ELEKTRİK, VOL.6, NO.1, 1998, ©TÜBİTAK EMO

Object-Oriented Computer Simulations of Physical Systems Using Dual Reciprocity Boundary Element Methodology

J. Friedrich
Computer Engineering Department
Karadeniz Technical University
61080 Trabzon - TURKEY

Abstract

Models of physical systems are essential in every engineering field. This work deals with computer simulations of physical systems that can be mathematically modelled by differential equations together with sufficient boundary conditions. The computer simulations are based on object-oriented technology and the dual reciprocity boundary element method which is a universal solution scheme for various types of partial differential equations (e.g. Laplace, Poisson, diffusion, convection-diffusion, and steady Navier-Stokes equation). This technique fulfills efficiency criteria like precision, robustness, versatility, programmability, user-friendliness, need of computational time and computer memory to a very high degree. This is demonstrated by three examples: Laplace's solution for a potential flow problem, Poisson's solution for a torsion problem, and the diffusion solution for cooling a metal piece.

Keywords: *computer simulations, physical systems, object-oriented technology, dual reciprocity boundary element method.*

1. Introduction

The methodology used by scientists and engineers in trying to understand physical phenomena has been theoretical development (analysis) on one hand, and observations in nature or in the laboratory on the other. Theories without checking them against observations cannot be verified. With the advent of digital computers, the methods mentioned above have been enriched by rapid and accurate computations. Thus, three scientific approaches are being used: analysis, laboratory and computational experimentation or simulation. The role of the latter one has become more and more important because they not only provide answers to problems for which analytical solutions are not available, but they are in many cases easier to apply, faster, more precise and less expensive than experiments in laboratories or in nature.

Analysis of physical systems is based on mathematical models whose development usually begins by setting up the differential equation(s) governing the physical phenomena being investigated. This step usually involves a number of relationships between the physical variables involved. Generally, the governing equations consist of continuity, momentum and energy equations which are all determined by considering a differential element representative for the phenomena to be analyzed and deriving from there mathematical relations between the differential quantities involved. Following this initial step, the model equation(s)

is(are) applied over the domain where the solution is being sought. For finding a unique solution, sufficient conditions on the boundary of the domain have to be known. Thus, physical systems in this work are defined as systems with a given domain and boundary whose physical behavior can be mathematically modelled by differential equations in this domain.

Boundary conditions may be constant or vary in time. The most difficult part is to obtain a unique solution of the governing equation(s) subject to the given boundary conditions. If such a solution is obtainable by just analytical means, the model is called an 'analytical model' and the solution is referred to as the analytical or exact solution. The nonlinear nature of the considered governing equation(s) or a complex boundary geometry of the domain often precludes analytical solutions and numerical solution methods become the only feasible tools for obtaining results of some acceptable degree of accuracy and detail. At the heart of the numerical experimentation and computer simulations of physical systems lies the numerical model. It should fulfill efficiency criteria like precision, robustness, versatility, programmability, user-friendliness, need of computational time and computer memory as much as possible.

Once the computer code for a numerical model has been developed, debugged and calibrated (or benchmarked), it may be employed repeatedly to obtain solutions for different data input sets. In contrast to laboratory experimentation involving physical models, the alteration of boundary conditions and calibration of various parameters can be carried out in a much shorter time with much less cost and effort. Further, computational experimentation does not suffer from unclean data of real observations in nature or in a laboratory, allowing to remove or separate certain effects e.g. friction. Computer simulations also permit to change the governing equation(s) and parameter values so that alternative models and their solutions can be tested. Moreover, they allow to estimate the behavior of a physical system in the future and in the past by forward and backward time integration schemes applied to the governing equation(s).

The numerical model used in this work is based on the dual reciprocity boundary element method, a newly derived scheme for solving numerically a variety of partial differential equations without changing the overall solution algorithm. It was implemented by object-oriented techniques which are introduced in the next section before describing the numerical model.

2. Object-Oriented Technology

Object-oriented technology (OOT) is a method that seeks to mimic the way humans form models of the world. To cope with coplexities of life, human beings have evolved a wonderful capacity to generalize, classify and generate abstractions. Almost every noun in human vocabulary represents a class of objects sharing some set of attributes or behavioral traits. OOT exploits this natural tendency humans have to classify and abstract things. For example, the object-oriented programming language $C/C++$ was originally called 'C with Classes'. OOT is implemented by object-oriented programming, which can be characterized by supporting and enforcing the use of the OO key concepts encapsulation, inheritance, polymorphism, and dynamic (or late binding) summarized in the following table together with other key OO terms (McMonnies and McSporran 1995).

