Y/AMT-379

Y-12

OAK RIDGE
Y-12
PLANT

LOCKHEED MARTIN

CRADA Final Report
for
CRADA Number Y-1294-0306

SIMULATION OF 3-D ELECTROMAGNETIC FIELDS NEAR CAPACITANCE SENSORS

L. J. Gray
M. D. Morris
B. D. Semeraro
Lockheed Martin Energy Systems, Inc.
Eldon Cooper
Computer Application Systems, Inc.

September 30, 1996

Approved for Public Release;
distribution is unlimited.

Prepared by the
Oak Ridge Y-12 Plant
managed by
LOCKHEED MARTIN ENERGY SYSTEMS, INC.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC05-84OR21400

MANAGED BY
LOCKHEED MARTIN ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

UCN-13672 (26 6-95)

Abstract

Computer Application Systems, Inc. is currently developing a 'capiciflector' sensor for a variety of commercial applications, *e.g.*, object detection in robotics. The goal of this project was to create computational tools for simulating the performance of this device. The role of modeling is to provide a quantitative understanding of how the sensor works, and to assist in designing optimal sensor configurations for specific applications. A two-dimensional boundary integral code for determining the electric field was constructed, and a novel algorithm for solving the *inverse design problem* was investigated. Parallel implementation of the code, which will be required for detailed three-dimensional analysis, was also investigated.

Objectives

The goal of this project was to develop the capability of modeling the performance of a *capciflector* sensor, the ultimate goal being the optimization of sensor design. The two roles of modeling are to provide a quantitative understanding of sensor data (*e.g.*, estimating distance to the object in addition to merely detecting its presence), and to provide an effective engineering design tool. The sensor involves many adjustable parameters (capacitor plate geometry, plate separation, etc.), and identifying an optimal configuration is a highly nontrivial task. Compared to experiments, calculations are fast and simple, and thus the direct simulations can be combined with optimization algorithms to determine highly optimized sensor configurations for each specific application.

Assessment

This work has demonstrated that simulations can be effective in understanding and improving sensor performance, and thus the objectives of the CRADA were met. However, further development is required in order for CASI to have a truly functional engineering design system. The primary accomplishments of this project were:

Modeling A two-dimensional Symmetric-Galerkin boundary integral code for solving the Laplace equation was written. This program is capable of calculating

the capacitance of the sensor, and the electric field surrounding the sensor. The primary advantages of the Symmetric Galerkin method are the ability to compute an accurate solution at the sharp corners on the sensor plates, and reduced computational effort. Substantial progress was also made on the development of a three-dimensional code.

Design Optimization A function maximization algorithm and implementation were developed for use in conjunction with the Symmetric-Galerkin code, for finding the specific 'capaciflector' design which produces optimal performance for a specified setting. Given a range of 'capaciflector' design parameters the Laplace equations are solved for a sequentially determined series of problems, leading to a solution which maximizes the change in capacitance associated with the appearance of an object in the field. The optimization algorithm used is based on recent research in Bayesian function approximation methods, and has the advantage of requiring fewer function evaluations than other available methods, allowing faster solution to the inverse (device design) problem.

Parallel Implementation The close spacing of the capaciflector plates, and the singularity at the corners of the plates, imply that refined discretizations will be required for accurate modeling. Thus, a parallel implementation of the boundary element method described above was constructed, taking advantage of the ScaLAPACK parallel linear algebra software. ScaLAPACK is a package of programs to solve linear algebra problems on parallel platforms. In order to use this software, a block cyclic matrix decomposition was adopted and each processor only constructed its local portion of the matrix. The ScaLAPACK software was then employed to solve the linear system. The parallel code was exercised on a network of four IBM RS6000/590 workstations, with a discretization of the capaciflector having of 3000 nodes. Table 1 shows the execution time (in seconds) for the matrix build and solve portions of the program.

Time vs Processors		
PROC	BUILD	SOLVE
1	253	120
2	130	71
3	89	55
4	68	42

Table 1: Execution Times

DOE Benefits

This work has contributed to the advancement of ruggedized proximity sensors for robotics work and other applications, and has made the applications knowledge available for DOE projects. Experience has been gained in parallel-processing applications of the boundary element method, which may now be applied in other weapons applications.

Technical Discussion

Boundary Integral Formulation

A typical configuration for a capaciflector sensor is shown in figure 1. The ground plate is held at zero potential, while the shield and sensor plates are at possibly different potentials, but driven at the same frequency. Objects in the vicinity of the sensor will be detected from changes in the measured current: the object alters the electric field around the plates, which in turn changes the capacitance of the circuit and is reflected in the current.

The capacitance of the circuit can be obtained by integrating the flux, the normal component of the electric field, over the surface of the shield and sensor. The flux is obtained by solving the electrostatic boundary value problem. The electrostatic potential $\phi(x, y)$ satisfies the Laplace equation

$$\nabla^2\phi = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = 0 \tag{1}$$

in the infinite domain *exterior* to the capaciflector plates. The electric field E is the gradient $\nabla\phi$ of the potential, and the flux on the plates is the normal component $\nabla\phi \cdot n$ (*i.e.*, the normal derivative of ϕ).

