An Efficient Three-Dimensional FNPF Numerical Wave Tank for Large-Scale Wave Basin Experiment Simulation

Seshu B. Nimmala

Solomon C. Yim

School of Civil & Construction Engineering,
Oregon State University (OSU),
Corvallis, OR 97331

Stephan T. Grilli

Department of Ocean Engineering,
University of Rhode Island (URI),
Narragansett, RI 02882

This paper presents a parallel implementation and validation of an accurate and efficient three-dimensional computational model (3D numerical wave tank), based on fully nonlinear potential flow (FNPF) theory, and its extension to incorporate the motion of a laboratory snake piston wavemaker, as well as an absorbing beach, to simulate experiments in a large-scale 3D wave basin. This work is part of a long-term effort to develop a "virtual" computational wave basin to facilitate and complement large-scale physical wave-basin experiments. The code is based on a higher-order boundary-element method combined with a fast multipole algorithm (FMA). Particular efforts were devoted to making the code efficient for large-scale simulations using high-performance computing platforms. The numerical simulation capability can be tailored to serve as an optimization tool at the planning and detailed design stages of large-scale experiments at a specific basin by duplicating its exact physical and algorithmic features. To date, waves that can be generated in the numerical wave tank (NWT) include solitary, cnoidal, and airy waves. In this paper we detail the wave-basin model, mathematical formulation, wave generation, and analyze the performance of the parallelized FNPF-BEM-FMA code as a function of numerical parameters. Experimental or analytical comparisons with NWT results are provided for several cases to assess the accuracy and applicability of the numerical model to practical engineering problems. [DOI: 10.1115/1.4007597]

Keywords: numerical wave tank, three dimensional, wave-basin experiment, fully nonlinear waves, potential flow, piston wavemaker, high-performance computing, boundary element method, fast multipole algorithm

1 Introduction

Over the past decade, as modern computing platforms gradually increased in power, accurate and efficient three-dimensional (3D) computational wave basins (called numerical wave tanks or simply NWTs in this paper) have been under development. These NWTs are intended to simulate complex processes of ocean wave generation; propagation over arbitrary bottom topography; interaction with ocean structures; and dissipation over sloping beaches. The NWTs will be useful in the development, design, and analysis of many ocean engineering systems including offshore platforms, vessels and crafts, wave-energy conversion devices, and coastal infrastructure (breakwaters, piers, and docks). Until recently, the analysis and design of the complex ocean processes and engineering systems had been mostly investigated by performing laboratory experiments in large-scale 3D (or directional) wave basins, which are both expensive and time consuming to operate. To complement such facilities, NWTs can be used to accurately duplicate the exact physical operation of large-scale wave basins, including wave-generation algorithms, to simulate and optimize planned physical experiments ahead of time, and thus allow the users to more efficiently devote time and efforts to targeted laboratory experiments. As an added advantage, once validated, the NWTs can simulate time series of detailed flow parameters (e.g., velocity, pressure) everywhere in the numerical model, while these are usually available only at a limited number of experimental probes (and at the sacrifice of flow-field intrusion) at physically accessible locations in laboratory experiments.

It is beyond the scope of this paper to provide an exhaustive literature review of the many methods that have been used to develop NWTs. We will only present and discuss a limited number of references, targeted to the type of models used in our work, i.e., models simulating nonlinear waves based on inviscid fully nonlinear potential flow (FNPF) theory, and implemented based on a higher-order boundary integral equation (BIE) method, in a finite-element (FEM) formalism, which is referred to as the boundary-element method (BEM). Besides its numerical efficiency and accuracy, the main advantage of the BEM in engineering applications is that the dimensionality of the discretized problem is reduced by one. Thus, 3D problems can be discretized using a surface-only (i.e., two-dimensional) mesh, which reduces the effort devoted to developing relevant numerical grids. Additionally, while the governing equation (here Laplace's equation) is satisfied only approximately over the 3D-BEM domain boundary, it is satisfied exactly within the domain. Due to the reduced dimensionality, the numerical solution can be computed efficiently even for higher-order schemes and highly resolved BEM surface grids. Hence, problems such as free-surface waves can be solved very accurately. Finally, if required, it is easier to regrid the (2D) boundary mesh, unlike (3D) domain-discretization based methods (e.g., FEM). This is particularly useful for moving-boundary problems (e.g., free-surface waves), wherein regridding will be redistributing nodes evenly during wave propagation. The main drawback of the standard BEM, however, is that it yields nonsymmetric and fully populated linear system matrices, which for large problems becomes prohibitive to solve, and thus require fast solution methods or a more advanced implementation that

Contributed by the Ocean Offshore and Arctic Engineering Division of ASME for publication in the JOURNAL OF OFFSHORE MECHANICS AND ARCTIC ENGINEERING. Manuscript received May 20, 2012; final manuscript received September 6, 2012; published online February 25, 2013. Assoc. Editor: Daniel T. Valentine.

creates sparse matrices. This aspect of improving the numerical efficiency of a 3D-BEM-NWT is one of the main aspects of this paper that is extensively discussed later.

Historically, the BEM has been studied from the perspectives of classical boundary integral equations, mathematical analysis, and engineering applications (see, e.g., [1–7]). Following recent developments and advances in high performance computing (HPC), however, the BEM is receiving a renewed attention from various research communities resulting in new interesting practical developments and applications of the methodology.

More specifically, in the context of the present work, we present the parallelization and more efficient numerical implementation, of an existing 3D-NWT, on small to medium size HPC platforms consisting of multicore systems with a large shared central memory. The solution technique used, the fast multipole algorithm (FMA; see, e.g., [8–12]), makes the 3D-BEM much more efficient when using large-size discretizations required when solving practical engineering problems (which may require $O(10^5$ to $10^6)$ BEM nodes). The FMA is an important and relatively recent development (particularly in the context of NWTs) that has brought the BEM to the forefront of numerical methods used in such problems. The principle of the FMA is first to approximate the free-space Green's function of the problem governing equations by a series of spherical harmonics. Then, for each BEM discretization node, a hierarchy of increasingly distant subdomains is defined, in which the full Green's function is used only in the nearest subdomains, and a decreasing number of harmonics are used to represent the Green's function (down to no harmonics), for increasingly distant subdomains. Doing so both accelerates the computation (through numerical integration) of the nonzero coefficients in the BEM algebraic system matrices and creates large empty blocks in such matrices for subdomains beyond a cutoff distance; thus yielding a sparse structure. When properly implemented, a BEM-FMA code can achieve an N log N numerical complexity (with N the boundary discretization size). References [9,10] include several numerical examples in elastostatics, BEM-FEM coupling, and elastodynamics, in which grid sizes up to millions of nodal unknowns were used, and Ref. [11] presents a variety of practical engineering applications using FMA-BEM models. Reference [12] reports on the initial (scalar) implementation and application of the FMA method in the 3D-NWT used in this work. Note that an alternative numerical algorithm also based on the boundary-element method, but using a precorrected fast Fourier transformation (BEM-PFFT) to accelerate the evaluation of the far-field influences of source and/or normal dipole distributions [13], also achieving an N log N efficiency, has been recently implemented by Yan and Liu [14]. Details of the implementation of this algorithm can be found in a Ph.D. thesis by Yan [15].

2 Background of FNPF Theory and Free-Surface Waves Modeling

Potential flow theory solves inviscid, incompressible, Euler equations for irrotational flows. The governing equation for potential flows is mass conservation, which is expressed as a Laplace's equation for the potential, i.e., a second-order linear elliptic partial differential equation. Nonlinearity in wave processes originates from the presence of, and equations governing, the free surface, i.e., the dynamic (DFSBC) and kinematic (KFSBC) free surface boundary conditions. When full nonlinearity is kept in the latter, this yields FNPF equations.

Earlier works in applying the FNPF theory to modeling various strongly nonlinear near shore waves [16,17] indicate that it is accurate outside of the surf zone, up to the breaking point, where viscous effects are usually negligible, and hence vorticity is not generated, except in thin bottom and free-surface boundary layers. Additionally, in the presence of submerged or floating ocean structures, viscous effects are also negligible for large-scale bodies (such as ships); for small bodies (e.g., pipelines), while viscosity may be locally important for the flow around the structure, it will typically be negligible for the large scale (or far-field) wave flow itself. The assumption of incompressible fluid is valid when there is no air-water mixing (i.e., no bubbles), which is mostly the case for non-breaking waves. Accordingly, most results of classical wave dynamics and applications to date have been based on FNPF theory, or on other equations derived from it through perturbation expansions (e.g., Stokes waves [18]). One severe limitation of the FNPF theory, however, is that wave overturning and breaking will cause flow (i.e., breaking jet) penetration, which violates the governing equations and thus interrupts computations based on this flow model. Hence, in FNPF-NWTs, numerical absorbing beaches have been developed and used to prevent waves from overturning, through the absorption of the energy of steeper waves, usually by specifying an "absorbing" pressure distribution on the free surface; more details on this aspect of NWT simulations are given later.