Encapsulation greatly facilitates the usage of complex data structures to model a group- or semigroup structure, thus allowing to collect information of different types that belong together naturally into one class as a single unit, e.g. data of a boundary element node containing its number, coordinates, type and value of boundary condition. Therefore, encapsulation increases the readability and maintainability of the model and code. Further, encapsulation sets a boundary between a class and the rest of the program, protecting its contents from unwanted side effects and thereby making classes more robust.

Table 1. List of important object-oriented terms

Object	A data structure that incorporates type-specific methods for self-management
Class	A template for objects. A class definition is the means by which objects are designed.
Member	A variable or method (function) within an object
Instance	An object of a particular class
Encapsulation	The binding of data structures and methods into a class of objects
Inheritance	Means that derived classes inherit the variables and methods from their base classes, while possibly redefining or adding new variables and methods. This creates a hierarchy of ancestor classes.
Polymorphism	The ability of classes in a hierarchy to share names for methods that behave appropriately to the particular class for which they were designed
Late binding	The binding of virtual methods to an object when it is created during run-time
Constructor/Destructor	Methods to create / eliminate objects

It is also very beneficial to use existing classes when defining new classes. This can be achieved by composition, or by extension of already defined classes using the concept of inheritance. The main purpose of inheritance is to allow sharing of attributes and methods among classes according to a hierarchical relationship. From one class, any desired number of class objects can be initialized and used, a process similar to the casting of pieces from a single mould. Each of these copies is autonomous, doing its job on its own, but if it needs some code or data from another class object, the concept of inheritance allows them to share it. Inheritance is of crucial importance for the realization of tree-structured dependencies between class objects. The ability to transfer common attributes and methods of several classes to a base class and to inherit them from there can greatly reduce repetition within a model and/or program and is one major benefit of the OO approach (Pohl 1993).

Sometimes, a set of class objects may share a routine, but it can only be defined at a specific program level or stage. Thus, this routine can be initialized as a virtual method (function), but later defined at run-time when it is required. This is part of the polymorphism concept which makes it possible to adopt an initialized function to the needs of an object class in the same inheritance tree by adding new features or overwriting existing ones. This process is also called dynamic (or late) binding, which makes the object-oriented method so flexible and efficient.

All four concepts (encapsulation, inheritance, polymorphism, dynamic binding) are the foundation of the OOT, which is a new way of looking at, analyzing, designing, and implementing software, concentrating on the underlying real-world concepts and principles of a system, rather than processing or implementation details. Thus, there is a general trend in software development away from programming language issues towards the fundamentals and rules within the application domain, which includes the identification and organization of system components and concepts, their behavior and relations. This is of great importance for reducing the software mountain because general concepts enjoy a longer life span than some implementation details.

In conclusion, OOT is conceptual process independent from a programming language until the last stages (Rumbaugh et al. 1991). It is a fundamental new way of thinking about software and not a programming technique. Its greatest advantages originate from supporting client/users and designers to get a better and more complete picture of a problem domain and communicate it more clearly to each other. It serves as a more efficient platform for all parts of the software life cycle including concepts, specifications, analysis, design, implementation, testing, documentation, interfacing, and maintenance.

3. The Dual Reciprocity Boundary Element Method

Since its beginnings in the 1960's, the boundary element method (BEM) has become a well-established numerical technique which provides an efficient alternative to finite difference and finite element methods for solving a variety of engineering problems (Brebbia 1978, Banerjee 1994). One main restriction of the classical BEM has been the requirement for a fundamental solution to the original partial differential equation in order to obtain an equivalent boundary integral equation. Another has been that non-homogeneous parts in the governing equations are incorporated by means of domain integrals so that the BEM looses its original attraction of a 'boundary-only' method. The dual reciprocity method (DRM) appears to be a solution to the mentioned difficulties of the classical BEM (Partridge et al. 1992, Power and Partridge 1994, Power and Wrobel 1995). The DRM uses a fundamental solution to a simpler governing equation and takes into account the remaining non-homogeneous terms in the original equation by applying reciprocity principles and certain approximating functions. Its simplicity allows for implementation of numerical solutions for Laplace, Poisson, convection, convection-diffusion, and steady Navier-Stokes problems for moderate Reynolds numbers without changing the basic solution procedure.

The DRM can be characterized as a specialization of the BEM formulation where domain integrals are eliminated by introducing a superposition of localized particular solutions, which at the same time approximate the fundamental solution of the considered governing equation (Partridge et al. 1992). Take, for example, the 2D Poisson equation for an unknown function $u = u(x, y)$ in a closed domain or volume V (Fig. 1).