Although the solution of Eq. (1) can be carried out by many methods, the boundary integral approach [3] is especially advantageous for this problem. Instead of solving the partial differential equation everywhere in the domain, this method solves the integral equation

$$\phi(P) + \int_\Gamma \phi(Q)\frac{\partial G}{\partial \mathbf{n}}(P,Q)\,dQ = \int_\Gamma G(P,Q)\frac{\partial \phi}{\partial \mathbf{n}}(Q)\,dQ\ , \tag{2}$$

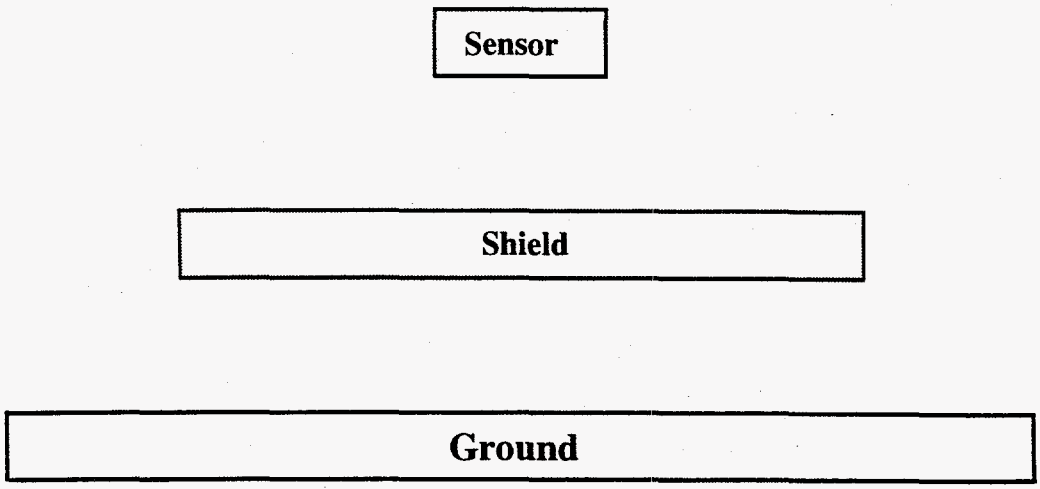

Figure 1: A capaciflector sensor consists of a base ground plate, a shield plate, and the sensor plate.

for the unknown boundary values of the flux. In this equation, Γ is the boundary of the domain (*i.e.*, the capacitor plates), and $G(P,Q) = -\log\|Q-P\|/2\pi$ is the point source potential. Note first that understanding the performance of the sensor requires evaluation of the current, which in turn is determined solely by the flux on the boundary (capacitor plates). The boundary integral method calculates these surface derivatives directly and accurately, and as interior values are not needed, this is an efficent technique. Second, the Laplace equation holds in the *infinite domain* exterior to the plates, and the boundary formulation is clearly simpler for this type of problem.

Standard collocation techniques, as employed in all commercial boundary integral software, for solving this equation would have difficulties at the corners of the sensor plates. As the supplied boundary condition is potential, there are two unknown flux values at the corner, and standard techniques cannot handle this situation. However, in the Galerkin approach, two separate equations at the corner point can be constructed, resulting in a more accurate corner treatment. As the 'edge effects' are important in the sensor operation, this is an important advantage. In addition, the Symmetric Galerkin method provides a fast algorithm [1].

Sensor Optimization

Design of the 'capaciflector' consists of specifying the dimensions and locations of plates displayed in Fig. 1. Because the device allows detection of objects through changes in the circuit capacitance, device designs resulting in the largest changes for a given object are deemed optimal for this purpose. The associated inverse problem is then optimization of the difference in computed capacitances, with and without the object to be sensed, with respect to the design parameters.

The primary practical difficulty with this problem is the relatively large number of function evaluations required to complete the optimization process, particularly when a number of design parameters are considered simultaneously. Most standard numerical techniques are iterative, and are based on a series of local approximations to the objective function and its derivatives, within small neighborhoods of the design space. Computational overhead for such search procedures is generally small, but if objective function evaluations are time consuming, the overall task of optimization using most popular techniques can be prohibitive.

In this work, we have developed a function optimization procedure which, while requiring somewhat more overhead than conventional techniques, accomplishes maximization of smooth objective functions using fewer evaluations. The method is based on recent research in Bayesian function approximation [2] including a sequential strategy based on the following steps:

1. A standard starting set of Symmetric Galerkin evaluations is made, which are well-spread throughout the available design region.
2. Using the data from the set of evaluations which have been made at a given point, the objective function is approximated via a nonparametric statistical method.
3. Response (change in capacitance) is predicted for each possible device configuration, and predictive standard errors of these predictions are determined within a Bayesian formulation.
4. The design with the largest "credible" response is identified, based on these predictions and standard errors, the Symmetric Galerkin code is executed for this configuration, the results of the run is added to the collection, and a new cycle of modeling and prediction is begun.