In the 2D-FNPF models (and NWTs) initially developed [19–25], both the solution method (based on a direct Gaussian elimination scheme) and the computation of the matrix elements limited the size of problems that could be handled. For the integration of free-surface boundary conditions (i.e., time updating), a fourth-order multistep implicit Adams–Bashforth–Moulton predictor-corrector scheme (ABM4, initialized by a fourth-order Runge–Kutta (RK4) scheme), or an explicit second-order Taylor expansion scheme, were used. Regridding, combined in some cases with smoothing techniques, was employed to minimize numerical (sawtooth) instabilities.

With regard to fluid-structure interactions, using a 2D-FNPF theory, Lin [22] studied the nonlinear behavior of the flow near the intersection point of a free surface and a floating body (represented by a piston wavemaker). A numerical algorithm was proposed to accommodate the singularity resulting from satisfying conflicting boundary conditions near the intersection point. This aspect was revisited by Grilli and Subramanya [25] in the context of a different BEM model implementation, which used double nodes with specific continuity and compatibility conditions, and extended to the generation of waves by paddle flap or piston wavemakers [24,25].

As part of a pioneering project on 3D numerical simulations of nonlinear water waves, several studies [26–29] generated substantial research based on using the BEM and FNPF approaches. This work confirmed that, for large objects with characteristic dimensions on the order of one wavelength, viscous as well as compressibility effects can be neglected in the fluid flow: While the assumption that the flow is potential is not valid at all stages of physical processes, it is valid for many practical engineering situations. A detailed discussion of mathematical properties, such as existence, uniqueness, and well-posedness of BIE-BEM problems can also be found in these references. In these early 3D-NWT models based on FNPF, the time updating scheme was based on a RK4 scheme. The BEM was a low-order panel method. Romate [26] used a conjugate gradient square (CGS) algorithm for solving the linear system of discretized BIE equations at a given time step. The results were accurate and stable for linear and weakly nonlinear waves, but displayed significant stability problems in the case of highly nonlinear waves. Broeze [27] focused on improving the panel method developed in [26] for highly nonlinear wave problems. The CGS algorithm was used in this work, with several additional optimizations in the matrix-vector multiplications (which have a N^2 numerical complexity). It was found that more iterations were needed for solving larger problems, due to the occurrence of matrix conditioning issues, particularly when using large length to depth ratios in the computational domain (this typically occurs in wave propagation problems in NWTs representing near shore regions). In such cases, the increase in computational time will be worse than quadratic in the number of unknowns. The generalized minimal residual (GMRES) method was tried and found to be faster and have better convergence, but it required more memory than CGS. Memory limitation is much less of a problem nowadays, and our 3D-NWT uses GMRES with good success, as the basic method of solution. One obvious

disadvantage of panel methods over the BEM is that the geometry and solution are known only at discrete collocation points in the panels. This necessitates additional techniques to find the solution at the intersection of various surfaces. (For completeness, other early works should be noted for interesting contributions. Liu et al. [30] compared BIE solutions of 2D nonlinear water wave problems using free space or periodic Green's functions. Cooker [31] further develops the method in [21] for 2D nonlinear wave propagation over irregular beds; discretization is only needed on the free surface, resulting in a faster solution method; lateral periodicity, however, is required.)

In a series of research papers that represents the basis for the present work [32–35], Grilli and co-workers developed an accurate and versatile 3D-NWT based on FNPF theory, which directly extended their earlier 2D work (e.g., [17,24,25]). This 3D-NWT was initially based on combining a third-order BEM, based on a free-space Green's function in the physical space, with an explicit time updating based on second-order Taylor series expansions, in a mixed Eulerian–Lagrangian (MEL) formulation [32,33]. To improve the computational efficiency, the method was later extended [12,34,35] to using the fast multipole algorithm. The method used to solve the BEM algebraic system was GMRES. In applications of the BEM-FMA model, the numerical complexity was found to be nearly proportional to the problem size N. Wave energy dissipation was accomplished in the 3D-NWT by using an absorbing beach, which combines free-surface (pressure) damping and a lateral active absorbing wavemaker. This initial work and applications to the propagation of highly nonlinear waves in a 3D-NWT was summarized in a recent book chapter [35].

Despite its demonstrated success for academic-type applications, the range of engineering applications that could practically be solved using the 3D-NWTs developed by Grilli and coworkers, however, those models were limited by the size of problems that could be solved on a single processor machine. This was because both the FMA and the BEM had been implemented in scalar mode in the original model. Also, while wavemakers had been used in a few applications [34], the FMA implementation of the 3D-NWT neither feature a snake piston wavemaker for simulating laboratory directional wave generation (e.g., such as in Oregon State University's (OSU) 3D wave basin), which is an important requirement to be able to model water waves in a physical wave basin, nor was there a fully operational absorbing beach (AB), as used in 2D applications.

In this work we report on recent improvements in the 3D-NWT implementation and validation by (i) parallelizing the code for multicore shared memory computers; (ii) adding a snake piston wavemaker for arbitrary wave generation (with the development of a wave generation module consistent with the control software driving the actual wavemaker at OSU's wave basin); and (iii) an efficient absorbing beach. These improvements bring the 3D-NWT model to the level of a practical engineering tool, able to simulate meaningful laboratory experiments in a large-scale wave basin, on a moderate size HPC platform. This is demonstrated by comparing numerical results for 3D wave generation and propagation to laboratory measurements.

In the following we first present a summary of the mathematical model and numerical algorithms underlying the FNPF 3D-NWT, and then we describe the enhancements implemented into the NWT and their systematic validation for the generation of solitary waves and periodic waves (cnoidal and airy waves), which are some of the most fundamental wave generation capabilities of directional wave basins (e.g., the one at OSU). Applications are presented, with a comparison of numerical results to experimental data and theoretical solutions. Finally, a summary discussion of the NWT parallelization (particularly FMA-BEM) and resulting performance characterization demonstrating the relevance of the newly added capabilities is presented. This work is part of a long-term effort to develop a "virtual" numerical wave basin to facilitate and complement large-scale physical wave basin experiments.

3 Mathematical Model

As indicated before, under FNPF theory, the governing differential equation, i.e., mass conservation, is governed by the Laplace equation. With ϕ denoting the velocity potential and the velocity given by $\mathbf{u} = \nabla\phi = (u, v, w)$, this reads

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

Figure 1 shows the general setup and typical geometry of the 3D-NWT, in the case of wave generation by a snake piston wavemaker on the leftward boundary Γ_w, with stroke function $x_p(y,t)$. Wave elevation $\eta(x,y)$ on the free surface boundary Γ_f is defined with respect to the still water level, corresponding to the (x,y) plane (that is $z=0$). The tank axis is at $y=0$, and x is positive rightward, in the initial direction of wave propagation away from the wavemaker. As in OSU's directional wave basin, we assume here that the four sides of the tank are vertical boundaries, one of these being the moving snake wavemaker and the opposite end, parallel to the (y,z) plane, a moving absorbing piston boundary Γ_{ap}. The two sidewalls Γ_s, parallel to the (x,z) plane, are fixed in location. The impermeable bottom boundary Γ_b is represented in the figure with a constant depth h_0, but can be specified to be sloping or with an arbitrary topography.

On the free surface Γ_f, the potential satisfies the nonlinear kinematic (KFSBC) and dynamic (DFSBC) boundary conditions,

$$\frac{D\mathbf{R}}{Dt} = \mathbf{u} = \nabla\phi \tag{2}$$

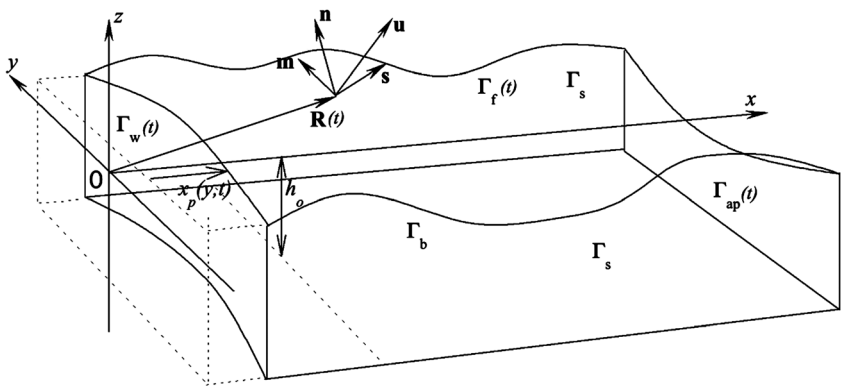

Fig. 1 Sketch of 3D-NWT geometry and parameters, for wave generation by a snake piston wavemaker (notation and details of mathematical model can be found in Sec. 3)

$$\frac{D\phi}{Dt} = -gz + \frac{1}{2}\nabla\phi \cdot \nabla\phi \tag{3}$$

respectively, with $\mathbf{R}(t)$ the position vector of a fluid particle on the free surface, g is the acceleration due to gravity, and D/Dt is the material (or Lagrangian) derivative. Both conditions are required since there are two unknowns (position and potential) on the free surface. The KFSBC states that the normal velocity is equal to the normal fluid velocity at the surface, when following a fluid particle at the free surface (this means that such particles remain on the free surface). The DFSBC, obtained from Bernoulli's equation (i.e., an integration of the Euler equations), states that the pressure on the free surface equals the atmospheric pressure, which is assumed here to be zero for simplicity.