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = b, \tag{1}$$

where $b = b(x, y)$ is a function of position (x, y) that needs to be given together with enough conditions on the boundary $S = S_1 + S_2$ to be able to solve for u (n is the unit outward normal to S):

$$u = \overline{u} \quad \text{on} \quad S_1, \quad q = \partial u/\partial n = \overline{q} \quad \text{on} \quad S_2. \tag{2}$$

The superposition used in the DRM consists of a solution to Laplace's equation plus a series of particular solutions u_j^P in field points P so that

$$b = \sum_{j=1}^{N+L} \alpha_j \Delta u_j^P, \tag{3}$$

where N is the total number of boundary nodes after the discretization, L the number of field points and α_j a set of initially unknown coefficients. Substituting Eq.(3) into (1) gives

$$\Delta u = \sum_{j=1}^{N+L} \alpha_j \Delta u_j^P. \tag{4}$$

Further, for deriving the DRM integral equation, a weighted residual technique is applied whereby by differences to the correct solution of the governing equation and the boundary conditions are minimized over the considered volume and its boundary (see Partridge et al. 1992). Using this approach, Eq.(4) is multiplied by the fundamental solution of the Laplace equation u^* and integration over the domain produces

$$\int_V \Delta u \;\; u^* dV = \sum_{j=1}^{N+L} \alpha_j \int_V \Delta u_j^P \;\; u^* dV. \tag{5}$$

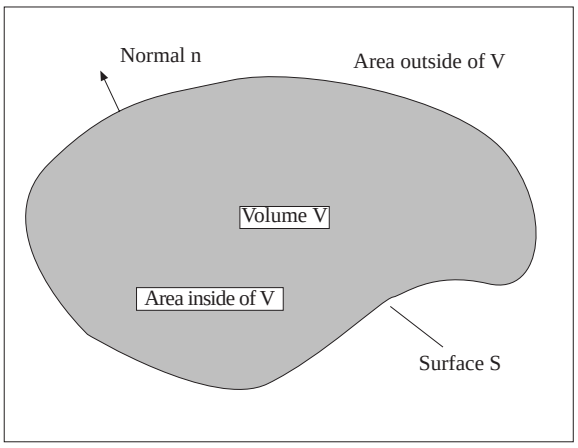

Figure 1. Definition sketch of a single bounded volume

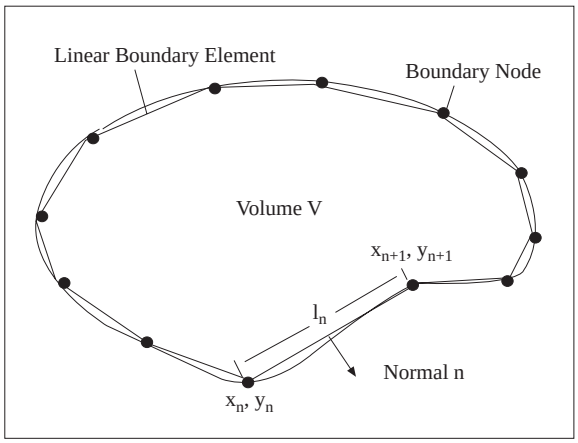

Figure 2. Boundary discretization with linear boundary elements $[n_x = (y_{n+1} - y_n)/\ell_n, \;\; n_y = -(x_{n+1} - x_n)/\ell_n]$.

Integrating by parts the Laplacian terms in Eq. (5) results in an integral equation for each source node i

$$c_i u_i + \int_S q^* u \; dS - \int_S u^* q dS = \sum_{j=1}^{N+L} \alpha_j \left(c_i u_{ij}^P + \int_S q^* u_j^P dS = \int_S u^* q_j^P dS \right), \tag{6}$$

where the term q_j^P represents the normal derivation of u_j^P defined by

$$q_j^P = \frac{\partial u_j^P}{\partial n} = \frac{\partial u_j^P}{\partial x}\frac{\partial x}{\partial n} + \frac{\partial u_j^P}{\partial y}\frac{\partial y}{\partial n}. \tag{7}$$

From this procedure the technique received its name dual reciprocity method: reciprocity has been applied to both sides of Eq.(5) to take all terms to the boundary. Thus, all domain integrals in Eq.(5) have been replaced by boundary integrals in Eq.(6). The term c_i is equal to the internal angle at node i divided by 2π, thus, for example, it has the value 0.5 for a node on a smooth boundary. After discretizing the boundary and replacing the boundary integrals by summations over N boundary elements, e.g. by linear boundary elements with a length ℓ_n (Brebbia 1978), the boundary integrals in Eq.(6) are replaced by finite sums over all boundary elements $E_n, \;\; n = \{1, 2, \ldots, N\}$, (Fig. 2).

