitrary Three-Dimensional Bodies," Douglas Aircraft Company, ES40622, 1962.

[2] V. Rokhlin, "Rapid Solution of Integral Equations of Classical Potential Theory," *Journal of Computational Physics*, vol. 60, pp. 187-207, 1985.

[3] L. F. Greengard and V. Rokhlin, "A Fast Algorithm for Particle Simuliations," *Journal of Computational Physics*, vol. 73, pp. 325-348, 1987.

[4] Joseph Katz and Allen Plotkin, *Low Speed Aerodynamics From Wing Theory to Panel Methods*. New York: McGraw-Hill Inc., 1991.

[5] Lindsey E. Browne and Dale L. Ashby, "Study of the Integration of Wind Tunnel and Computational Methods for Aerodynamic Configurations," 1989.

[6] Jin-Tae Lee, "A Potential Based Panel Method for the Analysis of Marine Propellers in Steady Flow," Department of Ocean Engineering, Massachusetts Institute of Technology, Ph.D. Thesis 1987.

[7] J. A. C. Falcão de Campos, "A Panel Method for Calculation of the Incompressible Potential Flow on Propellers," 2000.

[8] A. E. Magnus and M. A. Epton, "PAN AIR-A Computer Program for Predicting Subsonic or Supersonic Linear Potential Flows About Arbitrary Configurations Using a Higher Order Panel Method," Vol.1. Theory Document (Version 1.0) NASA CR-3251, 1980.

[9] Hiren Dayalal Maniar, "A Three Dimensional Higher Order Panel Method Based on B-Splines," Department of Ocean Engineering, Massachusetts Institute of Technology, Ph.D. Thesis 1995.

[10] Y. J. Liu and T. J. Rudolphi, "Some Identities for Fundamental Solutions and Their Applications to Weakly-Singular Boundary Element Formulations," *Engineering Analysis with Boundary Elements*, vol. 8, no. 6, pp. 301-311, 1991.

[11] Y. J. Liu and T. J. Rudolphi, "New Identities for Fundamental Solutions and Their Applications to Non-singular Boundary Element Formulations," *Computations*

Mechanics, vol. 24, no. 4, pp. 286-292, 1999.

[12] Jen-shiang Kouh and Jyh-bin Suen, "A 3D Potential-Based and Desingularized Higher Order Panel Method," *Ocean Engineering*, vol. 28, pp. 1499-1516, 2001.

[13] Evert Klaseboer, Carlos Rosales Fernandes, and Boo Cheong Khoo, "A Note on True Desingularisation of Boundary Integrals Methods for Three-Dimensional Potential Problems," *Engineering Analysis with Boundary Elements*, vol. 33, pp. 796-801, 2009.

[14] Evert Klaseboer, Qiang Sun, and Derek Y. C. Chan, "Non-singular Boundary Element Methods for Fluid Mechanics Applications," *Journal of Fluid Mechanics*, vol. 696, pp. 468-478, 2012.

[15] Y. J. Liu, *Fast Multipole Boundary Element Method: Theory and Applications in Engineering*. New York: Cambridge University Press, 2009.

[16] Kenichi Yoshida, "Applications of Fast Multipole Method to Boundary Integral Equation Method," Department of Global Environment Engineering, Kyoto University, Ph.D. Thesis 2001.

[17] Ziliang Lin and Shijun Liu, "Calculation of Added Mass Coefficients of 3D Complicated Underwater Bodies by FMBEM," *Communications in Nonlinear Science and Numerical Simulation*, vol. 16, pp. 187-194, 2011.

[18] K. Nabors, F. T. Korsmeyer, F. T. Leighton, and J. White, "Preconditioned, Adaptive, Multipole-Accelerated Iterative Methods for Three Dimensional First-Kind Integral Equations of Potential Theory," *SIAM Journal on Scientific Computing*, vol. 15, no. 3, pp. 713-735, May 1994.

[19] F. X. Caradonna and C. Tung, "Experimental and Analytical Studies of a Model Helicopter in Hover," Technical Memorandum NASA-TM-81232, 1981.

[20] (2012, Aug.) JavaFoil - Analysis of Airfoils. [Online]. www.mh-aerotools.de/airfoils/javafoil.htm

[21] Kuo Morino L., "Subsonic Potential Aerodynamic for Complex Configurations: A General Theory," *AIAA Journal*, vol. 12, 1974.

[22] Youcef Saad and Martin H. Schultz, "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *Journal of Scientific and Statistical Computing*, vol. 7, no. 3, 1986.

[23] Fluent Inc. (2006, March) Gambit Neutral File Format. [Online]. http://www.stanford.edu/class/me469b/handouts/gambit_write.pdf

[24] Ian S. Sullivan, Joseph J. Niemela, Robert E. Hershberger, Diogo Bolster, and Russel J. Donnely, "Dynamics of Thin Vortex Rings," *Journal of Fluid Mechanics*, vol. 609, pp. 319-347, 2008.

[25] Larry A. Young, "Vortex Core Size in the Rotor Near-Wake," California, 2003.

[26] Wim R. M. Van Hoydonck and Michel J. T. Van Tooren, Validity of Viscous Core Correction Models for Self-Induced Velocity Calculations, 2012.

[27] Spyros A. Kinnas and Ching-Yeh Hsin, "Boundary Element Method for the Analysis of the Unsteady Flow Around Extreme Propeller Geometries," *AIAA Journal*, vol. 30, no. 3, 1992.

[28] Liang Shen and Yijun J. Liu, "An Adaptive Fast Multipole Boundary Element Method for Three-Dimensional Potential Problems," *Computational Mechanics*, vol. 39, no. 6, pp. 681-691, 2007.

[29] H. Cheng, L. Greengard, and V. Rokhlin, "A Fast Adaptive Multipole Algorithm in Three Dimensions," *Journal of Computational Physics*, vol. 155, pp. 468-498, 1999.

[30] A. D'Alascio, A. Visingardi, and P. Renzoni, "Explicit Kutta Condition Correction for Rotary Wing Flows," in *19th World Conference on the Boundary Element Method*, Rome, 1997.

[31] J. Baltazar and J. A. C. Falcão de Campos, "Hydrodynamic Analysis of a Horizontal Axis Marine Current Turbine with a Boundary Element Method," *Journal of Offshore Mechanics and Arctic Engineering*, vol. 133, 2011.

[32] J. Gordon Leishman, *Principles of Helicopter Aerodynamics*. New York: Cambridge University Press, 2006.

[33] J. G. Leishman, *Principles of Helicopter Aerodynamics*. New York: Cambridge University Press, 2000.

[34] Michael Metcalf and John K. Reid, *FORTRAN 90/95 Explained*.: Oxford University Press, 2004.

[35] NREL. (2012, April) NREL 10-m Wind Turbine Testing in NASA Ames 80'x120' Wind Tunnel. [Online]. wind.nrel.gov/amestest/

[36] N. N. Sorensen, J. Johansen, and S. Conway, "CFD Computations of Wind Turbine Blade Loads During Standstill Operation," Riso National Laboratory, Roskilde, KNOW-BLADE Task 3.1 Report Riso-R-1465, 2004.