For fixed boundaries including rigid vertical sidewalls and tank bottom, a no-flow condition (zero flux) is specified as $\partial\phi/\partial n = 0$, where $\mathbf{n}$ is the normal vector to the surface (pointing outside the fluid).

For a moving (piston) wavemaker boundary, both the motion (stroke function) and velocity are prescribed based on waves to be generated, by way of a wavemaker theory, as

$$x = x_p(y,t), \quad \frac{\partial\phi}{\partial n} = \mathbf{u}_p \cdot \mathbf{n} \tag{4}$$

where x_p and $\mathbf{u}_p$ are the wavemaker stroke and velocity, respectively

The initial free surface boundary condition (at $t=0$) is given by specifying a cold start in the NWT, with a still water level ($z=0$) and zero potential (Dirichlet BC), and all other surfaces having zero (i.e., vertical sidewalls and tank bottom) or specified normal fluxes (Neumann BC).

4 Numerical Implementation

A higher-order BEM is used to solve Eq. (1), whereby Green's second identity is applied to transform Laplace's equation into a BIE, which is discretized on the boundary. The numerical implementation is summarized below. For details, see [32,33,35].

4.1 Governing Equation. The BIE representation of Eq. (1) reads

$$\alpha(\mathbf{x}_l)\phi(\mathbf{x}_l) = \int_{\Gamma(t)} \left[\frac{\partial\phi}{\partial n}(\mathbf{x})G(\mathbf{x},\mathbf{x}_l) - \phi(\mathbf{x})\frac{\partial G}{\partial n}(\mathbf{x},\mathbf{x}_l)\right] d\Gamma \tag{5}$$

where Γ denotes the boundary of the fluid domain, $\alpha(\mathbf{x}_l)$ is function of the exterior solid angle made by the boundary at the collocation point $\mathbf{x}_l$ [32], and the 3D free space Green's function for Laplace's equation is defined as

$$G(\mathbf{x},\mathbf{x}_l) = \frac{1}{4\pi r} \tag{6}$$

where $r = |\mathbf{x} - \mathbf{x}_l|$ is the distance from the source point $\mathbf{x}$ to the collocation point $\mathbf{x}_l$ (both on the boundary). The normal derivative of the Green's function further reads

$$\frac{\partial G}{\partial n}(\mathbf{x},\mathbf{x}_l) = \nabla G \cdot \mathbf{n} = -\frac{1}{4\pi}\frac{\mathbf{r}\cdot\mathbf{n}}{r^3} \tag{7}$$

4.2 Time Integration. Second-order explicit Taylor series expansions are expressed for the free surface position and potential, updated to the next time step $t+\Delta t$, as a function of the solution at t,

$$\begin{aligned}\mathbf{R}(t+\Delta t) &= \mathbf{R}(t) + \Delta t\frac{D\mathbf{R}}{Dt}(t) + \frac{(\Delta t)^2}{2}\frac{D^2\mathbf{R}}{Dt^2}(t) + O\left\{(\Delta t)^3\right\}\\ \phi(t+\Delta t) &= \phi(t) + \Delta t\frac{D\phi}{Dt}(t) + \frac{(\Delta t)^2}{2}\frac{D^2\phi}{Dt^2}(t) + O\left\{(\Delta t)^3\right\}\end{aligned} \tag{8}$$

These expressions yield an explicit, stable, and efficient MEL time stepping scheme [32]. In these equations, zeroth-order coefficients are given by the free-surface geometry and potential at time t. First-order coefficients are evaluated from the free surface BCs (Eqs. (2) and (3)), also as a function of geometry and the BIE solution for ϕ and $\partial\phi/\partial n$ at time t. Based on Eqs. (2) and (3), second-order coefficients are expressed as

$$\begin{aligned}\frac{D^2\mathbf{R}}{Dt^2} &= \frac{D\mathbf{u}}{Dt} = \frac{\partial\mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} = \nabla\frac{\partial\phi}{\partial t} + \nabla\phi\cdot\nabla(\nabla\phi)\\ \frac{D^2\phi}{Dt^2} &= -g\frac{Dz}{Dt} = \frac{1}{2}\frac{D(\nabla\phi\cdot\nabla\phi)}{Dt} = -gw + \mathbf{u}\frac{D\mathbf{u}}{Dt}\end{aligned} \tag{9}$$

The expressions in Eq. (9) can be calculated as a function of geometry and the BIE solutions for both the potential and the time derivative of the potential at time t (i.e., $\partial\phi/\partial t$ and $\partial^2\phi/\partial t\partial n$). It should be emphasized that this second BIE solution uses the same system matrix as the first one and has boundary conditions, which can be calculated as a function of the solution of the first BIE (see Refs. [32,33] for details).

5 Salient Features of the Numerical Algorithms

Salient features of the numerical algorithms combining higher-order BIE solutions and an explicit time updating scheme can be summarized as follows (as above, see [32,33,35] for details).

- Second-order Taylor series coefficients, used in the time updating, are obtained from time derivatives of the boundary potential and flux, which are obtained from solving another BIE. Since this is performed using the same geometry as that for the first BIE for the potential, the same discretized BEM algebraic system matrix is used, with a different right hand side; hence the solution of the second BIE comes only at a moderate additional time cost [32].
- The time step is adapted as a function of the minimum distance between two nodes on the free surface, based on a constant mesh Courant number $C_0 \sim 0.45$ [25,32].
- The numerical solution of the algebraic BEM systems uses GMRES, wherein the matrix vector products are replaced by the fast multipole algorithm (FMA) for distant sources points, relative to a given collocation node [12,35]. This algorithm, which uses multipole expansions and tree data structures, avoids the full assembling of the discretized system matrix in memory. The theoretical computational complexity of the FMA is $O(N \log N)$, where N is the number of nodes on the boundary [12]. This is a very good improvement over the standard GMRES implementation, which results in an $O(N^2)$ performance. More details on the FMA implementation can be found in [12]. Here the FMA was modified to allow for a parallel implementation on computer systems with multicore nodes that share a large central memory (details are given later).
- The majority of the 3D-NWT code was developed in FORTRAN, while the FMA algorithms used libraries written in C language.
- The user's input to the 3D-NWT was designed to be minimal [32], in the form of the broad dimensions of the NWT and number of elements in each direction, plus a few other control and FMA parameters. The 3D surface mesh is automatically generated in the model, based on input parameters; thus resulting in a considerable time saving for the user.
- A node regridding technique can be automatically applied for any user-specified iterations so that free surface nodes are redistributed evenly over the grid. This option helps prevent inaccuracies and instabilities due to a very uneven distribution of nodes during wave propagation (which may occur due to Stokes drift for strongly nonlinear waves).

6 Piston Wavemaker Motion and Wave Generation

Boundary conditions for a plane (2D) piston wavemaker were derived in [24,25], for generating solitary cnoidal or other types of elementary waves. Fochesato et al. [34] extended this 2D wavemaker generation to arbitrary irregular waves, based on a target wave energy spectrum, and the possibility to correct the wavemaker stroke function for reflection in the NWT. In the 3D-NWT, Grilli and co-workers implemented a snake flap wavemaker to model 3D wave focusing (see [34]).

To demonstrate the practicality of the code and to more accurately generate shallow water waves, a snake piston wavemaker corresponding to the one at the OSU directional wave basin is implemented in the 3D-NWT. To ensure a match between experimental and numerical wave generation, the wavemaker paddle stroke function in Eq. (4) is essentially identical to those used to specify the motions of the mechanical actuators of the OSU directional wave basin. In the latter, the multidirectional (or snake) wavemaker has 29 rigid segments (or paddles) connected at the edges to 30 independently controlled actuators. While (for the purpose of validating the new 3D-NWT implementation) we will only present in this paper applications with long-crested 2D waves due to space limitation, more general and complex cases of wave generation will be reported in future work.

The wavemaker stroke were general functions of time t and lateral position y (there is no vertical variation for a piston wavemaker), with stroke $x_p(y,t)$ (Fig. 1), velocity: $|\mathbf{u}_p| = u_p = dx_p/dt$, and acceleration $a_p = du_p/dt$, so that an arbitrary snaking motion could be specified.