Thus, Eq.(6) changes to

$$\begin{aligned} & c_i u_i + \sum_{k=1}^{N} \int_{S_k} q^* u dS - \sum_{k=1}^{N} \int_{S_k} u^* q dS \\ & = \sum_{j+1}^{N+L} \alpha_j \left(c_i u_{ij}^P + \sum_{k=1}^{N} \int_{S_k} q^* u_j^P dS - \sum_{k=1}^{N} \int_{S_k} u^* q_j^P dS \right). \end{aligned} \tag{8}$$

The evaluation of the boundary integrals on the left-hand side in form of

$$\sum_{k=1}^{N} \int_{S_k} q^* u dS = \sum_{k=1}^{N} H_{ik} u_k, \quad \sum_{k=1}^{N} \int_{S_k} u^* q dS = \sum_{k=1}^{N} G_{ik} q_k, \tag{9}$$

where k is the index for a boundary node used as field point, is well described in literature, for example in Brebbia (1984) for linear elements. These expressions are also applied to the right-hand side of Eq.(8) in

order to increase the method's efficiency, although this incorporates an error, but it has been shown to be small compared to the gained advantage. After inserting Eq.(9) on both sides of Eq.(8) one gets

$$c_i u_i + \sum_{k=1}^{N} H_{ik} u_k - \sum_{k=1}^{N} G_{ik} q_k = \sum_{j=1}^{N+L} \alpha_j \left(c_i u_{ij}^P + \sum_{k=1}^{N} H_{ik} u_{kj}^P - \sum_{k=1}^{N} G_{ik} q_{kj}^P \right), \tag{10}$$

or shorter in matrix notation

$$Hu - Gq = \sum_{j=1}^{N+L} \alpha_j \left(Hu_j^P - Gq_j^P \right). \tag{11}$$

where the terms c_i in Eq.(10) have been shifted to the principal diagonal of H in Eq.(11). If each of the vectors u_j^P and q_j^P are considered to be one column of the matrices U^P and Q^P respectively, Eq.(11) may be written without summation as

$$Hu - Gq = (HU^P - GQ^P)\alpha. \tag{12}$$

This equation is the starting point of the DRM solution. To evaluate the vector α one has to go back to Eq.(3) in the form of

$$b = \sum_{j=1}^{N+L} \alpha_j \Delta u_j^P = \sum_{j=1}^{N+L} \alpha_j f_j, \tag{13}$$

where the series of particular solutions is replaced by a set of approximating functions f_j, which can be transferred to a matrix F that easily allows to compute α by using the known function b as follows:

$$b = F\alpha \Rightarrow \alpha = F^{-1} b. \tag{14}$$

The best results gained so far have been made with functions of type

$$f_j = 1 + r_j + r_j^2 + \cdots + r_j^m, \tag{15}$$

where r_j is the distance between a source and field point. The corresponding Laplacian function and its gradient are

$$u_j^P = \frac{r_j^2}{4} + \frac{r_j^3}{9} + \cdots + \frac{r_j^{m+2}}{(m+2)^2}, \tag{16}$$

$$q_j^P = \left(\frac{\partial r_j}{\partial x}\frac{\partial x}{\partial n} + \frac{\partial r_j}{\partial y}\frac{\partial y}{\partial n} \right) \left(\frac{1}{2} + \frac{r_j}{3} + \cdots + \frac{r_j^m}{(m+2)} \right). \tag{17}$$

Now one is able to solve Eq.(12) by reducing it to the form

$$Ax = y, \tag{18}$$

which consists of N equations for N unknowns, either potentials or gradients, but at least one potential value must be given to scale the problem. Then, potentials and gradients for field points can be computed from Eq.(10) with $c_i = 1$. An alternative to compute the gradients, which is valid for any governing equation, is to take advantage of Eq.(14) applied to the problem variable u such that

$$u = F\beta \Rightarrow \beta = F^{-1} u. \tag{19}$$

The partial derivatives of the left-hand equation are

$$\frac{\partial u}{\partial x} = \frac{\partial F}{\partial x}\beta, \quad \frac{\partial u}{\partial y} = \frac{\partial F}{\partial y}\beta. \tag{20}$$

