Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 200 (2004) 749–768

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

A Cartesian method for fitting the bathymetry and tracking the dynamic position of the shoreline in a three-dimensional, hydrodynamic model

XinJian Chen *,1

Resource Conservation and Development Department, Southwest Florida Water Management District, 7601 Highway 301 North, Tampa, FL 33637, USA

Received 8 January 2004; received in revised form 2 April 2004; accepted 3 May 2004
Available online 11 June 2004

Abstract

This paper presents a Cartesian method for the simultaneous fitting of the bathymetry and shorelines in a three-dimensional, hydrodynamic model for free-surface flows. The model, named LESS3D (Lake & Estuarine Simulation System in Three Dimensions), solves flux-based finite difference equations in the Cartesian-coordinate system (x, y, z). It uses a bilinear bottom to fit the bottom topography and keeps track the dynamic position of the shoreline. The resulting computational cells are hybrid: interior cells are regular Cartesian grid cells with six rectangular faces, and boundary/bottom cells (at least one face is the water–solid interface) are unstructured cells whose faces are generally not rectangular. With the bilinear interpolation, the shape of a boundary/bottom cell can be determined at each time step. This allows the Cartesian coordinate model to accurately track the dynamic position of the shorelines. The method was tested with a laboratory experiment of a Tsunami runup case on a circular island. It was also tested for an estuary in Florida, USA. Both model applications demonstrated that the Cartesian method is quite robust. Because the present method does not require any coordinate transformation, it can be an attractive alternative to curvilinear grid model.

Keywords: Tracking shoreline; Fitting bathymetry; Fitting boundary; Hybrid grid cells; LESS3D; Bi-linear interpolation; Flux-based finite difference equations

1. Introduction

Three-dimensional, hydrodynamic models have been widely used in studying circulations and transport processes in lakes, estuaries and coastal waters. While some 3D models solve the governing equations

* Tel.: +1-813-985-7481; fax: +1-813-987-6747.
E-mail address: xinjian.chen@swfwmd.state.fl.us (X. Chen).

[1] Sr. Professional Engineer.

0021-9991/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2004.05.004

directly in the Cartesian coordinate system (e.g. [1,2]), others solve these equations in a transformed coordinate system [3,4]. The purpose of transforming the coordinate system is to get a better representation of the topography and/or boundaries, especially when the bathymetry and shorelines vary dramatically. For example, some 3D models use the so-called σ-coordinate for a better representation of the topography and curvilinear grids for a better fit of shorelines.

The σ-coordinate is obtained by dividing the vertical coordinate (z) by the total water depth. In σ-coordinate models, the water column is divided into a fixed number of vertical layers. The thickness of each layer can vary in the vertical direction but the ratio of the layer thickness to the total water depth is fixed for the same layer. Therefore, if the water depth varies, the layer thickness varies accordingly. Although a σ-coordinate model can fit the topography, it pays the price of pressure gradient errors and increased numerical diffusions for areas with steep bottom slopes [5].

In a z-coordinate model, the water column is sliced into different layers with horizontal planes. Although the vertical spacing Δz may vary in the vertical direction, each layer has the same thickness everywhere. Model variables are placed at the same level for the same layer. Depending on the water depth, the number of grid points in the vertical direction varies in a z-coordinate model. Shallow areas have fewer vertical grid points than deep areas do. In most z-coordinate models, if the bottom elevation is lower than the middle point of a layer, then the entire cell will be taken as a valid one and the bottom of the layer is the bottom in the computation. Otherwise, the cell is abandoned and the top of the layer will be used as the bottom in the computation. As a result of this treatment, the bottom cell in the computation always has the same thickness as the horizontal layer. Recent improvements of z-level models involve the use of partial cells to get a better approximation of the bottom topography [6–8]. Instead of using the thickness of the layer, the difference between the top of the layer and the real physical bottom is used as the bottom cell thickness [8], which is generally smaller than the layer thickness. Partial cells are still rectangular boxes and the resulting bottom in the computation remains a staircase shape. To eliminate the staircase bottom, a piecewise linear bottom has been used in a laterally averaged model named LAMFE that can accurately fit the topography [7,9–11]. Fig. 1 compares different bottom treatment methods used in the z-level model LAMFE, with the mesh shown with thin solid lines. The solid line is the piecewise linear bottom, while the dashed line and the

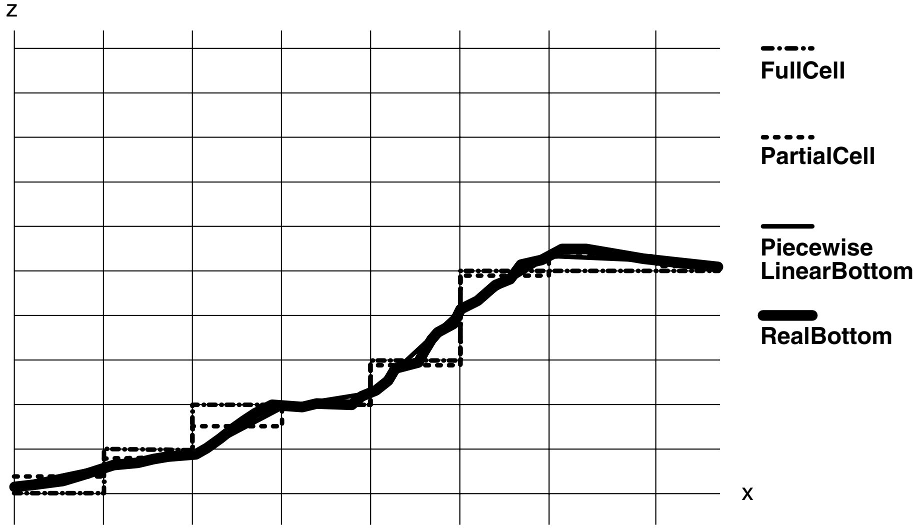

Fig. 1. Various topography treatment methods in z-level models.

dash-dotted line are the resulting computational bottom using full cells and partial cells, respectively. The real bottom (the thick solid line) is also plotted in the figure to show how each bottom treatment method fits the real bottom topography. It is clear that the piecewise linear bottom does a much better job in fitting the bottom topography than other two methods. A similar discussion on various bottom treatment methods can also be found in [12].

The piecewise linear bottom used in the laterally averaged 2D model LAMFE [7,9–11] bears a similarity to the shaved cells proposed by Adcroft et al. [12]. By defining the bottom elevation at the edge of the grid, the piecewise linear bottom used in LAMFE can be easily constructed by linking the bottom elevations of the simulation domain with straight lines. As a result, the bottom of each individual water column has one single slope (Fig. 1) regardless how many grid cells in the water column abut to the bottom. The resulting piecewise linear bottom for a vertical 2D problem is slightly different from that using shaved cells because in the shaved cell method, each grid cell abutting to the bottom is cut by a plane with its local slope. As a result, the bottom of a water column may have more than one slope.

In the 3D model, however, the piecewise linear bottom is extended to a bilinear bottom. Similar to the way the piecewise linear bottom is constructed in the LAMFE model, the bilinear bottom in the 3D model is constructed using bottom elevations defined at the four corners of a rectangular grid. The bottom elevation at any point within the grid is determined using the bilinear interpolation. If at least one corner of the grid is emerged while the remaining corners are submerged, then the shoreline runs through the grid and can be determined through the bilinear interpolation from the known water level and bottom elevations at the four corners. As a result, the shoreline can be accurately fit in the model, too.

This paper presents a Cartesian method that uses the bilinear interpolation to fit the bathymetry and track the dynamic position of the shoreline in a three-dimensional, hydrodynamic model named LESS3D. In the following, governing equations for three-dimensional shallow water flows are first given, followed by a description of the grid cells used in the 3D model LESS3D. Flux-based finite difference equations are then presented. A test of the method against laboratory data, along with a model application to a real estuary in Florida, is described before conclusions are drawn.