Let $\mathbf{n}$ be the outward normal vector to the piston wavemaker boundary, and ($\mathbf{s}$, $\mathbf{m}$) the tangential vectors with, $\mathbf{s}\cdot\mathbf{m}=0$ ($\mathbf{s}$ pointing in the global z direction; see Fig. 1) and $\mathbf{n}=\mathbf{s}\times\mathbf{m}$. We define

$$\phi_n = \frac{\partial\phi}{\partial n} = \mathbf{u}_p\cdot\mathbf{n}, \quad \phi_m = \frac{\partial\phi}{\partial m} = \mathbf{u}_p\cdot\mathbf{m}, \quad \phi_s = \frac{\partial\phi}{\partial s} = \mathbf{u}_p\cdot\mathbf{s} \tag{10}$$

with $\mathbf{x}_p = x_p\ \mathbf{i}$ and $\mathbf{u}_p = u_p\ \mathbf{i}$ (where ($\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$) denote unit vectors along the global x, y, and z axes, respectively). This yields the boundary condition for the second BIE problem as

$$\frac{\partial^2\phi}{\partial t\partial n} = \frac{D\mathbf{u}_p}{Dt}\mathbf{n} + \mathbf{u}_p\frac{d\mathbf{n}}{dt} + \phi_n(\phi_{ss}+\phi_{mm}) - \phi_m\phi_{nm} \tag{11}$$

Now, defining as (x_1, y_1, z) and (x_2, y_2, z), the coordinates of the edges of a plane individual waveboard in the snake piston wavemaker, we have the local vectors,

$$\mathbf{m} = (x_1-x_2)\mathbf{i} + (y_1-y_2)\mathbf{j}, \quad \mathbf{s} = \mathbf{k};$$
$$\mathbf{n} = -(y_1-y_2)\mathbf{i} + (x_1-x_2)\mathbf{j}$$

Finally, with these definitions,

$$\frac{d\mathbf{n}}{dt} = (\mathbf{u}_{p1}-\mathbf{u}_{p2})\mathbf{j} \tag{12}$$

since the actuators only move the wavemaker paddles in the x direction. In the 3D-NWT, derivatives with respect to n are obtained from the BIE solutions, while those with respect to s and m are analytically calculated using sliding fourth-order bipolynomial interpolation [32,33].

A new module (Wavegen) was implemented to generate the various stroke functions (x_p, u_p, a_p) required to simulate laboratory experiments in the OSU directional wave basin, for various type of standard waves, based on algorithms used in the OSU basin wavemaker driver software. At present, Wavegen offers three types of long-crested wave generation capabilities, which will be illustrated in the present applications: solitary, cnoidal, and airy waves. Hughes [36] presented various analyses related to wave generation using wavemakers, including more complex or nonlinear waves.

As shown in earlier work using a 2D- or this 3D-NWT [17,24,25,32,37–40], it should be noted that the wavemaker laws of motions used here are derived from linear or mildly nonlinear (i.e., Boussinesq) theories and; hence, would only generate strictly permanent form waves in a numerical model solving equations corresponding to these theories. Since the FNPF equations are solved here, only exact or numerically exact invariant wave solutions of the FNPF equations will keep an invariant wave form in the model. This was most extensively studied for (Boussinesq) solitary waves, such as the ones generated here, and it was shown that, for sufficient nonlinearity (usually for $H/h > 0.2$), wave height would start decreasing upon generation, during (inviscid) propagation over constant depth, while an oscillatory tail was shed behind the wave. For this matter, Goring [40] observed the exact same phenomenon in his wave tank experiments. To be able to propagate solitary waves of arbitrary nonlinearity in this and the earlier 2D-NWT, Grilli and co-workers had to specify these directly on the free surface based on Tanaka's [41] numerically exact solution. Similarly, to be able to propagate steady-state periodic waves in the NWT, a method for generating numerically exact fully nonlinear waves based on stream-function wave theory [18] was implemented. These phenomena will be illustrated in the present applications.

7 Verification of Generation, Propagation, and Absorption of Simple Waves

The original model, on which the 3D-NWT is based, has already been validated for a number of theoretical applications (e.g., for solitary waves) where both convergence and accuracy of the BEM solution were assessed as a function of mesh and time step parameters. None of these earlier validation tests, however, had been conducted for large 3D grids and with the solution performed using the new parallelized FMA algorithm. In addition, although the equivalent 2D-FNPF model had been validated for strongly nonlinear waves using experimental data, no such comparison of 3D-NWT results with detailed laboratory experiments in a large 3D wave basin, such as those available from the OSU directional wave basin, had been performed to date.

In this section we perform numerical simulations to verify that the new implementation of wave generation by a piston wavemaker, propagation over constant depth, and energy absorption by the "absorbing beach," perform as expected. Due to space limitation and for simplicity of presentation, in these validation applications, only simple long-crested (i.e., 2D) waves are generated; hence, the piston wavemaker moves as a whole, as discussed in the previous section. (Additional details of the parallel computational algorithm, snake wavemaker capability and 3D bathymetry effects, and corresponding speed-up studies, will be presented in future publications.) For such cases, the width of the NWT does not really matter and can be small, to save on computational time; we thus are dealing here with a narrow 3D-NWT. In the following computations, the 3D-NWT has a length of 25 m, a depth of $h = 0.75$ m, and a width of 1 m for solitary waves, or 0.5 m, for cnoidal or airy waves. These waves are generated at the wavemaker, propagate in the tank, and are absorbed at the far end in an absorbing beach (AB) of 3 m length (with an AB coefficient $\nu_o = 400$ kg/m s). Although the OSU directional wave basin has larger dimensions (length 48.8 m and width 26.5 m), the reduced dimensions (along length and width) in the model are acceptable, as already indicated, since we are only generating 2D waves (in the vertical plane) (Fig. 2). Results detailed below include a comparison with experiments and theoretical wave profiles.

A summary description of the 3D-NWT parallel implementation and assessment of model performance are given in Sec. 8. The computer platform used in all simulations in this section was a Dell Precision WorkStation 690, with 8 Intel Xeon 3 GHz

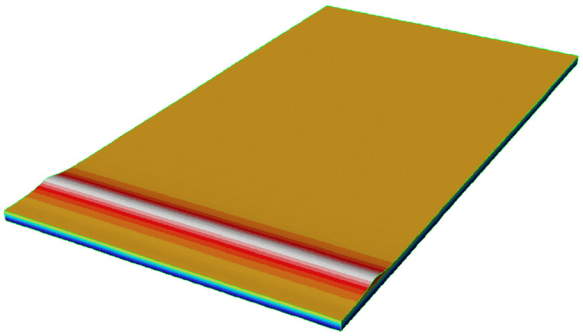

Fig. 2 Snapshot of 3D-NWT simulations for the propagation of a solitary wave over constant depth in a geometry identical to that of OSU's wave basin (48.8 m long, 26.5 m wide, 0.78 m deep)

processors, 64GB of shared RAM and a 64 bit Redhat Linux Enterprise 5.

7.1 Solitary Waves. In this simulation, the BEM grid has 200 elements along the length, 5 elements along the width, and 8 elements along the depth. This yields a total of $N = 6138$ nodes and $M = 5280$ elements. Two waves are generated, with targeted heights $H = 0.3$ and 0.45 m in depth $h = 0.75$ m, or $H/h = 0.4$ and 0.6, respectively (Fig. 3); hence these are strongly nonlinear waves selected to demonstrate the more challenging fully nonlinear capability of the numerical model. Results for $H = 0.3$ m, shown in Figs. 4(*a*) and 5(*a*), took 7.82 h of clock time to compute 13 s (600 time steps) of wave propagation. Results for $H = 0.45$ m shown in Figs. 4(*b*) and 5(*b*) took 7.91 h for a similar simulation. For both waves, numerical wave profiles, at the wavemaker and three wave gauges, and particle velocities at the third gauge (at a depth $z = -0.61$ m), are found to agree well with experimental results. Wave absorption (absorbing piston and absorbing pressure beach) at the end of the tank appears to be working well as no significant reflected waves can be seen in the wave gauge records. Note, as discussed above, the height reduction of waves as they propagate down the NWT and the physical basin, and the small oscillatory tails shed behind the waves.