The final step listed here is, itself, an optimization problem. However, it is computationally much less demanding than direct optimization using the Symmetric Galerkin code since the function approximation can be quickly evaluated and its derivatives are known. The process is terminated when the "credible" response (prediction plus a fixed multiple of the predictive standard error) for all possible devices is no greater than the largest calculated response so far observed.

Preliminary results indicate that this procedure effectively identifies optimal parameters for 'capaciflector' design using far fewer evaluations than would be required by most currently available derivative-free optimization techniques.

Inventions

No inventions arose out of this work. Several journal articles, describing the novel computational techniques developed for this project, are currently being prepared. A preprint of a paper concerning the parallel implementation, accepted for publication in *Engineering Analysis with Boundary Elements*, is attached to this report.

Commercialization

Although there appears to be no commercial potential for the code itself, the code will potentially expand the commercial applications of the capciflector sensor. The ability to model and optimize sensor design can (a) reduce design and experimentation time (time to market), (b) reduce manufacturing costs (*e.g.* if the sensor can be made smaller it lowers material costs), and (c) allow sensor design for novel applications.

Future Collaboration

CASI and the project team are exploring potential funding sources (DoD SBIR, commercial contracts, etc.) for continuing the development of the design software. In addition, CASI is investigating the application of sensors which are driven at high frequency. Modeling of these devices would require the solution of Maxwell's equations, rather than the Laplace equation, and efficient analysis of time dependent elec-

tromagnetic fields can also be carried out with boundary integral techniques. Thus, the work in this project can be extended to high frequency sensors.

Conclusions

This project has been successful in demonstrating the ability to model and analyze the capaciflector sensor, and in providing CASI with a preliminary design tool. Further development is required to establish a fully functional system which can be routinely employed in CASI's engineering design process.

References

[1] C. BALAKRISHNA, L. J. GRAY, AND J. H. KANE, *Efficient analytical integration of symmetric Galerkin boundary integrals over curved elements; thermal conduction formulation*, Comput. Methods Appl. Mech. Engrg., 111 (1994), pp. 335–355.

[2] C. CURRIN, T. MITCHELL, M. MORRIS, AND D. YLVISAKER, *Bayesian prediction methods of deterministic functions, with applications to the design of computer experiments*, J. Amer. Statistical Assoc., 86 (1991), pp. 953–963.

[3] J. H. KANE, *Boundary Element Analysis in Engineering Continuum Mechanics*, Prentice Hall, New Jersey, 1994.

PVM Implementation of the Symmetric-Galerkin Method *

B. D. Semeraro and L. J. Gray
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6367

Abstract

We report on initial progress towards a Parallel Virtual Machine (PVM) implementation of the Symmetric-Galerkin boundary integral method. We take advantage of software packages specifically designed to solve linear algebra problems on distributed memory parallel computers. In particular we use linear algebra routines from the ScaLAPACK, PBLAS, and BLACS, libraries. These routines assume a block cyclic decomposition of the matrix operands. The decomposition of the operands and its impact on the construction of the coefficient matrix are described. Computational results for solving the two-dimensional Laplace equation are presented. This program is being used to simulate the performance of a proximity sensor used in robotics and other applications.

Key words: Boundary Element method, parallel processing, workstation cluster, proximity sensor, block linear algebra algorithms.

*Invited paper, special issue of *Engineering Analysis with Boundary Elements.* This research was supported by the CRADA agreement Y1294-0306, and by the Applied Mathematical Sciences Research Program of the Office of Mathematical, Information, and Computational Sciences, U.S. Department of Energy under contract DE-AC05-96OR22464 with Lockheed Martin Energy Research Corp.

1 Introduction

The Symmetric-Galerkin (SG) method [19, 23, 24] has emerged as a robust and highly efficient boundary integral algorithm. The importance of this relatively new approximation primarily derives from two features. First, in this approach, hypersingular integrals can be evaluated *without* a C^1 boundary interpolation [16, 25], a consequence of the additional boundary integration in the Galerkin formulation. Thus, standard and relatively simple to implement C^0 conforming element technology can be employed. As is well known, however, the extra Galerkin integration is computationally quite expensive. The second aspect of this method, obtaining a symmetric coefficient matrix, is therefore equally important, in that it reduces computation times (on serial computers) to a level rivaling standard collocation [2, 3].

Hypersingular equations are essential for the boundary integral analysis of crack geometries [4, 7, 18, 21], and thus one of the main applications of Symmetric-Galerkin is the important area of fracture analysis [17]. However, even with the computational advantages of SG , realistic three-dimensional fracture analysis, multiple cracks in a composite material for example, will be beyond the computing power of a single workstation. Moreover, as will be discussed below, there are applications, even in two dimensions, which require large scale computing. Thus, the development of a parallel implementation of SG is highly desirable.