2. Governing equations

For shallow water flows in lakes and estuaries, the hydrostatic pressure assumption is generally valid in most cases. Situations where the hydrostatic pressure assumption may be questionable are discussed in several previous publications (e.g. [13,14]). Using the Boussinesq approximation, Reynolds averaged Navier–Stokes (RANS) equations for free surface flows in shallow lakes and estuaries have the following form:

$$\frac{\partial u_l}{\partial x_l} = 0, \tag{1}$$

$$\frac{\partial u_l}{\partial t} + \frac{\partial u_l u_m}{\partial x_m} = \phi_l - \frac{1}{\rho_0}\frac{\partial p}{\partial x_l} + \frac{\partial}{\partial x_m}\left(A\frac{\partial u_l}{\partial x_m}\right), \tag{2}$$

$$\frac{\partial c}{\partial t} + \frac{\partial u_m c}{\partial x_m} = \frac{\partial}{\partial x_m}\left(B\frac{\partial c}{\partial x_m}\right) + S_{\mathrm{s}} + R, \tag{3}$$

where t is time; p is pressure; x_l ($l = 1, 2,$ and 3) represents the Cartesian coordinates (x_1 is from west to east, x_2 is from south to north, and x_3 is vertical pointing upward); u_l is the velocity in the x_l-directions; ρ_0 is the reference density; $(\phi_1, \phi_2, \phi_3) = (fu_2, -fu_1, -g)$; f and g are, respectively, the Coriolis parameter and the gravitational acceleration; c denotes concentration and can be temperature, salinity, suspended sediment concentrations, or nutrient concentrations; S_s and R represent the sink/source terms and the reaction terms, respectively; A and B are eddy viscosity and diffusivity, respectively. In this paper, the Cartesian coordinates x_l ($l = 1$, 2, and 3) are also denoted with x, y, and z. Accordingly, u, v, and w are also used to represent velocities in three directions. Furthermore, (A, B) in the horizontal and vertical directions are distinguished with (A_h, B_h) and (A_v, B_v), respectively.

Density in above equations is a function of temperature and salinity [15]. In Eq. (3), if the material simulated involves settling, u_3 in the advective term includes the settling velocity of the material. Eq. (1) is the continuity equation and can be integrated over the water depth to get an equation for the free surface

$$\frac{\partial\eta}{\partial t} + \frac{\partial}{\partial x_1}\left(\int_{h_0}^{\eta} u_1 \,\mathrm{d}z\right) + \frac{\partial}{\partial x_2}\left(\int_{h_0}^{\eta} u_2 \,\mathrm{d}z\right) = r, \tag{4}$$

where η is the free surface elevation; h_0 is the bottom elevation; and r is the net rain intensity (precipitation minus evaporation) in cm/s. The above equation is obtained with the assumption that the flux through the bed is zero.

Boundary conditions in the horizontal directions are specified with either free surface elevations or velocities for open boundaries. At solid boundary, normal velocity is set to zero and the pressure gradient in the normal direction is set to zero. Boundary conditions at the free surface and at the bottom are implicitly specified by wind and bottom shear stresses, respectively. Details on specifying boundary conditions in the model are described in [16].

3. Hybrid grid cells

The model uses the Cartesian grid system with N_1, N_2, and N_3 cells in the west–east, south–north, and vertical directions, respectively. A colocated arrangement of model variables was used, where all variables are placed at the center of the grid cell. Fig. 2 shows the horizontal and vertical views of the computational stencil in Cartesian coordinates. In Fig. 2, Δx, Δy, and $\Delta\zeta$ are grid spacings in x-, y-, and z-directions, respectively. Low case letters i, j, and k are grid indexes in the three directions. While Δx varies only with i, Δy varies only with j. $\Delta\zeta$ is the thickness of the horizontal layer, which is constant for the same k-index. Three-dimensional model variables (e.g., u, v, w, p, c, etc.) at the center of the grid cell with indexes (i, j, k) are marked with subscripts i, j, and k. For example, at the center of the grid cell, concentration c is denoted as $c_{i,j,k}$, while velocities u, v, and w are marked as $u_{i,j,k}$, $v_{i,j,k}$ and $w_{i,j,k}$, respectively. As shown in Fig. 2, model variables at the center of the east face are denoted with the subscript group of $(i + 1/2, j, k)$, even though the Cartesian grids are not necessarily uniform. At the center of the north and top faces, the subscript groups are $(i, j + 1/2, k)$ and $(i, j, k + 1/2)$, respectively. For two-dimensional variables such as the surface elevation (η) and water depth (D), they are denoted as $\eta_{i,j}$ and $D_{i,j}$ at the center of the grid, as $\eta_{i+1/2,j}$ and $D_{i+1/2,j}$ on the east side of the grid, and as $\eta_{i,j+1/2}$ and $D_{i,j+1/2}$ on the north side of the grid.

The water depth is simply calculated from the surface elevation (η) and the bottom elevation (E) at the center of the horizontal grid ($D = \eta - E$). The model reads bottom elevation data (bathymetry) given at the four corners of the horizontal grid. Using the bilinear interpolation, bottom elevations everywhere within the grid are defined as follows (Fig. 3):

$$E(x + a, y + b) = \begin{bmatrix} 1 - b & b \end{bmatrix} \begin{bmatrix} E(x,y) & E(x+1,y) \\ E(x,y+1) & E(x+1,y+1) \end{bmatrix} \begin{bmatrix} 1 - a \\ a \end{bmatrix}. \tag{5}$$

At the center of the grid cell, $a = b = 1/2$ and the bottom elevation equals to the mean of the bottom elevations at the four corners. In Fig. 4, typical grid cells (rectangular boxes) used in the model are shown with dotted lines. The bathymetry is given at the four corners of the cell (looking downward) with the

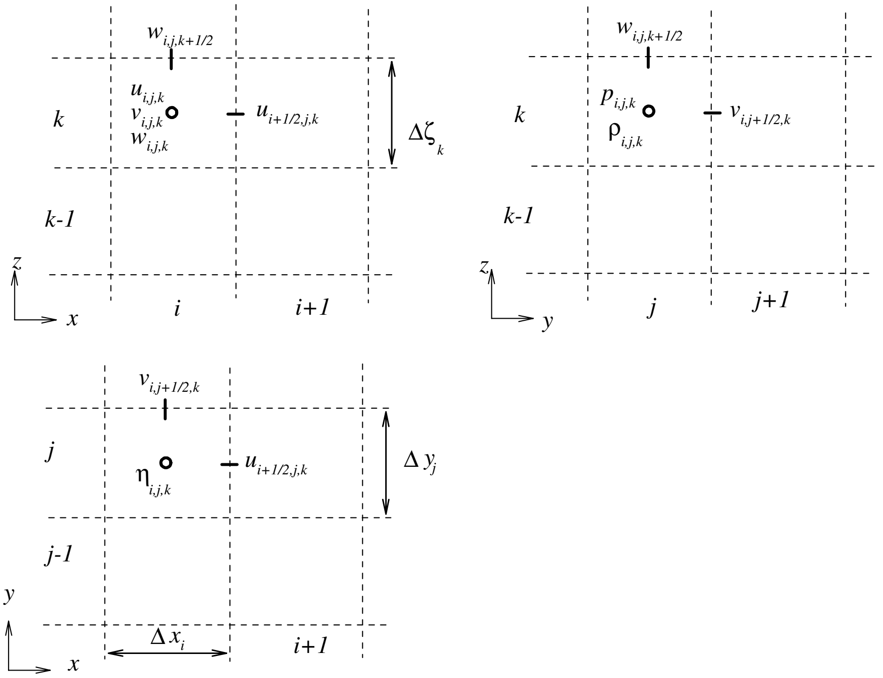

Fig. 2. A Cartesian grid system used in the LESS3D model with a colocated arrangement of model variables. The top-left graph is a view looking north, while the top-right and bottom graphs are views looking west and downward, respectively.

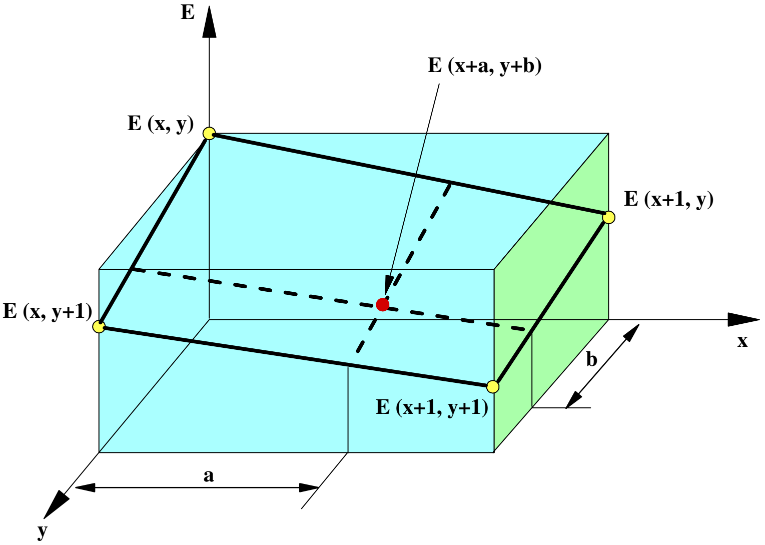

Fig. 3. With the bilinear interpolation, bottom elevation everywhere within the grid can be defined.