7.2 Cnoidal Waves. Here the NWT-BEM grid has 200 elements along the length, 5 elements along the width, and 5 elements along the depth. This yields a total number of nodes $N = 4896$ and elements $M = 4050$. The targeted theoretical wave height is $H = 0.3$ m and period $T = 3.5$ s; the expected wavelength from cnoidal theory is $L = 9.84$ ($m = 0.989$); hence, $L/h = 13.1$, which corresponds to a fairly long (but still intermediate depth) wave, and $H/h = 0.4$, indicating a strongly nonlinear wave. Additionally, cnoidal theory predicts a trough depth of -0.08 m and a crest height of 0.22 m (Fig. 6). Results shown in Figs. 6–8 took 20.5 h, on the same 8 CPU computer, to compute 27.6 s of wave

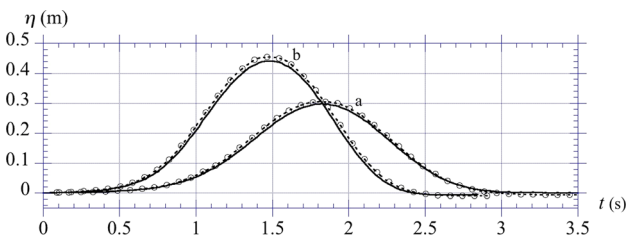

Fig. 3 Solitary waves of target height $H = (a)$ 0.3 m; (b) 0.45 m, in water depth $h = 0.75$ m. Surface elevations versus time at the wavemaker, in (—) numerical model; (-o-) OSU's 3D tank experiments (experimental data were shifted by a 0.04 s time lag; only 25% of experimental points are shown).

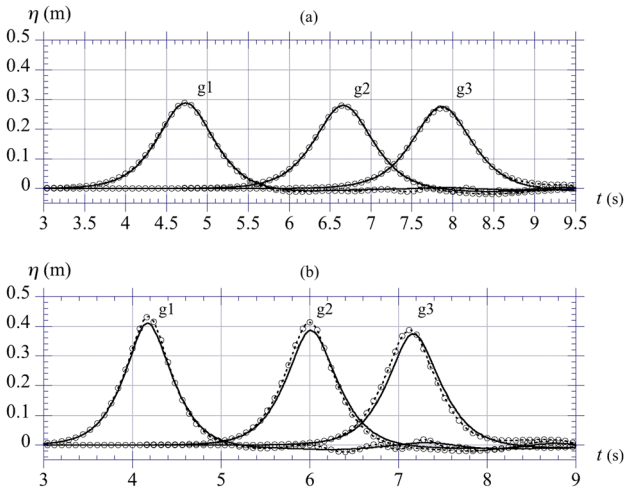

Fig. 4 Solitary waves of target wave height $H = (a)$ 0.3 m; (b) 0.45 m, in depth $h = 0.75$ m. Numerical (—) and experimental (-o-) surface elevations as functions of time, at three gauges at $x = 8.8$ m (g1), 14.9 m (g2), and 18.7 m (g3), with $y = 0$ (experimental data were shifted by a 0.16 s time lag; only 25% of experimental points are shown).

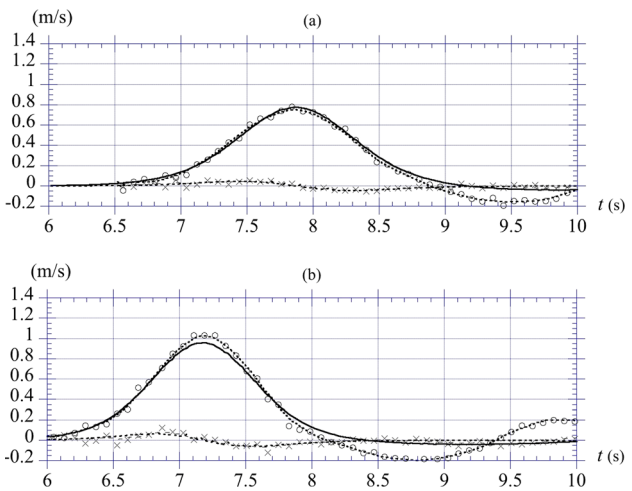

Fig. 5 Solitary waves of target wave height $H = (a)$ 0.3 m; (b) 0.45 m, in depth $h = 0.75$ m. Numerical (—) and experimental (-o-) water particle velocity components (u, w) as functions of time, at gauge g3 location: $x = 18.7$ m, $y = 0$, at depth $z = -0.61$ m (only 25% of experimental points are shown).

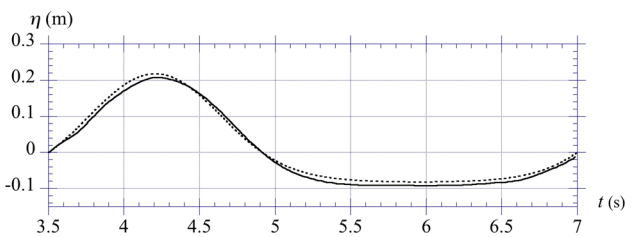

Fig. 6 Cnoidal wave of target height $H = 0.3$ m and period $T = 3.5$ s, in water depth $h = 0.75$ m. Numerical (—) and theoretical (- - -) surface elevations at the wavemaker as functions of time.

propagation. Figure 6 shows that the initial numerical wave profile at the wavemaker agrees well with well-known analytical solution. Figure 7 shows wave profiles computed at a few gauges, and Fig. 8 shows particle velocity components at one location and

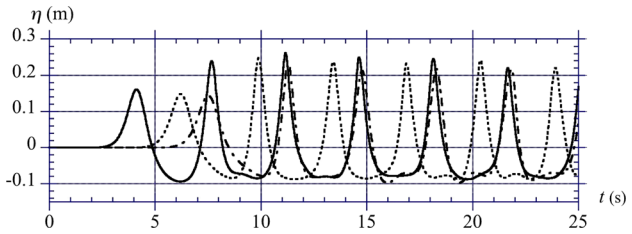

Fig. 7 Cnoidal wave of Fig. 6 numerical wave elevation as a function of time, at three wave gauges at x = (g1: —) 8.8 m, (g2: - - -) 14.9 m, and (g3: —.) 18.7 m, with $y = 0$

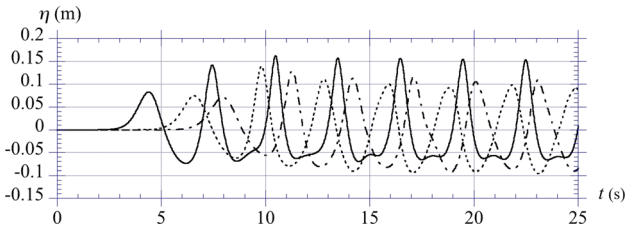

Fig. 10 Airy wave of Fig. 9 numerical surface elevation as a function of time, at three wave gauges at x = (g1: —) 8.8 m, (g2: - - -) 14.9 m, and (g3: —.) 18.7 m, with $y = 0$

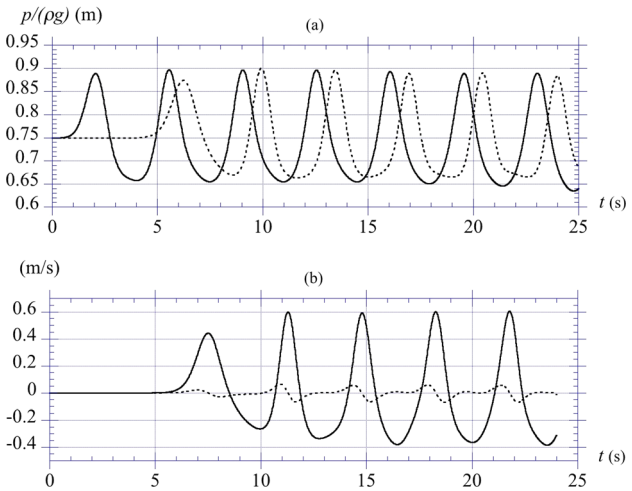

Fig. 8 Cnoidal wave of Figs. 6 and 7. Numerical: (*a*) bottom pressure as a function of time at two gauges at x = (—) 4 and (- - -) 16 m, with $y = 0$; (*b*) wave particle velocity components u (—), w (- - -) as functions of time at gauge g3, with $x = 18.7$ m, $y = 0$, at depth $z = -0.61$ m.