For engineering firms, parallel supercomputers are generally unavailable. The most readily accessible form of parallel computing for most companies is networked workstations, and consequently we investigate the performance of the SG algorithm using the Parallel Virtual Machine (PVM) software. PVM is a portable message passing library. PVM can be used to support message passing on many parallel platforms from massively parallel supercomputers to network connected collections of workstations. In this work PVM is the message passing layer on which the parallel linear algebra packages, ScaLAPACK, PBLAS, and BLACS, are built. The interface to the message passing is hidden from the user in these packages in that no explicit buffer packing and sending is done. The required data transfer is handled by the linear algebra packages themselves when an operation is performed on a global data item.

Previous investigations of parallel boundary integral methods (see [8, 9] and references therein) have primarily dealt with collocation methods. Nev-

ertheless, the structure of a Galerkin code is not very different from a collocation algorithm, and thus is is possible to take advantage of previous work. Following [13], we have chosen to employ a block decomposition of the coefficient matrix, *i.e.*, each processor is responsible for constructing specific sub-blocks of the matrix. We have exploited the availability of a parallel linear algebra routine based upon a block decomposition.

Of primary interest here, in our opinion, is the need for parallel computing to carry out actual engineering calculations. The modeling application involves solution of the two dimensional Laplace equation, and is aimed at improving the performance of a capacitance type proximity sensor. The simulations can provide a quantitative understanding of the sensor measurement (changes in current), and can be used to optimize the sensor design for specific applications. As will be discussed further below, these simulations require extensive computing resources, even in two dimensions.

2 Symmetric-Galerkin

For simplicity, and as it relevant to the calculations presented below, the *SG* method will be presented in the context of the two dimensional Laplace equation. If $\nabla^2\phi = 0$ holds in the domain Ω with boundary curve γ, the corresponding boundary integral formulation is given by [5, 20]

$$\phi(P) + \int_\gamma \phi(Q) \frac{\partial G}{\partial \mathbf{n}}(P,Q)\, dQ = \int_\gamma G(P,Q) \frac{\partial \phi}{\partial \mathbf{n}}\, dQ \ , \tag{1}$$

where $\mathbf{n} = \mathbf{n}(Q)$ is the unit outward normal at the point $Q \in \gamma$ and $\partial/\partial\mathbf{n}$ denotes the normal derivative. Although the fundamental solution $G(P,Q)$ is usually taken to be the point source potential

$$G(P,Q) = -\frac{1}{2\pi} \log \|Q - P\| \ , \tag{2}$$

specialized Green's functions, which partially satisfy the prescribed boundary conditions, are sometimes advantageous. As discussed in Section 3, the sensor simulations will in fact use a modified version of Eq. (2).

As written, Eq. (1) holds for a point $P \in \Omega$ interior to the domain, and defining the singular integrals in terms of a limit to the boundary [15], *also for* $P \in \gamma$ [22]. Differentiating this equation with respect to P in the

direction $\mathbf{N} = \mathbf{n}(P)$ results in the corresponding (hypersingular) equation for surface flux,

$$\frac{\partial \phi}{\partial \mathbf{N}}(P) + \int_\gamma \phi(Q) \frac{\partial^2 G}{\partial \mathbf{N} \partial \mathbf{n}}(P,Q)\, dQ = \int_\gamma \frac{\partial G}{\partial \mathbf{N}}(P,Q) \frac{\partial \phi}{\partial \mathbf{n}}(Q)\, dQ \ . \qquad (3)$$

It can be shown that, assuming $\phi(Q)$ is differentiable at P, the limit as the interior point P approaches the boundary exists, and thus Eq. (3) remains valid for $P \in \gamma$ [15].

In a standard collocation approximation, the unknown boundary values of potential or flux are determined by insisting that either Eq. (1) or Eq. (3) holds at the boundary nodes. This inevitably leads to a non-symmetric system of linear equations, as the source point P only enters through its coordinates, while the the complete neighborhood geometry of Q is taken into account. However, P and Q do enter in a symmetric fashion in a Galerkin formulation. A Galerkin approximation is a weighted residual formulation in which the shape functions $M_k(Q)$ that are used to approximate the boundary functions,

$$\begin{aligned} \phi(Q) &= \sum_k \phi(P_k) M_k(Q) \\ \frac{\partial \phi}{\partial \mathbf{n}}(Q) &= \sum_k \frac{\partial \phi}{\partial \mathbf{n}}(P_k) M_k(Q) \end{aligned} \qquad (4)$$

also serve as the weight functions in the residual statement. Thus, Eq. (1) is approximated as

$$\begin{aligned} \int_\gamma M_k(P) \left[\phi(P) + \int_\gamma \phi(Q) \frac{\partial G}{\partial \mathbf{n}}(P,Q)\, dQ \right] dP = \\ \int_\gamma M_k(P) \left[\int_\gamma G(P,Q) \frac{\partial \phi}{\partial \mathbf{n}}\, dQ \right] dP \ , \end{aligned} \qquad (5)$$