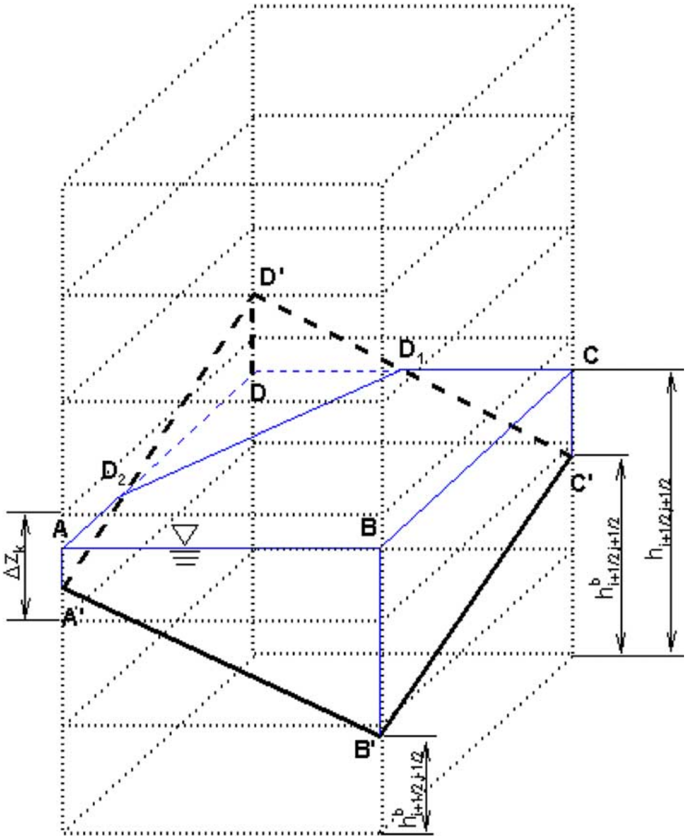

Fig. 4. A boundary cell with one emerged corner and three submerged corners.

bottom elevations (measured form a baseline elevation) at A′, B′, C′ and D′. For example, at B′ in Fig. 4, the bottom elevation is $h^b_{i+1/2,j-1/2}$. ABCD in Fig. 4 is the free surface at a certain time point. It can be seen that one corner of the cell (D′) is emerged, while other three corners are submerged.

The bottom face A′B′C′D′ in Fig. 4 is generally a curved face. It intersects with the water surface, resulting in a shoreline (D_1D_2 in Fig. 4). As shown in Fig. 5, shoreline D_1D_2 is not necessarily a straight line. If both A′ and C′ are higher (or lower) than B′ and D′, the bottom face will have a saddle point (Fig. 5(b)) and there will be two shorelines passing through the grid. In Figs. 5(a) and (b), dashed lines are bottom elevation contours, or shorelines for various water levels. When the water level varies, so do the cell volume, areas of cell faces, and the shoreline (Fig. 6).

For the convenience of the presentation, let us call Cartesian grid cells that involve the bed and/or shorelines boundary cells. For Cartesian grid cells that are on the land and do not have water in them, let us call them land cells. For Cartesian grid cells that are in the water but are not boundary cells, let us call them internal cells. As will be described in the next section, the LESS3D model solves Eqs. (1) through (3) using flux-based finite difference equations for control volumes, instead of the Cartesian grid cells plotted with dashed lines in the Fig. 4. Let us call these control volumes computational cells. For example, internal cells are all computational cells. However, a computational cell is not always necessarily the same as a Cartesian grid cell. For example, the multi-face control volume $ABCD_1D_2A'B'C'$ shown in Fig. 4 is a computational cell. Unlike a Cartesian grid cell that is always a rectangular box, a computational cell is just a multiple-face

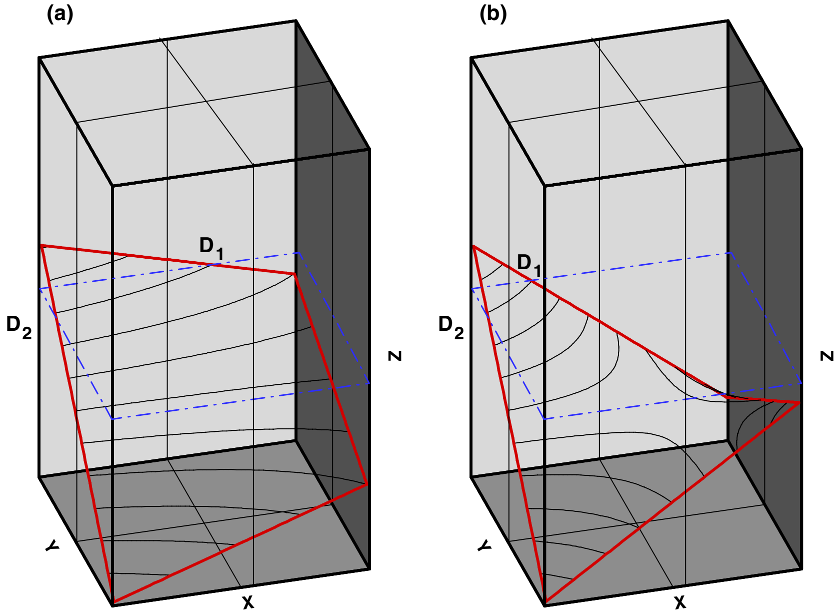

Fig. 5. The bottom face is generally curved and elevation contours (shorelines) are not necessarily straight lines. In (b), the bottom face has a saddle point.

unstructured cell for which the model solves the governing equations. As can be seen from Fig. 4, a computational cell may involve several Cartesian grid cells, of which only one cell center point is within the computational cell. In other words, a computational cell can only have one grid cell center.

As a part of the pre-process, the model calculates and saves geometric parameters of all grid cells at time $= 0$ s. These geometric parameters include the water volume, side areas, and the location of the centroid of the water volume of the Cartesian grid cell. Note that while the volume of a Cartesian grid cell is $\Delta x \Delta y \Delta \zeta$ the water volume may be smaller than $\Delta x \Delta y \Delta \zeta$ due to the fact that some portions of the grid cell may be chopped off by solid boundaries, including the bottom and the shoreline. The centroid of the water volume of a grid cell is the mass center for the portion that can actually contain water. After these geometric parameters are calculated at time $= 0$ s, the model only needs to calculate the geometric parameters for surface boundary cells at the subsequent time steps. An additional geometric parameter the model calculates at each time step is the wetted free surface areas (see $ABCD_1D_2$ in Fig. 4) of boundary cells. As will be described in the next section, the instantaneous values of these geometric parameters are used in the model in solving the equations.