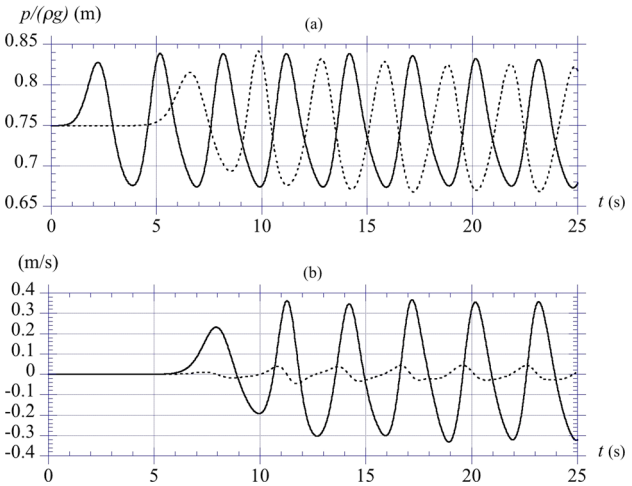

Fig. 11 Case of Figs. 9 and 10, numerical: (*a*) bottom pressure as a function of time at two gauges at x = (—) 4 and (- - -) 16 m, with $y = 0$; (*b*) wave particle velocity components u (—), w (- - -) as functions of time at $x = 18.7$ m, $z = -0.61$ m, $y = 0$

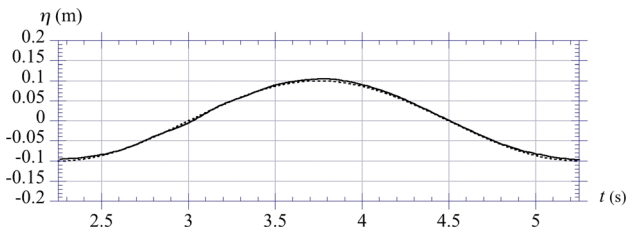

Fig. 9 Airy wave of target height $H = 0.2$ m and period $T = 3.0$ s, in water depth $h = 0.75$ m. Numerical (—) and theoretical (- - -) surface elevations at the wavemaker as functions of time.

bottom pressure at two gauges. The good periodicity of these parameters, once simulations have reached a quasi-steady state, indicates that wave absorption is working adequately in the NWT. Note, in Fig. 7, while both crest height and trough depth stay quite close to the expected values, small oscillations are present in each wave trough, which are a well-known indication of the generation of higher-order harmonics in the generation of strongly nonlinear waves using a low-order wave theory.

7.3 Airy Waves. The grid used here is the same as for cnoidal waves. The wave height is $H = 0.2$ m and period $T = 3.0$ s, yielding a linear wavelength $L = 7.68$ m; hence, $L/h = 10.2$, which corresponds to an intermediate wave, and $H/h = 0.27$ and $H/L = 0.026$, indicative of a fairly strongly nonlinear wave. Results in Figs. 9–11 took 30.5 h to compute, on the 8-CPU computer, for a 40.6 s maximum time. In Fig. 9, which presents a comparison of the numerical predictions with analytical solution, we see the initial wave shape at the wavemaker is, as expected, closely sinusoidal and trough/crest symmetric. Figure 10, however, shows that, as expected, the linear wave gradually adjusts its shape, to reach a permanent form profile consistent with wave nonlinearity. Thus, at the farther gauge down the NWT (g3), wave profiles have become fairly steady, with narrower and taller crests (at 0.15 m) and wider and shallower troughs (at −0.06 m). Dynamic pressure and wave particle velocities at locations far down the tank also become quasi-steady (Fig. 11). Wave absorption in the absorbing beach is thus also working well in this case.

8 Parallelization and Performance Studies

This section summarily reviews numerical features of the new 3D-NWT implementation, discusses alternatives explored for parallel computing, choices adopted, and presents performance studies and insight gained in the process.

8.1 Serial Version. The initial BEM algorithms of the 3D-NWT code [32,33] were implemented in scalar mode, using Fortran 77, which for a long time has been the dominant language in scientific computing. Thus, memory was statically assigned in common blocks during compilation, which allowed subroutines called in sequence to access the same data. One disadvantage of this approach for evolving codes (such as part of research projects), however, is that the data structure is spread all over the code and it is thus more difficult to modify parts of the model as other associated processes may directly depend on such modifications. With the addition of new features over several years (which is inevitable in evolving computational/numerical software, as

evidenced by publications [32,35]), this dependency between data and implementation became increasingly more complex to manage.

8.2 Alternatives in Parallelization and Implementation Using OpenMP. Consistent with the leading computing paradigm at the time, a vectorized version of the original code was implemented on a CRAY-90 computer using GMRES for the solution of linear algebraic systems. Although the computational efficiency was greatly improved, the size (discretization) of problems that could be solved was still very limited, due to the need to assemble the full BEM system matrix in the computer memory. The FMA implementation that was later developed for the BEM system solution [12], allowed both a significant gain in efficiency and the use of larger grid discretizations in applications [34,35]. However, this was still a scalar implementation. Additionally, the BEM and time stepping parts were written in FORTRAN while the FMA library was written in C. Hence, the initial scalar code used a hybrid compiler technology (GNU C compiler and Intel Fortran), with different paradigms of passing memory chunks from one part to another.

In order to parallelize the 3D-NWT with the FMA implementation, an off-the-shelf component was first tested in the code: The distributed parallel multipole tree algorithm (DPMTA) library [42]. This software was written in C language, using the message passing interface (MPI) parallelization protocol. Benchmark tests of the DPMTA indicated that, with proper parameter selection, it could yield nearly linear speedup with respect to the number of processors, for the same problem size, and also with increase in data size. In its original form, the 3D-NWT had a significant serial part in the computations (e.g., mesh generation/regridding, iterative scheme of solution at each time step, and also time stepping, until a user-specified number of time steps or a maximum time is reached). In order to gain maximum advantage from MPI, we first tried introducing a division of mesh data and work among processors, with a minimization of message passing. However, this would have required a major code rewriting and redesigning of the model flow chart, with the risk of coding errors and the requirement of extensive testing. An alternative was to entirely rewrite/design the code, using one of the latest (object oriented) software architectures, but this would have been prohibitive and at the risk of introducing errors.

Hence, for the NWT parallelization, we opted for a more pragmatic and efficient (in terms of developmental time and efforts), albeit less optimum, strategy and used OpenMP [43], rather than the DPMTA library (and MPI). Parallel codes using OpenMP work by creating a user-specified number of threads, which are independent sets of instructions within a process. All threads created by the same process have access to the same data in physical memory, without duplicating or transferring data (as in MPI); however, one has to make sure that threads do not write on the same physical location in memory at the same time. In order to achieve load balancing (i.e., each processor taking equal load), the maximum number of threads can be set to the number of processors, but this is not required. Note that using more threads than the number of processors can cause the system to become unresponsive to other tasks. A practical point of concern is that the overhead (including the creation of threads) for parallelization is not negligible. Hence, the size of and speed-up resulting from parallel components must be significant enough to overcome this overhead. In other words, the problem size (or model size in a simulation) should be large enough as to achieve good speedups. This will be illustrated and quantified in applications below.

Finally, to efficiently implement OpenMP, the blocks in the code that are major bottlenecks were identified by profiling. Parallelization code was limited to such areas, with careful modifications such that no two threads write to the same memory location at the same time.

8.3 Performance Studies. In this section we assess the performance of the new parallel implementation of the 3D-NWT code using OpenMP, in terms of computational efficiency measured by clock time on a multicore computer platform. Rather than using a large shared parallel computer cluster, it was decided to use a local desktop minicluster that was 100% available to the project. This was a Dell Precision WorkStation 690, with 8 Intel Xeon 3 GHz processors, 64GB of shared RAM and a 64 bit Redhat Linux Enterprise 5. Larger multiprocessor computer clusters could of course be used to achieve improved CPU times.

The FMA uses multipole expansions and tree data structure [6,11], and avoids the full assembling of the algebraic system matrix in memory. Results clearly showed that, in addition to problem size (i.e., N, M), the code run time was also influenced by the choice of FMA parameters, such as the number n of tree levels in the grid region subdivisions.

The applications used in this performance study correspond to wave generation/propagation in a 3D-NWT with the actual geometry of the OSU directional wave basin (Fig. 2): 48.8 m long and 26.5 m wide, with depth $h = 0.78$ m. For simplicity we used the canonical case of a strongly nonlinear solitary wave of height $H = 0.39\,\text{m} = 0.5\,h$, propagating over constant depth. In all cases, we used eight terms in the multipole expansion and computed for five time steps. This small number is aimed at speeding up computations for larger problems; all things being equal, the CPU time is simply proportional to the number of time steps. (This linear scaling phenomenon was verified by a number simulation runs with significantly longer duration for several model cases.) The FMA cube length (which should be specified to be at least twice the sum of the maximum dimension with an allowance for wave absorption [12]) covering the wave basin is 105 m. Further details, such as the number of tree levels, are provided below for each model application.