In matrix form, this equation can be written

$$\mathcal{H}_a \Phi = \mathcal{G}_a \Phi^{\mathbf{n}} \qquad (6)$$

where Φ, $\Phi^{\mathbf{n}}$ denote the column vectors of boundary values of potential $\{\phi(P_k)\}$ and flux $\{\partial\phi/\partial\mathbf{n}(P_k)\}$. Similarly, the hypersingular equation Eq. (3) reduces to

$$\mathcal{H}_b \Phi = \mathcal{G}_b \Phi^{\mathbf{n}} \ . \qquad (7)$$

The double integration over the boundary, together with the symmetry properties of the fundamental solution (note that $G(P,Q) = G(Q,P)$) ensure that the matrices $\mathcal{G}_a$ and $\mathcal{H}_b$ are symmetric. Thus, for a Dirichlet problem, Eq. (5) produces a symmetric coefficient matrix, and similarly the hypersingular equation yields a symmetric matrix for a Neumann problem. In 1985, Hartman [19] showed that symmetry is achieved for a mixed boundary value problem if Eq. (5) is written on the Dirichlet surface and the hypersingular equation on the Neumann surface (see also [23, 24]). The main advantage provided by the symmetry is that a direct solution is twice as fast as for a nonsymmetric matrix.

3 Sensor Modeling

The purpose of this section is to describe an industrial problem, the modeling of a capacitance type sensor, which requires parallel computing resources, even for two dimensional simulations. These proximity sensors have many applications, and the computations are essential for understanding the performance of the sensor and for optimizing sensor design for specific uses. This work has been carried out in collaboration with *Computer Application Systems, Inc.*, a company that designs and builds these devices.

A typical configuration for a capaciflector sensor [26], consisting of ground, shield, and sensor plates, is shown schematically in figure 1. The ground plate is held at zero potential, while the shield and sensor plates are at constant, possibly different, potentials. For low frequency applied voltages, the problem can be modeled as electrostatic, with the potential $\phi(x,y)$ satisfying the Laplace equation in the infinite domain *exterior* to the plates. Objects in the vicinity of the sensor will be detected from changes in the measured current: the object alters the electric field around the plates, which in turn changes the capacitance, and hence the current, of the circuit.

While the geometry appears to be quite innocuous, these calculations are actually very difficult: the horizontal and vertical axes in Fig. 1 are *not* on the same scale. The width (vertical dimension) of the plates is typically *three orders of magnitude smaller* than the length. Moreover, the vertical gap between plates is the same order of magnitude as plate width, and thus the plates cannot be idealized as infinitely thin. Consequently, both sides of all plates must be included, and furthermore the element size must be on the

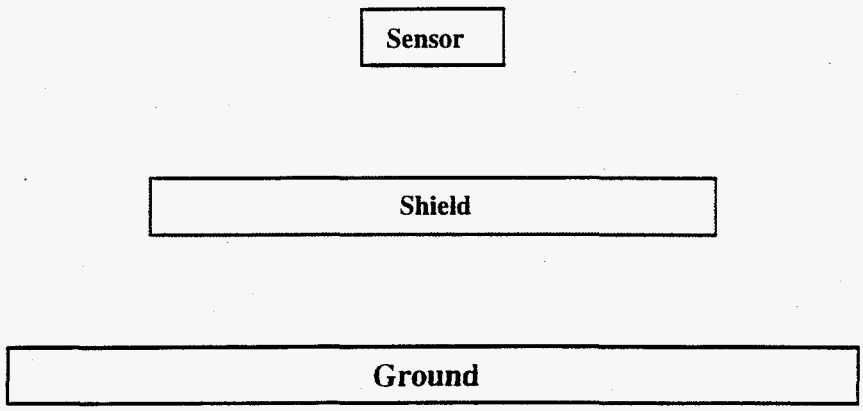

Figure 1: A capaciflector sensor consists of a base ground plate, a shield plate, and the sensor plate.

order of the plate width. This last condition is required for accurate evaluation of the near-singular integrals which arise. The computation is further complicated by the re-entrant (for the exterior domain) rectangle corners. The constant potential boundary condition guarantees that the surface flux approaching a corner is singular, of the form $r^{-1/3}$, where r is the distance to the corner [10]. This singularity clearly presents a challenge for any numerical method, and will also necessarily require a refined discretization near the corners.

Modeling of this sensor is therefore inherently difficult, due to the disparity of the length scales that are present. While specialized techniques could be invoked to partially reduce the computational work, any approach will involve large scale computing. In addition, as indicated above, the goal of this work is to employ simulations to optimize the design of the sensor configuration. This will require an iterative process involving many computations, and thus parallel computing is essential.