The concept of using multi-face control volumes to fit the bed in a three-dimensional, *z*-coordinate hydrodynamic model is akin to that of using shaved cells to fit the bed [12], even though it was developed and implemented in the LESS3D model by the author without the prior knowledge of the existence of [12]. Comparing to the shaved cells used in [12], one of the major advantages of using bilinear bottom is that the bilinear bottom allows the bottom face to be a curved face and can thus yield a better fit of the bed variation, especially when the bottom face has a saddle point. Although the use of shaved cells can generate a piecewise bottom (instead of a staircase bottom) for a vertical 2D problem, this author believes that shaved cells proposed in [12] cannot guarantee the elimination of a staircase bottom in a 3D situation where the bed varies in both the *x*- and *y*-directions (in the two test cases shown in [12], the bed only varies in one

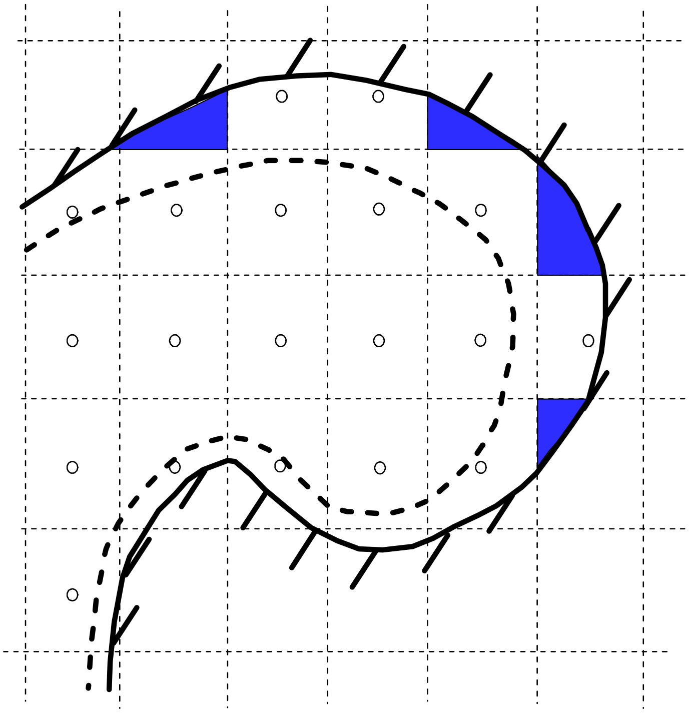

Fig. 6. The change of the shoreline. The thick solid line is the shoreline at one time point. At another time point when the water level is lower, the shoreline changes to a new position shown with the thick dashed line.

direction). The reason is that each grid cell abutting to the bottom can only be shaved by one plane with a local slope in [12]. From Fig. 4, it is clear that to avoid a staircase bottom, the plane used to shave the grid cell needs to contain all four corners of the bottom face (A′, B′, C′, and D′ in the figure). Unfortunately, it is not always possible to have all four points in one plane. As a result, two planes are generally needed to shave the bottom grid cell to avoid a staircase bottom.

In summary, computational cells used in the LESS3D model are hybrid grid cells that include both rectangular boxes for internal areas of the computational domain and multi-face control volumes near the bottom and the shorelines. As a result, the model can fit both the bed and shorelines. As the water level varies, shapes of surface computational cells near the shorelines may vary accordingly. Using the bilinear interpolation, geometric parameters of these surface computational cells can be determined and the shoreline variation can be precisely tracked.

4. Flux-based finite difference equations

Unlike other finite difference methods, the LESS3D model takes advantage of the both the finite volume method and the finite difference method by using flux-based finite difference equations to solve RANS equations for free-surface flows. By treating each computational cell as a control volume, the flux-based finite difference equations are derived from the mass and momentum balances of the computational cell. Fig. 7 gives representative shapes of an internal computational cell (Fig. 7(a)) and three kinds of boundary cells (Figs 7(b)–(d)). In the figure, grid cells are plotted with thin dashed lines. Shaded areas bounded with

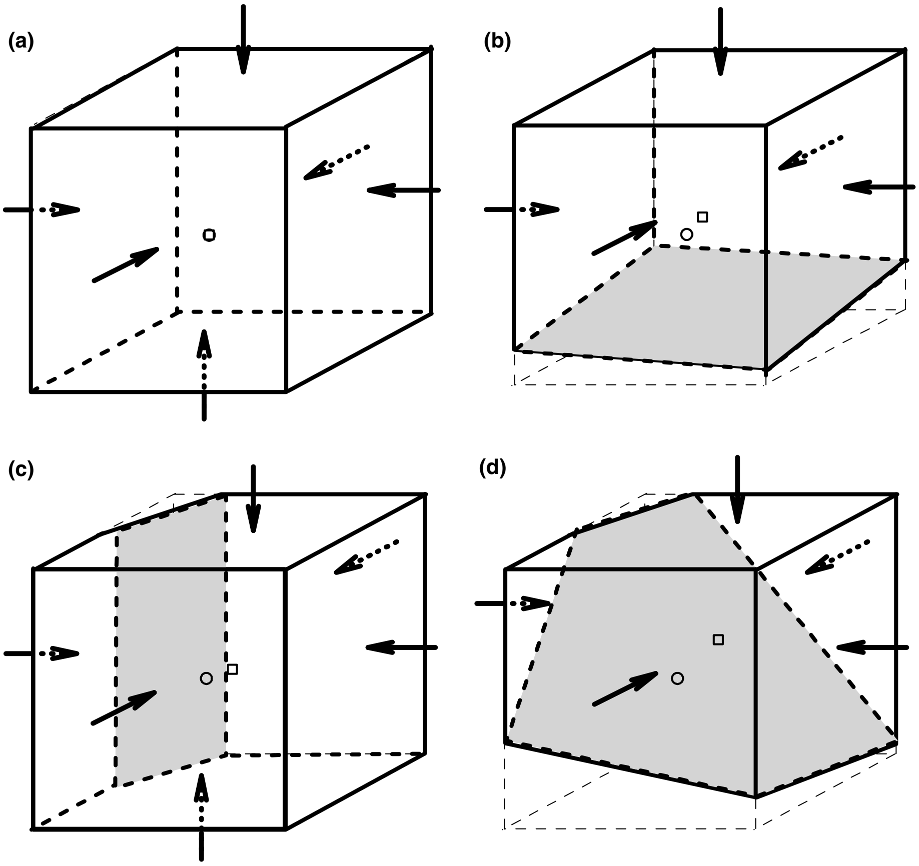

Fig. 7. Shapes of various computational cells. Grid cells are plotted with thin dashed lines. Shaded areas bounded with thick dashed lines are boundaries. Arrows represent fluxes passing through the side areas into the computational cell. Small squares are the locations of the centroids of the computational cells. Small circles are centers of the grid cells. (a) is an internal cell and (b)–(d) are boundary cells.

thick dashed lines are boundaries. Arrows represents fluxes passing through the side areas into the computational cell. Small squares are the locations of the centroids of the computational cells, while small circles are centers of the grid cells.

For each computational cell in the domain, the flux-based finite difference equation for the momentum equation can be written as

$$\frac{u_{l_{cc}}^{n+1}-u_{l_{cc}}^{n}}{\Delta t}=\varphi_i-\frac{\theta_1}{\rho_0 V_{i,j,k}^n}\iiint_V\left(\frac{\partial p}{\partial x_l}\right)^{n+1}\mathrm{d}V-\frac{1-\theta_1}{\rho_0 V_{i,j,k}^n}\iiint_V\left(\frac{\partial p}{\partial x_l}\right)^{n}\mathrm{d}V+H_l^n$$
$$+\frac{1}{V_{i,j,k}^n}\left[a_{i,j,k+1/2}^n\left(A_v\frac{\partial u_l}{\partial x_3}\right)_{i,j,k+1/2}^{n+1}-a_{i,j,k-1/2}^n\left(A_v\frac{\partial u_l}{\partial x_3}\right)_{i,j,k-1/2}^{n+1}\right], \tag{6}$$

where superscripts n and $n+1$ denote the previous and present time instants, respectively; subscript cc denotes the centriod of the computational cell; θ_1 is an implicit parameter for momentum equations; V is the volume of the computational cell; a is the area of the side face of the computational cell ($a^n_{i+1/2,j,k}$, $a^n_{i,j+1/2,k}$, and $a^n_{i,j,k+1/2}$ are areas of the east, north, and top faces, respectively); $u^{n+1}_{l_{cc}}$ and $u^n_{l_{cc}}$ are velocities at the centroid of the computational cell at the new and previous time steps, respectively; and H^n_l represents the explicit treatment of the convective terms and horizontal eddy viscosity terms

$$\begin{aligned} H^n_l = \frac{1}{V^n_{i,j,k}} \Bigg\{ & \left[a\left(A_h \frac{\partial u_l}{\partial x_1} - u_1 u_l \right) \right]^n_{i+1/2,j,k} - \left[a\left(A_h \frac{\partial u}{\partial x_1} - u_1 u_l \right) \right]^n_{i-1/2,j,k} + \left[a\left(A_h \frac{\partial u_l}{\partial x_2} - u_2 u_l \right) \right]^n_{i,j+1/2,k} \\ & - \left[a\left(A_h \frac{\partial u}{\partial x_2} - u_2 u_l \right) \right]^n_{i,j-1/2,k} - [a u_3 u_l]^n_{i,j,k+1/2} + [a u_3 u_l]^n_{i,j,k-1/2} \Bigg\}. \end{aligned} \tag{7}$$