Figure 12 first shows the 3D-NWT performance for two small applications: 1 with $N = 2592$ nodes and 2 with $N = 13{,}838$ nodes. The number of FMA tree levels is selected to $n = 6$ to 7, respectively, to obtain the best CPU times and speed-ups with eight processors; speed-up is the ratio of CPU time using a single processor by that using multiple processors. We see maximum speed-ups

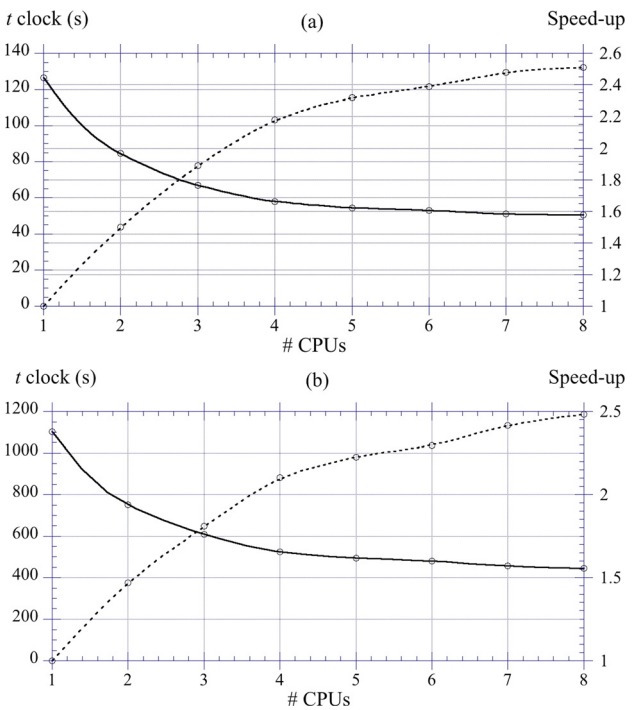

Fig. 12 Clock time (—) and speed-up (- - -) versus number of CPUs, for five time steps of 3D-NWT simulations (Fig. 2) for application: (*a*) 1, using $N = 2592$ nodes and $n = 6$ FMA tree levels; (*b*) 2, using $N = 13{,}838$ nodes and $n = 7$

achieved in these applications are 2.51 and 2.48, respectively (i.e., about one-third the number of processors). It is observed that, in these applications, speed-up is not markedly increasing beyond 5–6 processors. This is because speed-up depends on the size of parallel components, out of the total computation and associated overhead. As indicated, the code still has inherently sequential components and some parallelized ones; using additional processors only reduces the CPU time of the parallelized components. For smaller applications (such as here), an increase in the number of processors, beyond some number, does not cause a significant reduction in processing times because of the additional overhead this incurs (due to thread creation, setup, etc), which overcomes gains from parallel computation.

From a user's perspective this means that higher speed-ups will be achieved with a larger numbers of processors only for increased application size (which gives rise to larger parallelizable components in the code). However, for all cases, the maximum speed-up is limited to a maximum value by the sequential components in the code, which is a function of the number of processors. These observations are demonstrated in the first two applications in Fig. 12 (where the number of nodes in the second application is 6 times that of the first one, but both have about the same speedup for eight processors).

Figure 13 more broadly illustrates the performance of the parallel 3D-NWT using either one (i.e., scalar mode) or eight processors, for the same solitary wave problem, using many different grid sizes $N = 2592$ to 149,250 nodes. The FMA number of tree levels n is varied from 5 to 8, with increasing values of N. In all cases, faster clock times are achieved with eight processors but, for a given n value, the speed-up from parallelization gradually reduces beyond some threshold N value. Around the latter, clock time increases as $O(N \log N)$. This computational complexity rate was already identified for the scalar 3D-NWT using the FMA [12], for large enough N (usually more than 6000 or so), and is also visible on Fig. 13 for the one-processor curve with $n = 5$. In addition, when reaching the threshold N value of a given n, one can reduce the rate of clock time increase by increasing n. In Fig. 13, using eight CPUs, this occurs for $N = 10{,}000$, from $n = 6$ to 7; and for $N = 50{,}000$, from $n = 7$ to 8. We see that for the lower envelope of these various performance curves, the computational complexity is on average $O(N^{1.3})$, which is still very good. Hence, when increasing grid size N for a given number of CPUs, smaller clock times can only be achieved when increasing the FMA n levels.

Based on Fig. 13, a user of the 3D-NWT model can thus select a relevant FMA n level for a given grid size N, as a function of the number of processors, to achieve an optimal clock time. It is expected that similar curves could be derived using larger numbers of CPUs on larger computer cluster systems.

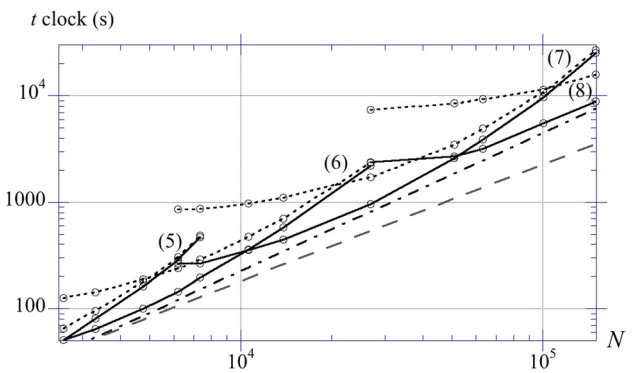

Fig. 13 Clock time versus number of BEM nodes *N* and CPUs: 1 (- - -) or 8 (—), for five time steps of 3D-NWT simulations for various application (o) (Fig. 2), using *n* = (5) to (8) FMA tree levels. The straight lines (—.) and (— —) indicate clock time: $\propto N^{1.3}$ and *N* log *N*, respectively.

8.4 Insight Gained. Some of the insight and experience gained during this parallelization effort are summarized below to serve as guidelines for other developers undertaking similar efforts.

8.4.1 FMA Input Parameter—Number of Tree Levels. This parameter indicates the number of hierarchical levels into which the "cube" (containing the discretization) is divided. The computing speed-up is considerably affected by this parameter, as the work shared by the threads is dependent on the number of cells in the hierarchy. The user should be able to experiment and determine the optimum n parameter for a specified number of threads and given problem size N. The practical values of this parameter range from 4 to 8. A much larger number of tree levels (> 8) would considerably increase the memory requirement and the run may not be able to fit within the shared memory resources of the computing system. This limitation, however, can be alleviated in view of the anticipated future availability of larger (and faster) hardware resources.

8.4.2 FMA Input Parameter—Cube Length. This is the cube size surrounding the entire geometry of the discretization. The cube is defined with the origin of the grid at its center. The user has to identify the maximum dimension, estimate the possible change in length (due to wave absorption for instance), and use twice this value for the cube length. One should use just as low a value as possible, for optimal performance of the FMA.

8.4.3 FMA Input Parameter—Number of Terms in a Multipole Expansion. The larger this number, the better the accuracy of the expansions, but the larger the computing time. While this parameter can range from 4 to 16, the practical value is around 6–8 (the latter was used here).

8.4.4 Distribution of Nodes. The distribution of nodes (or the grid geometry and number of nodes) in the FMA cube can also affect computing time. A study performed for one of the applications, using a cube size of 60 m instead of 120 m, showed a reduction in computing time from 19.76 to 8.53 h. The reason is that a cube size of 60 m caused nodes to be more uniformly distributed in FMA cells and thus reduced the computational effort.

9 Concluding Remarks

This paper presents improvements in the implementation of an existing computation model (3D-NWT), based on the FNPF theory, to make it an efficient tool to complement experimental facilities such as the OSU directional wave basin or elsewhere. The numerical algorithms and mathematical formulations involved in furthering the development of wave generation capabilities (i.e., for solitary, cnoidal, and airy waves) are provided. Comparisons with experiments for simple waves illustrate that numerical and experimental results are in good agreement and provide a strong basis for the use of the 3D-NWT code for all appropriate and practical engineering purposes.

The insight gained from the parallelization effort, along with the performance studies, provides guidelines for a more efficient use of the code and also could help with similar efforts on developing other numerical codes to be run on multicore processor clusters. The near-linear performance of the parallel algorithms in the 3D-NWT is demonstrated for a sufficiently large number of BEM nodes and a proper selection of FMA parameters. Hence, the BEM–GMRES–FMA–parallel capabilities, provide an "optimal" combination of tools for solving FNPF problems, such as 3D nonlinear free surface waves, in a reasonable computing time, on small (and larger) computer clusters.

More complex cases of wave generation, for fully nonlinear 3D waves and/or varying bathymetry in the NWT, will be reported in future work. Additionally, work on improving wave generation using wavemakers, to account for wave nonlinearity (not detailed here), was also conducted to derive the stroke motion necessary to

generate fifth order Stokes waves. Although this was successful in the NWT, currently there is no practical methodology to implement this capability in physical facilities. It should be noted that the generation of exact stream function waves using a "porous" piston wavemaker that was implemented in the 2D-NWT could be easily incorporated in the 3D-NWT. However, such a wavemaker cannot be simulated in a physical facility such as the OSU directional wave basin. Nevertheless, finding a relevant piston wavemaker motion that could create approximate fully nonlinear periodic waves in a physical facility could be investigated using the present improved 3D-NWT. Such an addition, which could be done in future work, would benefit both the numerical wave tank and corresponding laboratory wave basins.

Acknowledgment

The first two authors are thankful to the Office of Naval Research ONR-N00014-11-1-0094 and the Department of Energy DOE-G0107G for the financial support that made this work possible.