Note that the Laplace equation is posed in the infinite exterior domain, and this too presents somewhat of a problem. Although the physical boundary condition at infinity is that the potential decay to zero, the Green's function, Eq. (2), diverges as $\|Q - P\| \to \infty$. The boundary element procedure, using just the finite boundary, will pick out the solution for which the integrals over a far-field boundary in either Eq. (1) or Eq. (3) cancel

out of the equation [5]. This, however, does not necessarily correspond to the physical solution. In particular, sensor simulations carried out without any regard for the far-field boundary condition produced unphysical answers. This problem has been dealt with by placing the entire calculation in a strip, $-R \leq y \leq R$, R large, and requiring that the flux vanish on the strip boundaries $y = \pm R$. Rather than contend with these boundaries directly, which would significantly increase the size of the matrix and the solution time, an approximate Green's function technique was employed. This method, which will be described elsewhere publication [14], is based upon the well-known reflection technique [5] for incorporating symmetry. It provides a physical solution without having to directly introduce a far-field boundary, which would have the undesirable consequence of increasing the problem size.

4 PVM Implementation

In this section we discuss the parallel implementation of the SG method outlined above. This initial work has centered on the simplest, but nevertheless useful, situation, the solution of the two dimensional Laplace equation with Dirichlet data. The main consequence of this is that only the boundary integral equation, Eq. (1), is employed. Nevertheless, the algorithm for constructing the hypersingular flux equation is structured the same as the potential equation, and thus the performance results for this Dirichlet code are expected to carry over to a general SG program.

The Laplace algorithm employed in this work differs from a 'conventional' boundary integral approach in two respects. First, in many implementations, both $\mathcal{H}_a$ and $\mathcal{G}_a$ matrices are built and stored. While this is a convenient approach for relatively small problems, the storage of both matrices can limit the size of problem that can be attacked. Thus, the parallel code only stores $\mathcal{G}_a$, constructing the right hand side vector, which consists of $\mathcal{H}_a$ times the vector of known potentials, 'on the fly'. This effectively cuts the memory requirement in half, and allows consideration of larger problems. Moreover, this approach is not an impediment for the design iteration, as changes in the geometry would necessarily force recomputation of both matrices. The other modification, incorporated specifically for the sensor problem, addressed the requirement that the potential vanish at infinity. As discussed above, this boundary condition was included in the Green's function. The tradeoff here

is between the additional computation required for this fundamental solution, versus the increased size of $\mathcal{G}_a$ (and the additional computation) that results from adding more boundary surface. Again viewing the problem size as the main limiting factor, the Green's function approach was deemed preferable.

Networked parallel processing, such as PVM, offers a reasonable solution to the problem of solving large scale boundary element problems. This is especially true for small or medium sized engineering companies, as access to mainframe machines will be nonexistent or too expensive. A collection of network connected workstations was employed as the parallel platform for this work. The workstation cluster consisted of four IBM RS6000 590 workstations connected via Ethernet, with each workstation having at least 128 megabytes of memory. The message passing capabilities were provided by the PVM system.

Linear Algebra

A boundary integral algorithm consists of two main parts, the construction and solution of a large dense system of linear equations. The matrix construction consists of completely independent integrations and is inherently perfectly parallel. The construction of the right hand side involves a matrix vector multiplication and will involve some communication. The factorization of the coefficient matrix and solution of the linear system also require interprocess communication. Thus, a main determining factor in the deciding on the division of work among processors is the availability of effective parallel matrix operation methods. The data decomposition strategy used in our work was heavily influenced by the desire to use the ScaLAPACK [11] library. ScaLAPACK is a library of routines for solving linear algebra problems on distributed memory computers. It was designed to be a parallel version of the LAPACK [1] library, and like LAPACK it uses block factorization algorithms and parallel versions of the level 2 and level 3 Basic Linear Algebra Subprograms (BLAS). The decomposition used in our work is actually imposed by the parallel BLAS, PBLAS [6]. The PBLAS assumes a block-cyclic decomposition of the matrix operands. That is the global matrix is partitioned into blocks and the blocks are distributed cyclically among the processors.

Both ScaLAPACK and the PBLAS routine are designed to be portable and to have calling syntax similar to their scalar counterparts. The PBLAS

in particular are meant to be building blocks for implementing parallel linear algebra algorithms. In order to achieve this, in a distributed memory environment, a set of Basic Linear Algebra Communications Subprograms (BLACS) [12] was designed. The BLACS is a communications library that allows the programmer to perform message passing on matrix subsections rather than lower level data. Portability and performance are achieved by using local message passing libraries and efficient BLAS implementations in support of the PBLAS and BLACS.

The current ScaLAPACK library unfortunately does not contain a solution algorithm for symmetric indefinite systems. Symmetry is of course a basic motivation for using the SG approach, and this is therefore a major drawback. Nevertheless, as will be discussed below, the ScaLAPACK nonsymmetric solver produced excellent efficiencies.