The free-surface location is calculated using the following finite difference equation:

$$\begin{aligned} \eta^{n+1}_{i,j} = \eta^n_{i,j} & - \frac{\Delta t \theta_2}{a^\tau_{i,j}} [U^{n+1}_{1i+1/2,j} - U^{n+1}_{1i-1/2,j} + U^{n+1}_{2i,j+1/2} - U^{n+1}_{2i,j-1/2}] \\ & - \frac{\Delta t(1-\theta_2)}{a^\tau_{i,j}} [U^n_{1i+1/2,j} - U^n_{1i-1/2,j} + U^n_{2i,j+1/2} - U^n_{2i,j-1/2}] + \Delta t r^{n+1/2}, \end{aligned} \tag{8}$$

where θ_2 represents the implicitness for the continuity equation; and U_1 and U_2 are vertically integrated fluxes

$$\begin{aligned} U^n_{1i+1/2,j} &= \sum_{k=k_{un}}^{k_{um}} u^n_{1i+1/2,j,k} a^n_{i+1/2,j,k}, \quad U^n_{2i,j+1/2} = \sum_{k=k_{vn}}^{k_{vm}} u^n_{2i,j+1/2,k} a^n_{i,j+1/2,k}, \\ U^{n+1}_{1i+1/2,j} &= \sum_{k=k_{un}}^{k_{um}} u^{n+1}_{1i+1/2,j,k} a^n_{i+1/2,j,k}, \quad U^{n+1}_{2i,j+1/2} = \sum_{k=k_{vn}}^{k_{vm}} u^{n+1}_{2i,j+1/2,k} a^n_{i,j+1/2,k}, \end{aligned} \tag{9}$$

where k_{un} and k_{um} are, respectively, the bottom and top k-indexes on the east side of the grid; and k_{vn} and k_{vm} are the bottom and top k-indexes on the north side of the grid. As mentioned before, subscript group $(i+1/2, j, k)$ indicates value at the center of the east face of the cell, while subscript group $(i, j+1/2, k)$ represents value at the center of the north face of the cell. A third order interpolation is used in the model to obtain horizontal velocities (u_1 and u_2) at the centers of the east and north faces from those at the centers of grid cells.

The flux-based finite difference equation for the transport equation takes the following form:

$$\begin{aligned} \frac{V^{n+1}_{i,j,k} c^{n+1}_{i,j,k} - V^n_{i,j,k} c^n_{i,j,k}}{\Delta t} = {} & \mathbf{F}^n_{i-1/2,j,k} - \mathbf{F}^n_{i+1/2,j,k} + \mathbf{F}^n_{i,j-1/2,k} - \mathbf{F}^n_{i,j+1/2,k} + \mathbf{F}^n_{i,j,k-1/2} - \mathbf{F}^n_{i,j,k+1/2} \\ & + a^n_{i+1/2,j,k} \left(B_h \frac{\partial c}{\partial x_1} \right)^n_{i+1/2,j,k} - a^n_{i-1/2,j,k} \left(B_h \frac{\partial c}{\partial x_1} \right)^n_{i-1/2,j,k} \\ & + a^n_{i,j+1/2,k} \left(B_h \frac{\partial c}{\partial x_2} \right)^n_{i,j+1/2,k} - a^n_{i,j-1/2,k} \left(B_h \frac{\partial c}{\partial x_2} \right)^n_{i,j-1/2,k} \\ & + a^n_{i,j,k+1/2} \left(A_v \frac{\partial c}{\partial x_3} \right)^{n+1}_{i,j,k+1/2} - a^n_{i,j,k-1/2} \left(A_v \frac{\partial c}{\partial x_3} \right)^{n+1}_{i,j,k-1/2} + S_s + R, \end{aligned} \tag{10}$$

where $\mathbf{F}_{i+1/2,j,k}$, $\mathbf{F}_{i,j+1/2,k}$ and $\mathbf{F}_{i,j,k+1/2}$ are advective fluxes of the material flowing out of the cell through the east, north, and top faces.

The left-hand side of Eq. (6) represents the local acceleration of the water particle at the centroid of the computational cell in the x_l-direction. As a first order approximation, it can related to those at the center of the grid cell with the indexes i, j, and k as follows:

$$\left.\frac{\partial u_l}{\partial t}\right|_{\mathrm{cc}} = \left.\frac{\partial u_l}{\partial t}\right|_{i,j,k} + \nabla\left(\frac{\partial u_l}{\partial t}\right)\bigg|_{i,j,k} \cdot \delta\vec{s}, \tag{11}$$

where ∇ denotes the divergence operator; and $\delta\vec{s}$ is a vector pointing from the center of the grid cell (i, j, k) to the centroid of the computational cell: $\delta\vec{s} = \delta x\vec{i} + \delta y\vec{j} + \delta z\vec{k}$, where δx, δy, and δz are the projections of the distance between the center of the grid cell and the centroid of the computational cell in the x_1-, x_2-, and x_3-directions, respectively.

The finite difference form of Eq. (11) is

$$\frac{u^{n+1}_{l_{\mathrm{cc}}} - u^{n}_{l_{\mathrm{cc}}}}{\Delta t} = \frac{u^{n+1}_{l_{i,j,k}} - u^{n}_{l_{i,j,k}}}{\Delta t} + \nabla\left(\frac{u^{n}_{l_{i,j,k}} - u^{n-1}_{l_{i,j,k}}}{\Delta t}\right)\Bigg|_{i,j,k} \cdot \delta\vec{s}. \tag{12}$$

For internal cells, because the center of the grid cell is the centroid, $|\delta\vec{s}| = 0$ and the above two equations are reduced to

$$\frac{u^{n+1}_{l_{\mathrm{cc}}} - u^{n}_{l_{\mathrm{cc}}}}{\Delta t} = \frac{u^{n+1}_{l_{i,j,k}} - u^{n}_{l_{i,j,k}}}{\Delta t}. \tag{13}$$

For bottom computational cells that do not involve shorelines, if the vertical length scale of the computational cell is much smaller than the horizontal length scale (this is true for most shallow water flows), δx and δy are insignificant in comparison with the horizontal grid size [11]. Therefore, changes of the local accelerations of the water particle in x_1- and x_2-directions due to δx and δy are negligible in comparison with that due to δz, or

$$\left.\frac{\partial u_l}{\partial t}\right|_{\mathrm{cc}} \cong \left.\frac{\partial u_l}{\partial t}\right|_{i,j,k} + \frac{\partial}{\partial t}\left(\frac{\partial u_l}{\partial x_3}\right)\bigg|_{i,j,k} \delta z. \tag{14}$$

Furthermore, if a log-layer distribution of velocity is assumed for fully developed turbulence, we have

$$\frac{\partial u_l}{\partial x_3} = \frac{u^*_l}{\kappa(x_3 - h_0)} = \frac{u_l}{(x_3 - h_0)\ln[(x_3 - h_0)/z_0]}, \quad l = 1 \text{ and } 2, \tag{15}$$

where κ is the von Karman constant (0.41); u^*_l is the frictional velocity; $z_0 = k_s/30$, and k_s is the bottom roughness. Therefore,

$$\left.\frac{\partial u_l}{\partial t}\right|_{\mathrm{cc}} = \left.\frac{\partial u_l}{\partial t}\right|_{i,j,k} + \frac{\delta z}{(x_3 - h_0)\ln[(x_3 - h_0)/z_0]}\left.\frac{\partial u_l}{\partial t}\right|_{i,j,k}, \quad l = 1 \text{ and } 2. \tag{16}$$

Or,

$$\frac{u^{n+1}_{l_{\mathrm{cc}}} - u^{n}_{l_{\mathrm{cc}}}}{\Delta t} = \frac{u^{n+1}_{l_{i,j,k}} - u^{n}_{l_{i,j,k}}}{\Delta t} + \frac{\delta z}{(x_3 - h_0)\ln[(x_3 - h_0)/z_0]}\frac{u^{n}_{l_{i,j,k}} - u^{n-1}_{l_{i,j,k}}}{\Delta t}, \quad l = 1 \text{ and } 2. \tag{17}$$

Substituting the local acceleration on the left-hand side of Eq. (6) with Eq. (12) or its reduced forms (Eqs. (13) and (17)) where appropriate, Eqs. (6) and (8) can be efficiently solved using a predictor–corrector procedure named the free-surface correction (FSC) method. Details about the FSC method are explained in [16].