By inserting the right-hand equation of (19) into (20), one gets as final result

$$\frac{\partial u}{\partial x} = \frac{\partial F}{\partial x} F^{-1} u, \beta, \quad \frac{\partial u}{\partial y} = \frac{\partial F}{\partial y} F^{-1} u, \tag{21}$$

for the wanted gradients of u. The analog integral equation to (12) for the general Poisson equation

$$\Delta u = \left(a \frac{\partial u}{\partial x} + b \frac{\partial u}{\partial y} \right) (cx^m + dy^n) + e, \tag{22}$$

where a, b, c, d, e, m and n are known constant coefficients, is given by

$$Hu - Gq = S \left[\left(a \frac{\partial F}{\partial x} + b \frac{\partial F}{\partial y} \right) (cx^m + dy^n) F^{-1} u + e \right] \tag{23}$$

with

$$S = (HU^P - GQ^P) F^{-1}. \tag{24}$$

The derivation of DRM integral equations for time-dependent problems is done in a similar way as for the Poisson equation. Considering e.g. the diffusion equation

$$\Delta u = \frac{1}{K} \frac{\partial u}{\partial t} \tag{25}$$

as the simplest time-dependent case with K as material constant, a comparison of Eq.(25) with Eq.(1) immediately shows that Eq.(3) needs to be replaced by

$$\frac{1}{K} \frac{\partial u}{\partial t} = \frac{1}{K} \dot{u} = \sum_{j=1}^{N+L} \alpha_j f_j, \tag{26}$$

so that Eqs.(12), (23) analogously change to

$$Hu - Gq = -\frac{1}{K} S \dot{u}, \tag{27}$$

which allows a direct implementation of a time-marching scheme. For example, a two-level time integration formula reads

$$S \frac{u(t_{n+1}) - u(t_n)}{t_{n+1} - t_n} + H \frac{u(t_{n+1}) - u(t_n)}{2} = G \frac{g(t_{n+1}) + q(t_n)}{2} \tag{28}$$

or better sorted

$$(2S + H\Delta t) u(t_{n+1}) - G\Delta t q(t_{n+1}) = (2S - H\Delta t) u(t_n) + G\Delta t q(t_n) \tag{29}$$

which can be integrated, if enough starting values $u(t_0), q(t_0)$ are given. In the same manner, using the fundamental solution to Laplace's equation, the transient convection-diffusion equation

$$D\nabla^2 u = v_x \frac{\partial u}{\partial x} + v_y \frac{\partial u}{\partial y} - Ku + \frac{\partial u}{\partial t} \tag{30}$$

can be treated, where D is the dispersion coefficient, v_x and v_y the components of a velocity field, and K the reaction coefficient. This procedure and applying Eqs.(19)-(21) gives a matrix equation of the form

$$Hu - Gq = S \left[\left(\frac{v_x}{D} \frac{\partial F}{\partial x} + \frac{v_y}{D} \frac{\partial F}{\partial y} \right) F^{-1} u - \frac{K}{D} u + \frac{1}{D} \dot{u} \right]. \tag{31}$$

Because all DRM integral equations (12), (23), (27), and (31) have the same structure, a numerical scheme can be easily designed and implemented in one single computer code. This was realized by the 'Object-oriented Boundary Element Program (OBEP) 1.0' for Windows^{TM} (Friedrich 1995). The program offers a visual interactive user interface which allows simple and fast pre-processing, computations, and post-processing to solve the considered problems.

4. Examples

4.1. Uniform Potential Flow Through two Parallel Walls

The first example deals with uniform potential flow through two parallel and horizontal walls (no flux condition at top and bottom boundary), and a potential difference of +1.0 between the left and right boundary. Every side of the considered square has a length of 1.0 (Fig.3). The boundary discretization uses four corner points with (x, y) coordinates $P_1(x, y) = (0, 0)$, $P_2(x, y) = (1, 0)$, $P_3(x, y) = (1, 1)$, $P_4(x, y) = (0, 1)$. The correct solution for internal potentials and gradient (flux) values is given by

$$u(x, y) = 1 - x, \quad q_x(x, y) = \frac{\partial u(x, y)}{\partial x} = 1, \quad q_y(x, y) = \frac{\partial u(x, y)}{\partial y} = 0. \tag{32}$$