Block decomposition

The routines that perform the task of constructing $\mathcal{G}_a$ and the right hand side vector contain two nested loops. The inner (Q integration) and outer (P integration) loop indices correspond to row and column indices respectively in the matrices. The block cyclic decomposition used by the linear algebra software has a direct influence on how the loop structure of the matrix element assembly process is parallelized. The reader is referred to the material on block decompositions in [6] for a detailed description of the decomposition.

The loop structure in the serial code assembles the matrix entries by rows. The parallel code assembles the matrix entries by blocks. The block size is that used in the decomposition of the matrix. Only local blocks are constructed. This adds an additional outer loop over blocks to the matrix construction routines. The outer loop calculates the global starting indices of the first element in the block to be constructed and the indices in the local array where this block is to be stored. These indices are used by the inner loops to calculate the entries for this block. This scheme adds a small additional overhead to the matrix construction process.

5 Performance

The numerical experiments consisted of solving a Dirichlet problem on a circle, and of solving the sensor problem described above. The problem size for the circle used 3000 nodes. This is easily solvable on a single workstation, and yet large enough to be a sensible test of the distributed computation. The sensor problem used 6600 nodes to describe the geometry. This problem was too large to solve on all but one of the workstations, as the memory requirement of the serial version of this code was close to 700 megabytes. The largest workstation had 512 megabytes of main memory and another 512 megabytes of swap space on disk. The solution of this problem benefited greatly from the use of parallel processing.

As a measure of the effectiveness of the implementation, the performance of the parallel code was compared to the serial code. As mentioned above, ScaLAPACK presently has no routine for solving symmetric indefinite systems and so a general LU factorization algorithm, *pdgesv*, was used. A general LU factorization method was also used in the serial case even though routines for solving symmetric indefinite systems on a single processor are readily available. The LAPACK routine *dgesv* was used to solve the linear system in the serial case.

The results of the test runs are shown in table 1. The single processor code assembles the matrix and solves the linear system of equations in 253 and 120 seconds respectively. The solution time of 120 seconds for this problem translates to a computational rate of 150 Mflops/sec. The LAPACK users guide [1] reports a rate of 168 Mflops/sec for this routine, (*dgesv*), on the same processor (IBM RS6000/590). The difference in execution rates is most likely due to the fact that we did not have access to the IBM ESSL library for the level 3 BLAS.

The multiprocessor times in table 1 show that the matrix build portion of the program scales reasonably well. This portion of the code achieved speed up factors of 1.9, 2.8, and 3.7 on 2, 3, and 4 processor runs respectively. The ScaLAPACK routine did not scale as well in this test achieving a speed up of approximately 3 on four processors. Considerable communication occurs in this section of the code. The impact of the communication cost appears in the less than linear scaling.

For the 6600 node sensor geometry, the one workstation having a large memory capacity and was used to run the serial version of the code. The

Time vs Processors		
PROC	BUILD	SOLVE
1	253	120
2	130	75
3	89	55
4	68	42

Table 1: Execution Times

execution time in this case was 3 hours 36 minutes. The parallel version, running on the four workstation cluster, executed in 43 minutes, which represents a 5 fold increase in speed. The reason for this superlinear speed up is that the serial code size (700 megabytes) far exceeded the physical memory of the workstation (512 megabytes), and thus computer must have paged in and out of physical memory.

6 Conclusions

Although the work presented here is preliminary, it nevertheless demonstrates that boundary integral computations, and in particular the *SG* approximation, can be effectively carried out on networked workstations. We intend to continue this work on several fronts. First a parallel iterative solver will be developed. The plan is to substitute parallel BLAS calls for the serial BLAS in a GMRES and QMR iterative package. The logic for building the coefficient matrix will also be revisited. It is expected that more efficient serial and parallel versions of this part of the code can be developed. Second, a general *SG* algorithm, one that includes the use of the hypersingular equation, needs to be investigated. The concern here is that the hypersingular integrations require more work than their potential equation counterparts, and thus load balancing of the calculation might become a problem. Finally, realistic three dimensional sensor simulations will clearly require more computing resources, and a larger cluster of workstations needs to be employed.

References

[1] E. ANDERSON ET AL., *LAPACK Users' Guide*, Society for Industrial and Applied Mathematics, New York, 1995.

[2] C. BALAKRISHNA, L. J. GRAY, AND J. H. KANE, *Efficient analytical integration of symmetric Galerkin boundary integrals over curved elements; thermal conduction formulation*, Comput. Methods Appl. Mech. Engrg., 111 (1994), pp. 335–355.

[3] ——, *Efficient analytical integration of symmetric Galerkin boundary integrals over curved elements; elasticity formulation*. Comput. Methods Appl. Mech. Engrg., 117 (1994), pp. 157–179.

[4] M. BONNET, *Regularized direct and indirect symmertic variational BIE formulations for three-dimensional elasticity*. submitted.