The LESS3D model can be run either with or without the hydrostatic pressure assumption. When the model is run with the hydrostatic pressure assumption, only the two horizontal velocity components are solved from Eq. (6). In this case, the following flux-based finite difference equation for the mass balance is used to calculate the vertical velocity component after u_1^{n+1} and u_2^{n+1} are found

$$w_{i,j,k+1/2}^{n+1}a_{i,j,k+1/2} = w_{i,j,k-1/2}^{n+1}a_{i,j,k-1/2} + F_{i-1/2,j,k}^{n+1} - F_{i+1/2,j,k}^{n+1} + F_{i,j-1/2,k}^{n+1} - F_{i,j+1/2,k}^{n+1}, \tag{18}$$

where F represents flux of water flowing out of the computational cell through the side faces.

As mentioned above, the LESS3D model is able to precisely track the variation of the shoreline by updating and bookkeeping actual water volumes and side and top areas of each top cells at each time step. If the water volume is zero for a water column with the horizontal grid indexes of (i, j), the grid (i, j) is a dry grid. Otherwise, it is a wet grid. Nevertheless, when the computation marches from one time step to the next time step, a wet grid is not necessarily a computational grid. The criterion for the model to determine if a wet grid is a computational grid or not at each time step is that the water level at the center of the grid should be higher or equal to the middle point of k_nth layer, where k_n is the k-index of the bottom layer for the horizontal grid (i, j). Fig. 6 shows examples of wet horizontal grids that are not included in the computation. The shaded grids shown in the figure are those grids. For such a wet horizontal grid that is not included in the computation, its surface elevation and concentrations are estimated using the averaged values of its neighboring computational grids that are directly connected to it [16]:

$$\bar{\eta} = \frac{1}{N_n}\sum_{l=1}^{N_n}\eta_l^{n+1}, \quad \bar{c} = \frac{1}{N_n}\sum_{l=1}^{N_n}c_l^{n+1}, \tag{19}$$

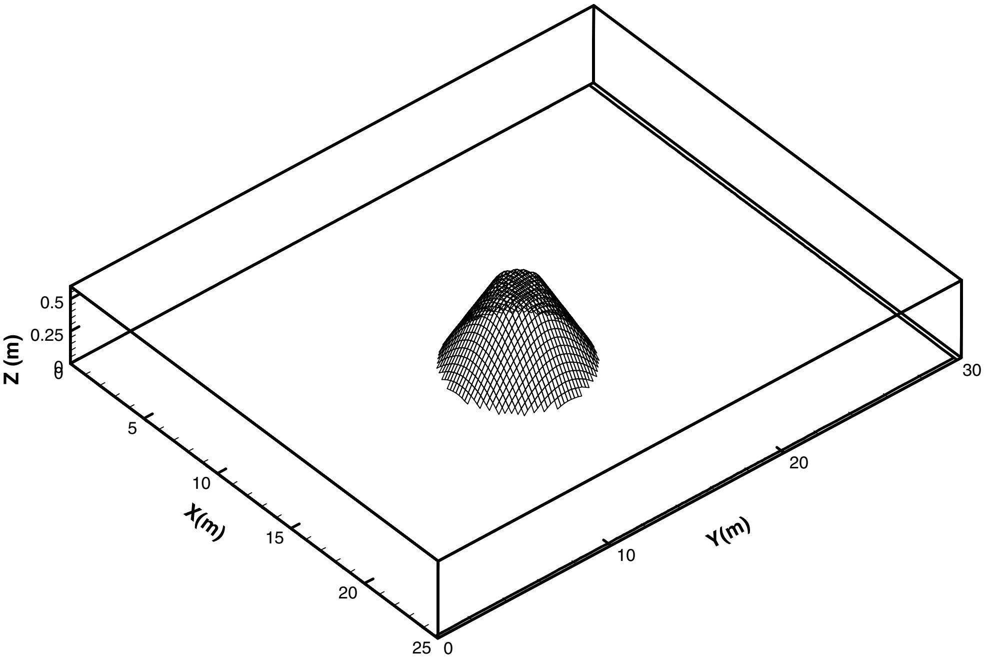

Fig. 8. A laboratory experiment of runup of a solitary wave on a circular island was conducted in a 30 m × 25 m wave basin. The wave maker is located at $x = 0$ m.

where $\bar{\eta}$ and $\bar{c}$ are averages of surface elevations and concentrations of the neighboring computational grids, which are labeled with the subscript l; and N_n is the total number of the neighboring computational grids. To ensure mass conservations, final results of all grids involved need to be corrected:

$$\begin{aligned} &\eta_l^{n+1} \Leftarrow \eta_l^{n+1} - \delta\eta^*, \quad c_l^{n+1} \Leftarrow c_l^{n+1} - \delta c^*, \\ &\eta_0^{n+1} \Leftarrow \bar{\eta} - \delta\eta^*, \quad c_0^{n+1} \Leftarrow \bar{c} - \delta c^*, \end{aligned} \tag{20}$$

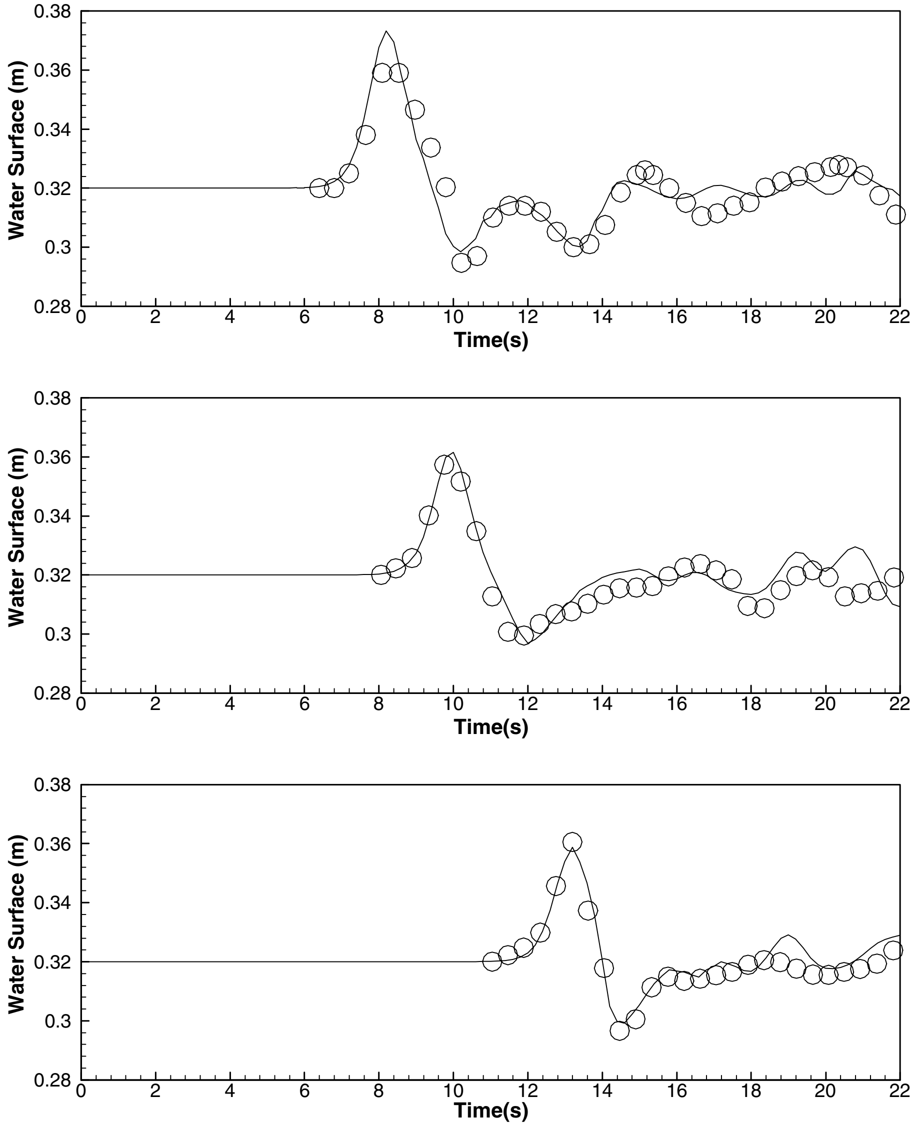

Fig. 9. Comparison of simulated surface elevation (solid lines) and measured surface elevations (circles) immediately in front of (top graph), to the side of (middle graph), and behind (bottom graph) the island.