References

[1] Banerjee, P. K., and Butterfield, R., 1981, *Boundary Element Methods in Engineering Science*, McGraw-Hill, London.
[2] Yeung, R. W., 1982, "Numerical Methods in Free-Surface Flow," Annu. Rev. Fluid Mech., **14**, pp. 395–442.
[3] Brebbia, C. A., Telles, J. C. F., and Wrobel, L. C., 1984, *Boundary Element Techniques: Theory and Applications in Engineering*, Springer, Berlin.
[4] Grosenbaugh, M. A., and Yeung, R. W., 1989, "Nonlinear Free-Surface Flow at a Two-Dimensional Bow," J. Fluid Mech., **209**, pp. 57–75.
[5] Kanwal, R. P., 1997, *Linear Integral Equations—Theory and Technique*, Birkhauser, Boston.
[6] Hsiao, G. C., and Wendland, W. L., 2008, *Boundary Integral Equations*, Springer, Berlin.
[7] Chen, G., and Zhou, J., 2010, *Boundary Element Methods With Applications to Nonlinear Problems*, World Scientific, Singapore.
[8] Nishimura, N., 2002, "Fast Multipole Accelerated Boundary Integral Equation Methods," Appl. Mech. Rev., **55**(4), pp. 299–324.
[9] Margonari, M., and Bonnet, M., 2005, "Fast Multipole Method Applied to Elastostatic BEM-FEM Coupling," Comput. Struct., **83**, pp. 700–717.
[10] Chaillat, S., Bonnet, M., and Semblat, J.-F., 2008, "A Multi-Level Fast Multipole BEM for 3-D Elastodynamics in the Frequency Domain," Comput. Methods Appl. Mech. Eng., **197**(49–50), pp. 4233–4249.
[11] Liu, Y., 2009, *Fast Multipole Boundary Element Method: Theory and Applications in Engineering*, Cambridge University Press, Cambridge, NY.
[12] Fochesato, C., and Dias, F., 2006, "A Fast Method for Nonlinear Three-Dimensional Free-Surface Waves," Proc. R. Soc. London Ser. A, **462**, pp. 2715–2735.
[13] Phillips, J. R., and White, J. K., 1997, "A Pre-Corrected-FFT Method for Electrostatic Analysis of Complicated 3D Structures," IEEE Trans. Comput. Aided Design Integrated Circuits Syst., **16**, pp. 1059–1072.
[14] Yan, H.-M., and Liu, Y.-M., 2011, "An Efficient High-Order Boundary Element Method for Nonlinear Wave-Wave and Wave-Body Interactions," J. Comput. Phys., **230**, pp. 402–424.
[15] Yan, H.-M., 2010, "Computations of Fully Nonlinear Three-Dimension Wave-Body Interactions," Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.
[16] Dommermuth, D. G., Yue, D. K. P., Lin, W. M., and Rapp, R. J., 1988, "Deep-Water Plunging Breakers: A Comparison Between Potential Theory and Experiments," J. Fluid Mech., **189**, pp. 423–442.
[17] Grilli, S., Subramanya, R., Svendsen, I. A., and Veeramony, J., 1994, "Shoaling of Solitary Waves on Plane Beaches," J. Waterway, Port, Coastal Ocean Eng., **120**(6), pp. 609–628.
[18] Dean, R. G., and Dalrymple, R. A., 2000, *Water Wave Mechanics for Engineers and Scientists*, World Scientific, Singapore.
[19] Dold, J. W., and Peregrine, D. H., 1986, "An Efficient Boundary-Integral Method for Steep Unsteady Water Waves," *Numerical Methods for Fluid Dynamics II*, K. W. Morton and M. J. Baines, eds., Clarendon, Oxford.
[20] Dold, J. W., 1992, "An Efficient Surface-Integral Algorithm Applied to Unsteady Gravity Waves," J. Comput. Phys., **103**, pp. 90–115.
[21] Dold, J. W., and Peregrine, D. H., 1984, "Steep Unsteady Water Waves: An Efficient Computational Scheme," Proceedings of 19th International Conference on Coastal Engineering, Houston, pp. 955–967.
[22] Lin, W. M., 1984, "Nonlinear Motion of the Free Surface Near a Moving Body," Ph.D. thesis, Department of Ocean Engineering, Massachusetts Institute of Technology, Cambridge, MA.
[23] Longuet-Higgins, M. S., and Cokelet, E. D., 1976, "The Deformation of Steep Surface Waves on Water. I. A Numerical Method of Computation," Proc. R. Soc. London Ser. A, **350**, pp. 1–26.
[24] Grilli, S. T., Skourup, J., and Svendsen, I. A., 1989, "An Efficient Boundary Element Method for Nonlinear Water Waves," Eng. Anal. Boundary Elements, **6**(2), pp. 97–107.
[25] Grilli, S. T., and Subramanya, R., 1996, "Numerical Modeling of Wave Breaking Induced by Fixed or Moving Boundaries," Comput. Mech., **17**, pp. 374–391.
[26] Romate, J. E., 1989, "The Numerical Simulation of Nonlinear Gravity Waves in Three Dimensions Using a Higher Order Panel Method," Ph.D. thesis, University of Twente, The Netherlands.
[27] Broeze, J., 1993, "Numerical Modelling of Nonlinear Free Surface Waves With a 3D Panel Method," Ph.D. thesis, University of Twente, The Netherlands.
[28] van Daalen, E. F. G., 1993, "Numerical and Theoretical Studies of Water Waves and Floating Bodies," Ph.D. thesis, University of Twente, The Netherlands.
[29] Berkvens, P. J. F., 1998, "Floating Bodies Interacting With Water Waves—Development of a Time-Domain Panel Method," Ph.D. thesis, University of Twente, The Netherlands.
[30] Liu, P. L. F., Hsu, H. W., and Lean, M. H., 1992, "Applications of Boundary Integral Equation Methods for Two-Dimensional Non-Linear Water Wave Problems," Int. J. Numer. Methods Fluids, **15**, pp. 1119–1141.
[31] Cooker, M. J., 1990, "A Boundary-Integral Method for Water Wave Motion Over Irregular Beds," Eng. Anal. Boundary Elements, **7**(4), pp. 205–213.
[32] Grilli, S. T., Guyenne, P., and Dias, F., 2001, "A Fully Nonlinear Model for Three-Dimensional Overturning Waves Over Arbitrary Bottom," Int. J. Numer. Methods Fluids, **35**(7), pp. 829–867.
[33] Fochesato, C., Grilli, S. T., and Guyenne, P., 2005, "Note on Non-Orthogonality of Local Curvilinear Coordinates in a Three-Dimensional Boundary Element Method," Int. J. Numer. Methods Fluids, **48**, pp. 305–324.
[34] Fochesato, C., Grilli, S. T., and Dias, F., 2007, "Numerical Modeling of Extreme Rogue Waves Generated by Directional Energy Focusing," Wave Motion, **44**, pp. 395–416.
[35] Grilli, S. T., Dias, F., Guyenne, P., Fochesato, C., and Enet, F., 2010, "Progress in Fully Nonlinear Potential Flow Modeling of 3D Extreme Ocean Waves," *Advances in Numerical Simulation of Nonlinear Water Wave* (Advances in Coastal and Ocean Engineering), Q. Ma, ed., Vol. 11, World Scientific, Singapore, pp. 75–128.
[36] Hughes, S. A., 1993, *Physical Models and Laboratory Techniques in Coastal Engineering*, World Scientific, Singapore.
[37] Svendsen, I. A., and Grilli, S. T., 1990, "Nonlinear Waves on Steep Slopes," J. Coastal Res., **7**, pp. 185–202. Available at http://www.jstor.org/stable/25735398
[38] Lakhan, C., 1982, "Calculating Modulus k of Cnoidal Waves," J. Waterway Port Coastal Ocean Eng., **108**(3), pp. 435–438.
[39] Abramowitz, M., and Stegun, I. A., 1972, *Handbook of Mathematical Functions*, Dover, New York.
[40] Goring, D. G., 1978, "Tsunamis—The Propagation of Long Waves Onto a Shelf," W. M. Keck Laboratory of Hydraulics and Water Resources, Division of Engineering and Applied Science, California Institute of Technology, Pasadena, CA, Report No. KH-R-38.
[41] Tanaka, M., 1986, "The Stability of Solitary Waves," Phys. Fluids, **29**(3), pp. 650–655.
[42] Rankin, W. T., 1999, "Efficient Parallel Implementations of Multipole Based *N*-Body Algorithms," Ph.D. thesis, Dept. of Electrical Engineering, Duke University, Durham, NC.
[43] Chapman, B., Jost, G., and van de Pas, R. V., 2007, *Using OpenMP: Portable Shared Memory Parallel Programming*, MIT Press, Cambridge, MA.