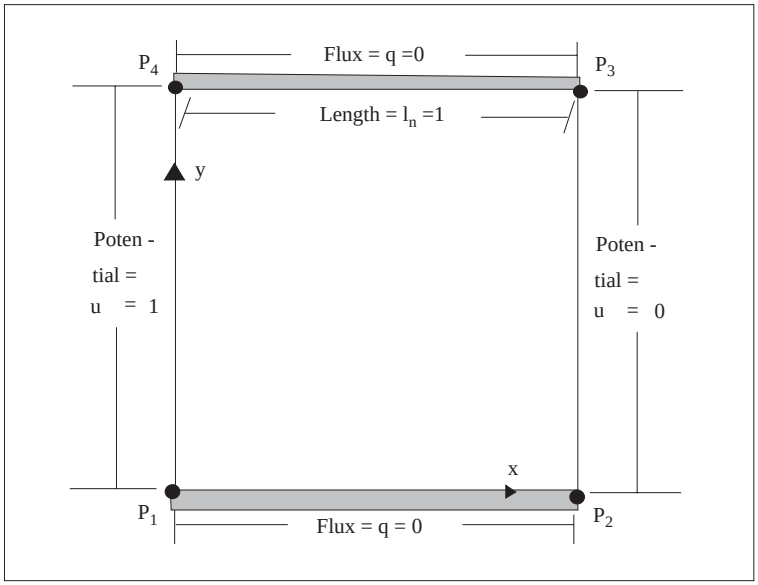

Figure 3. Potential flow in a square with four corner points

The potentials linearly decrease from one to zero when moving from the left to the right boundary whereas the gradient are constant everywhere in the volume, resulting in a uniform flow to the right. Using OBEP the results shown in Fig. 4a+b were obtained.

As can be seen from Fig. 4a, the OBEP results for the potentials fit the correct solution according to the left-hand side of Eq. (32) quite well with relative errors of approx. 1 %. The gradient results have also the same relative accuracy of about 1 % in the mid-region of the volume, which increases for field points closer to the boundary (Fig. 4b) because of the simplification made to evaluate Eq. (8) and the applied numerical integration scheme (4-point Gaussian quadrature formula) for computing the integrals in Eq. (9). These results also illustrate the known fact that gradients react more sensible to singularities than potentials because their order is larger by a factor of $(1/r)$ due to their definition.

4.2. Warping Function of an Elliptical Cross-Section

The second example considers a torsion problem for an elliptical cross-section $x^2/a^2 + y^2/b^2 = 1$ with a semimajor and -minor axis of $a = 2$ and $b = 1$. The governing Poisson equation is defined by

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -2, \tag{33}$$

where $u = u(x, y)$ is the torsion function to be determined (Partridge et al. 1992). As boundary conditions the torsion is taken to be zero on the whole boundary of the cross-section

$$\overline{u} = 0. \tag{34}$$

The correct solution for a shear modulus $\mu = 1$ and a twist angle $\phi = 1$ is given by

$$u(x, y) = -\frac{4}{5}\left(\frac{x^2}{a^2} + \frac{y^2}{b^2} - 1\right), \tag{35}$$

and the normal derivatives on the elliptical cross-sectionen by

$$q(x, y) = -\frac{1}{5}(x^2 + 8y^2). \tag{36}$$

The DRM solution for this torsion problem with OBEP brought forth the following results as shown in Figure 5a+b.

These results confirm the findings of the first example; the DRM technique is able to produce results with a relative error of about 1 % for both potentials and gradients, if the latter ones are located in the mid-region of the considered problem domain.