where the symbol $\Leftarrow$ means replacing the left-hand side with the right-hand side; the subscript 0 denotes the wet grid that is not computed and needs to be estimated; and $\delta\eta^*$ and δc^* are corrections and take the following forms:

$$\delta\eta^* = \frac{(\bar{\eta} - \eta^n_{i,j})a^n_0}{a_t}, \quad \delta c^* = \frac{[\mathcal{V}^n_0 + (\eta^{n+1}_0 - \eta^n_0)a^n_0]\bar{c} - \mathcal{V}^n_0 c^n_0 - \delta\eta \sum_{l=1}^{N_n} c^{n+1}_l a^n_l}{\mathcal{V}_t}, \tag{21}$$

where a^n_l represents the surface water area (the actual wet surface area) of grid l; $\mathcal{V}^n_0$ is the water volume of the wet grid whose surface elevation and concentrations need to be estimated; and a_t and $\mathcal{V}_t$ are sums of water surface areas (actual wet surface areas) and water volumes of all grids involved in the estimation, including the neighboring computational grids and the wet grid to be estimated.

5. Tests of the method

The Cartesian method described above for fitting the bed and tracking the shoreline variation was tested with laboratory data reported in [17,18] for a Tsunami runup case on a circular island. Liu et al. [18] and Bradford and Sanders [19] used the same experiment data to test their horizontal 2D models. The 2D model

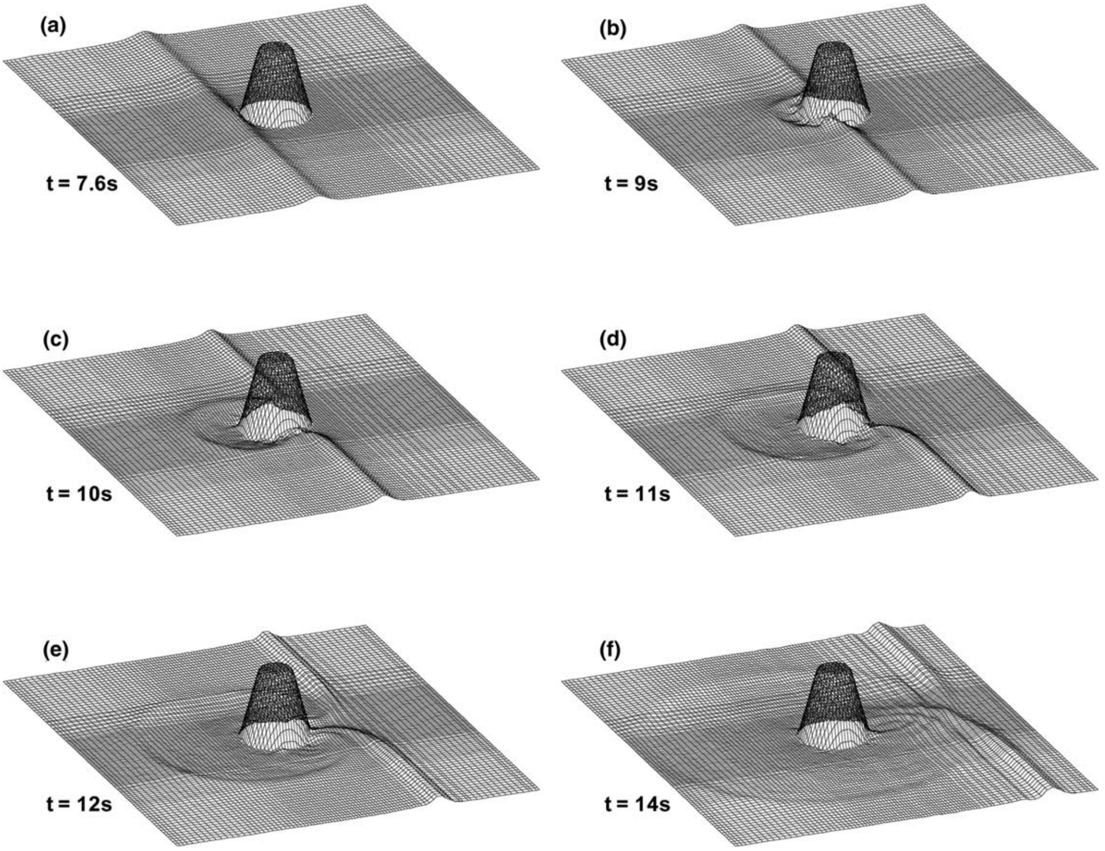

Fig. 10. Simulated snapshots of the wave runup on the circular island at $t = 7.6$, 9, 10, 11, 12, and 14 s.

of Liu et al. [18] is a finite difference model using an explicit finite difference leap-frog scheme, while the 2D model of Bradford and Sanders [17] is a finite volume model.

The laboratory experiment was conducted in a wave basin at the US Army Engineer Waterways Experiment Station, Coastal Engineering Research Center. The wave basin is 0.32 m deep, 30 m wide, and 25 m long (Fig. 8). The wavemaker is located at $x = 0$ m in Fig. 8 and generates the following solitary wave [19]:

$$\eta = H\,\mathrm{sech}^2\left(\frac{c_s(t-T)}{l_s}\right), \tag{22}$$

where $c_s = c_g[1 + H/(2d)]$; $l_s = d\sqrt{4dC_s/(3Hc_g)}$; $c_g = \sqrt{gd}$; d is the depth of the basin (0.32 m); and T is the time at which the wave crest enters the basin. In Eq. (22), $H = 0.032$ m and $T = 2.45$ s.

For this three-dimensional wave runup problem, this study used the LESS3D model with the above-mentioned bed-fitting and shoreline-tracking approach. The domain was discretized using $93 \times 103 \times 6 = 57{,}474$ grid cells. While each vertical layer has the same spacing ($\Delta\zeta$) of 0.1 m, the horizontal grid sizes (Δx and Δy) vary within a range between 0.2 and 0.4 m. Higher resolution was used near the island and near the upstream and downstream boundaries. In the model run, Eq. (22) was used as the upstream boundary condition at $x = 0$ m. A radiation boundary condition similar to that used in [18] was used for the downstream boundary at $x = 25$ m. The model was run without using the hydrostatic pressure assumption. The time step used was 0.1 s. Fig. 9 gives comparisons of simulated and measured surface elevations at three locations near the island. In the figure, model results are plotted with solid lines, while measured surface elevations are plotted with circles. The top graph of Fig. 9 compares model results with measured data immediately in front of the island, while the middle and bottom graphs compare modeled surface elevations with those measured to the side of and behind the island. Bradford and Sanders [19] also