[5] C. A. BREBBIA, J. C. F. TELLES, AND L. C. WROBEL, *Boundary element techniques*, Springer-Verlag, Berlin and New York, 1984.

[6] J. CHOI, J. DONGARRA, S. OSTROCHOV, ET AL., *A proposal for a set of parallel basic linear algebra subprograms*, Tech. Report CS-95-292, University of Tennessee, 1995.

[7] T. A. CRUSE, *Boundary Element Analysis in Computational Fracture Mechanics*, Kluwer Academic Publishers, Boston, 1988.

[8] A. J. DAVIES, *Parallel algorithms for the boundary element method*, in Boundary Element Technology VIII, H. Pina and C. A. Brebbia, eds., 1993, pp. 303–312.

[9] ——, *Parallel boundary element implementations: a survey*, Tech. Report 269, University of Hertfordshire, 1993.

[10] C. DETOURNAY, *A cauchy integral element for power-type singularities*, Appl. Math. Modelling, 16 (1992), pp. 450–463.

[11] J. DONGARRA ET AL., *Scalapack users' guide*, Tech. Report Draft, University of Tennessee, 1995.

[12] J. DONGARRA AND R. C. WHALEY, *A user's guide to the blacs v1.0*, Tech. Report CS-95-281, University of Tennessee, 1995.

[13] R. E. FLANERY, J. B. DRAKE, AND L. J. GRAY, *Boundary elements on distributed memory architectures*, Int. J. Numer. Meth. Engrg. submitted.

[14] L. J. GRAY, *An approximate Green's function method for boundary integral analysis.* in preparation.

[15] ——, *Boundary element method for regions with thin internal cavities*, Engrg. Analy. Boundary Elem., 6 (1989), pp. 180–184.

[16] ——, *Evaluation of hypersingular integrals in the boundary element method*, Mathematical and Computer Modelling, 15 (1991), pp. 165–174.

[17] L. J. GRAY, C. BALAKRISHNA, AND J. H. KANE, *Symmetric Galerkin boundary integral fracture analysis*, Engrg. Analy. Boundary Elements, 15 (1995), pp. 103–109.

[18] L. J. GRAY, L. F. MARTHA, AND A. R. INGRAFFEA, *Hypersingular integrals in boundary element fracture analysis*, Int. J. Numer. Meth. Engrg., 29 (1990), pp. 1135–1158.

[19] F. HARTMAN, C. KATZ, AND B. PROTOPSALTIS, *Boundary elements and symmetry*, Ingenieur-Archiv, 55 (1985), pp. 440–449.

[20] J. H. KANE, *Boundary Element Analysis in Engineering Continuum Mechanics*, Prentice Hall, New Jersey, 1994.

[21] G. KRISHNASAMY, F. J. RIZZO, AND T. J. RUDOLPHI, *Hypersingular boundary integral equations: Their occurrence, interpretation, regularization and computation*, in Developments in Boundary Element Methods - Advanced Dynamic Analysis, P. K. Banerjee and S. Kobayashi, eds., vol. 7, Elsevier Applied Science Publishers, 1991, ch. 7.

[22] E. D. LUTZ AND L. J. GRAY, *Exact evaluation of singular boundary integrals without CPV*, Comm. Num. Meth. Engrg., 9 (1993), pp. 909–915.

[23] G. MAIER, G. NOVATI, AND S. SIRTORI, *On symmetrization in boundary element elastic and elastoplastic analysis*, in Discretization methods in structural mechanics, G. Kuhn and H. Mang, eds., Springer-Verlag, Berlin and New York, 1990, pp. 191–200.

[24] G. MAIER AND C. POLIZZOTTO, *A Galerkin approach to boundary element elastoplastic analysis*, Computer Methods in Applied Mechanics and Engineering, 60 (1987), pp. 175–194.

[25] P. A. MARTIN AND F. J. RIZZO, *Hypersingular integrals: how smooth must the density be?*, Int. J. Numer. Meth. Engrg., 39 (1996), pp. 687–704.

[26] J. M. VRANISH, R. I. MCCONNEL, AND S. MAHALINGAM, *Capaciflector collision avoidance sensors for robots*, Int. J. of Computers and Electrical Engineering, 17 (1991), pp. 173–179.

Distribution

L. J. Gray, MS-6367, 6012
M. D. Morris, MS-6367, 6012
B. D. Semeraro, MS-6367, 6012
J. E. Ferguson, MS-8242, 701 SCA
G. W. Joe, MS-8084, 9203
C. M. Davenport, MS-8084, 9203
W. P. Painter, MS-6416, 5002
P. L. Gorman, MS-6269, 4500N
DOE Office of Patent Counsel, FOB
Laboratory Records, MS-6285, 4500-N
Y-12 Central Files, MS-8169, 9711-5 (3 copies)

Eldon Cooper, Computer Application Systems, Inc., 100 Cherokee Blvd., Chattanooga, Tennessee 34705 (5 copies)