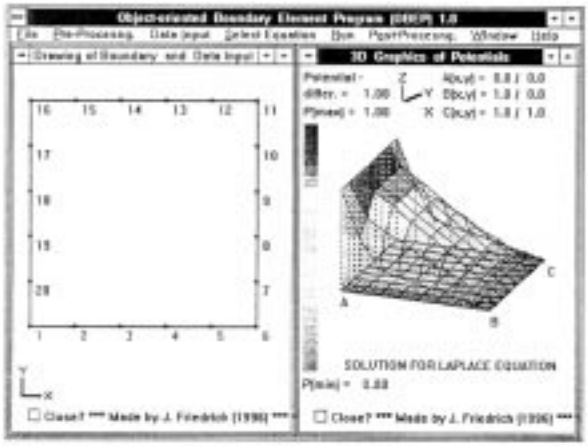

Figure 4a. Boundary and 3D plot of potentials

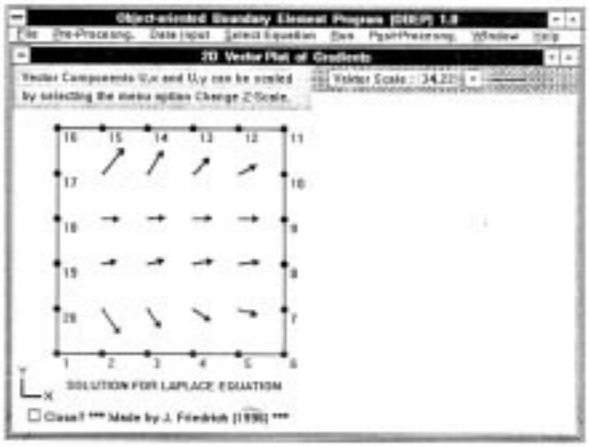

Figure 4b. 2D vector plot of gradients

4.3. Diffusion Problem for Cooling a Metal Piece

The last example is about the cooling process of a quadratic metal piece with a material constant of $K = 1.25$ and a side length of $1 = 3.0$ with coordinates $(x_{\min} = y_{\min} = 0.0)$, $(x_{\max} = y_{\max} = 3.0)$, that is cooled from a starting temperature $T_0 = 30°C$ down to $T(\Delta t) = 0.°C$ after a time period Δt. The analytical solution of this problem for a field point P at a position (x, y) is given with $K_x = K_y = K$ and $1_x = 1_y = 1$ by (Partridge et al. 1992)

$$\begin{aligned} T(\Delta t) \quad = \quad & \sum_{n=1}^{\infty}\sum_{m=1}^{\infty} \frac{4T_0}{nm\pi^2}[(-1)^n - 1][(-1)^m - 1] * \\ & \sin\frac{n\pi x}{\ell_x} \sin\frac{m\pi y}{\ell_y} \exp\left[-\Delta t\left(\frac{K_x n^2\pi^2}{\ell_x^2} + \frac{K_y m^2\pi^2}{\ell_y^2}\right)\right]. \end{aligned} \tag{37}$$

The analytical temperature gradients can be simply obtained by building the x and y derivatives of the sine terms in Eq. (37). The DRM solution of this diffusion problem governed by Eq. (25) with OBEP gave the following results as they are displayed in Fig. 6a+b after an integration time of $\Delta t = 0.150s$.

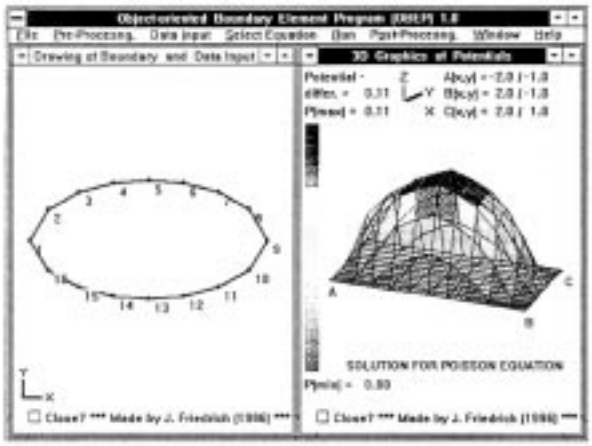

Figure 5a. Boundary and 3D plot of potentials

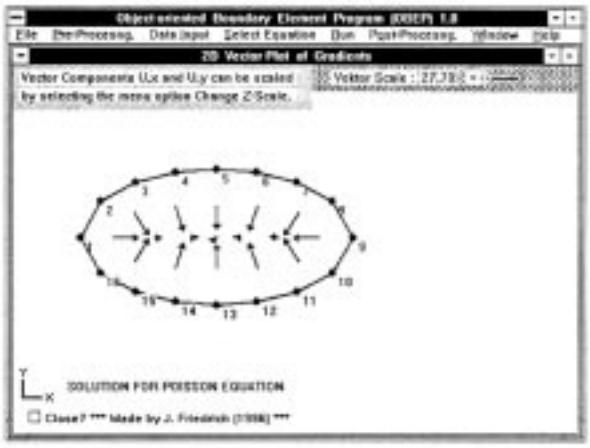

Figure 5b. 2D vector plot of gradients

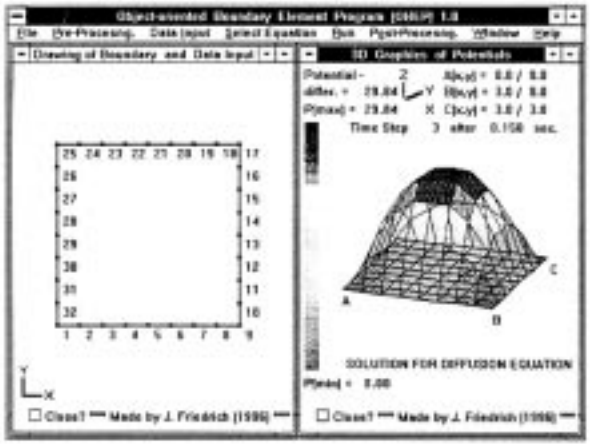

Figure 6a. Boundary and 3D plot of potentials

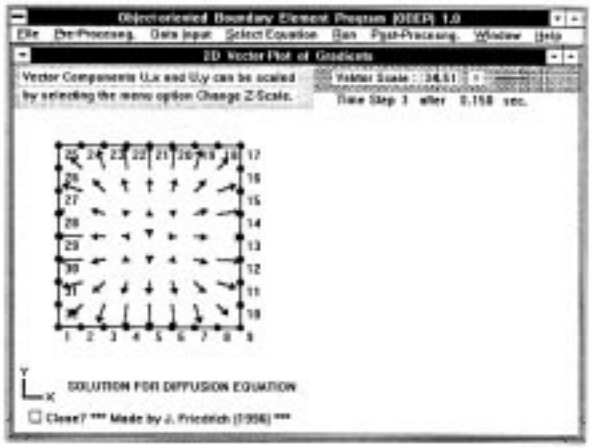

Figure 6b. 2D vector plot of gradients

As can be seen from these figures, the temperature distribution is symmetrical and the highest temperature is located at the center of the quadratic metal piece after the cooling started. Towards the boundary the temperature is declining where it becomes zero according to the boundary condition $T = 0.°C$. With a longer integration time the temperature constantly decreases and reaches zero after about $\Delta t = 30$. sec everywhere in the considered volume. This can be watched in movie style as OBEP prints new plots of both potentials and gradients at each time step on the screen. In this case, the computer simulation of the underlying physical system becomes much more realistic than it was the case for the two static examples solved before. Again the relative errors of temperature values and their gradients are about 1% in this application.

5. Conclusions

The aim of this paper was to create computer simulations of physical systems whose governing equation(s) need to be known together with sufficient information about boundary and geometrical conditions. The goal was realized by applying object-oriented technology and the dual reciprocity boundary element method which allows to integrate various types of partial differential equations by one solution procedure without major changes. The approach was demonstrated by three examples: Laplace's solution for a potential flow problem, Poisson's solution for a torsion problem, and the diffusion solution for cooling a metal piece. For time-dependent problems the advantages of the introduced method become much clearer; a user can observe onscreen how problem variables (e.g. potentials and gradients) are changing in time. By altering relevant system parameters a user can visually see the effects of such changes in the physical system. This is also very beneficial for solving inverse problems or optimizing designs by visual interaction when no mathematical approach exists to find directly such a solution or optimum.

References

[1] Benerjee, P.K. (1994) *The Boundary Element Method in Engineering*, McGraw-Hill, New York.

[2] Brebbia, C.A. (1978) *The Boundary Element Method for Engineers*, Pentech Press, London.

[3] Brebbia, C.A. (1984) *Boundary Element Techniques*, Springer Verlag, Berlin.

[4] Brebbia, C.A. und J. Dominguez (1989) *Boundary Elements: An Introductory Course*, McGraw-Hill, New York.

[5] Friedrich, J. (1995) The Advantages of Object-Oriented Modelling for BEM Coding demonstrated for 2D Laplace, Poisson, and Diffusion Problems using Dual Reciprocity Methodology, Proc. 10th Int. Conf. on Boundary Element Technology (BETECH 95), Comp. Mech. Publ. Southampton pp.229-236.

[6] McMonnies, A. and W.S. McSporran (1995) *Developing Object-Oriented Data Structures Using C++*, McGraw-Hill, New York.

[7] Partridge, P.W., C.A. Brebbia and L.C. Wrobel (1992) *The Dual Reciprocity Boundary Element Method*, Elsevier Science Publ. London.

[8] Pohl, I. (1993) *Object-Oriented Programming Using C++*, Benjamin Cummings Redwood City California.

[9] Power, H. and P.W. Partridge (1994) The Use of Stokes' Fundamental Solution for the Boundary Only Element Formulation of the Three-Dimensional Navier-Stokes Equations for Moderate Reynolds Numbers, Int. Journal for Numerical Methods in Engineering, Vol.37, pp.1825-1840.

[10] Power, H. and L.C. Wrobel (1995) Boundary Integral Methods in Fluid Mechanics, Computational Mechanics Publ. Southampton

[11] Rumbaugh, J.M. Blaha, W. Premerlani, F. Eddy and W. Lorensen (1991) Object-Oriented Modeling and Design, Prentice Hall Englewood Cliffs, New Jersey.