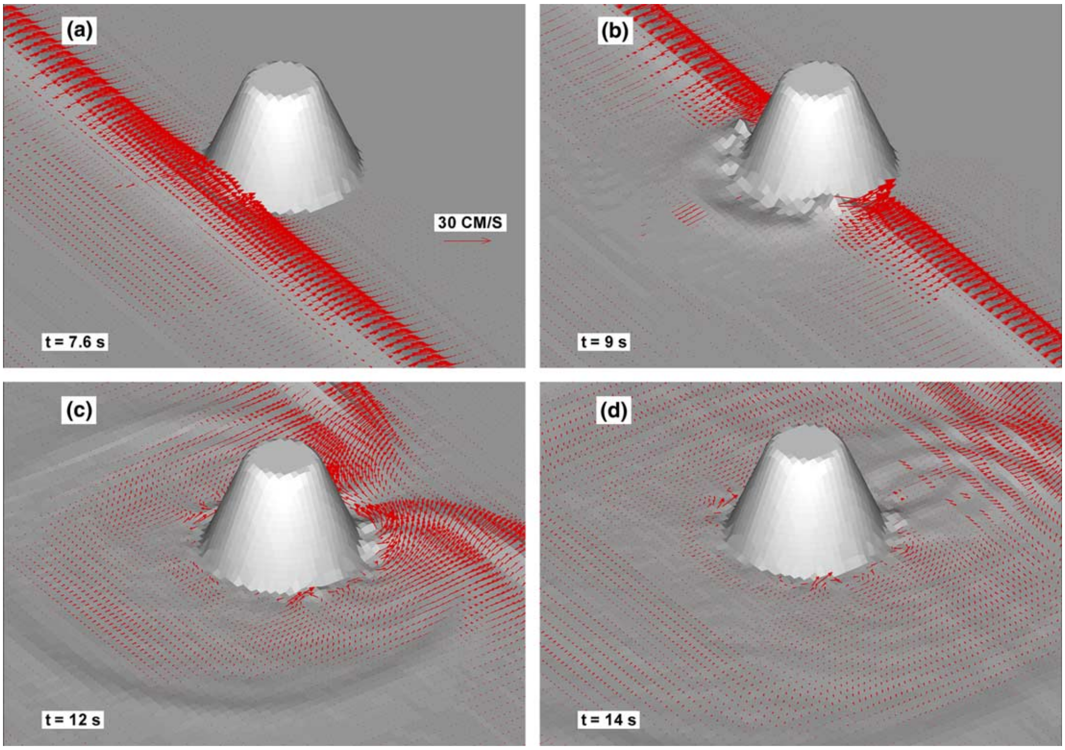

Fig. 11. Three-dimensional views of simulated velocity vectors at the surface layer around the circular island at $t = 7.6$, 9, 12, and 14 s.

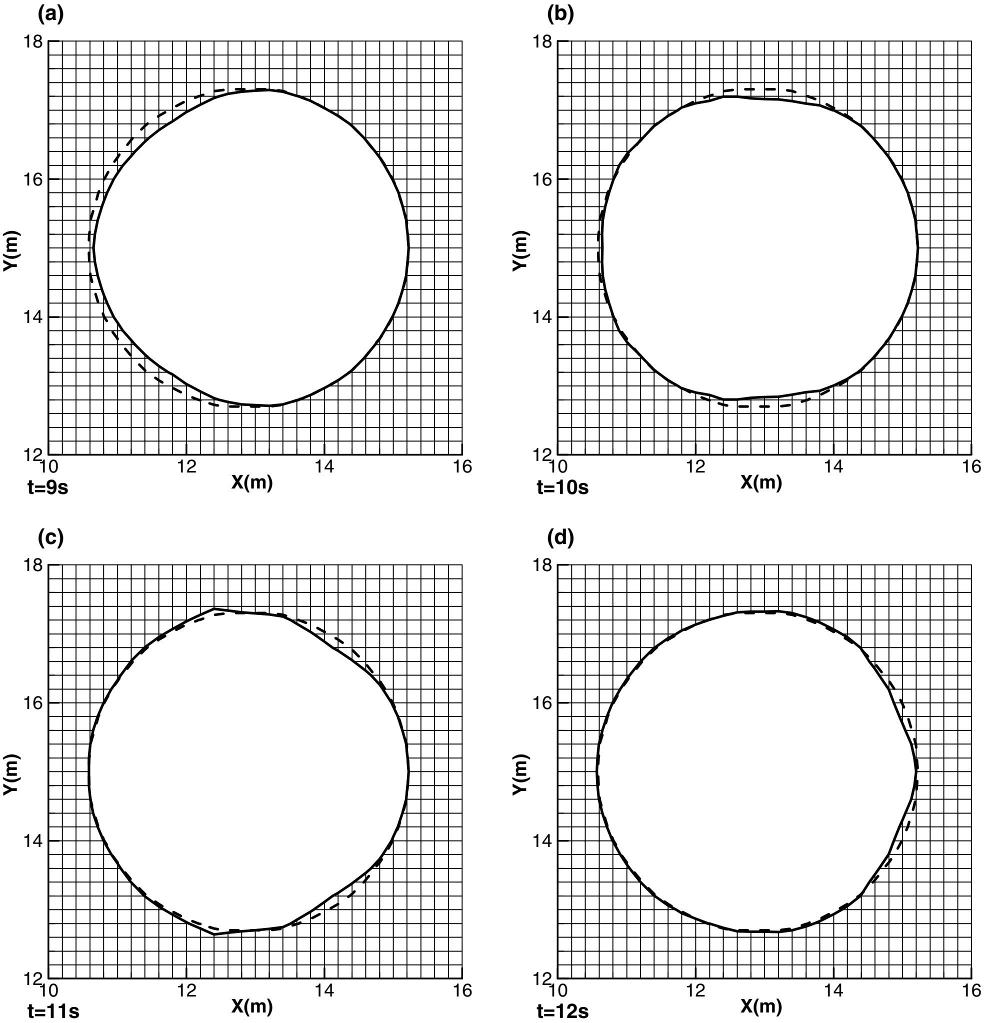

Fig. 12. Dynamic positions of the shoreline (solid lines) at $t = 9$, 10, 11, and 12 s as the solitary wave passes on the circular island. The original shoreline at $t = 0$ s is a circle (dashed lines).

compared their model results with measured data at the same three locations. As can be seen from Fig. 9, predicted surface elevations agree reasonably well with laboratory data. This study also ran the LESS3D model with the hydrostatic pressure assumption to see the significance of the non-hydrostatic effect on the flow. It turned out that the inclusion of the non-hydrostatic effect indeed improved the model prediction, but the improvement is not very significant in this solitary wave case.

Fig. 10 shows six snapshots of free-surface profiles at time = 7.6, 9, 10, 11, 12, and 14 s. In [18], Liu et al. presented simulated free-surface profiles at time = 7, 9, 11, 13, and 15 s. Their results are very similar to those shown in Fig. 10. In Fig. 10(a), the solitary wave approaches the island and the time is 7.6 s. At time = 9 s (Fig. 10(b)), the wave reaches the island and a wave runup occurs, causing a wave diffraction and resulting in a circular wave spreading out from the front of the island. Because of the shape of the island, the flow field becomes three-dimensional. A trapped wave is generated on each side of the island. As the solitary wave propagates around the island, the trapped waves propagate along the shoreline of the island (Figs. 10(b)–(e)). At around time = 12 s (Fig. 10(e)), the solitary wave passes the island and the two trapped waves collide in the lee-side of the island. The concentrated wave energy in the lee side of the island causes wave diffraction and results in an elliptic wave propagating from the lee-side of the island outward (Fig. 10(f)).

Fig. 11 presents four snapshots of simulated velocity distributions at the surface layer of the water column at time = 7.6, 9, 12, and 14 s. The velocity vectors in Fig. 11 are three-dimensional and include not only u- and v-components, but also the w-component. It can be seen from the figure that as the wave passes around the island, the water velocity near the shoreline becomes much larger than those away from the shoreline. Although the vector plots shown in the figure are similar to those reported in [18,19], a direct and detailed comparison between simulated velocity distributions in this study and those in either Liu et al. [18] or Bradford and Sanders [19] is impossible due to several obvious reasons. One of these reasons is that velocity vectors presented in [18] (or contours of the u-velocity for one case and contours of velocity magnitude for another case presented in [19]) are 2D model results and are thus depth-averaged, while those shown in Fig. 11 are 3D model results at the surface layer.

In Fig. 12, shorelines (viewing from the top) at time = 9, 10, 11, and 12 s are plotted with thick solid lines. To see the deviations of these shorelines from the original one at time = 0 s (a circle), the original shoreline is plotted with dashed lines. In the figure, the grid mesh is also plotted. As the solitary wave propagates around the island, the dynamic position of the shoreline is tracked in the model.

The Cartesian method for fitting the bed and tracking the shoreline variation was also tested for a real estuary. The lower portion of the Alafia River in Florida (Fig. 13) was used for the test. The river is narrow and meandering except for its most downstream 4 km where it is wider and has a few islands. There are five USGS (United States Geological Survey) continuous recording stations along the Alafia River. Surface elevation and salinity data were collected at 15-min intervals at four downstream stations shown in Fig. 13. At the most upstream station (Alafia River at Lithia) located about 24 km upstream from the mouth, only

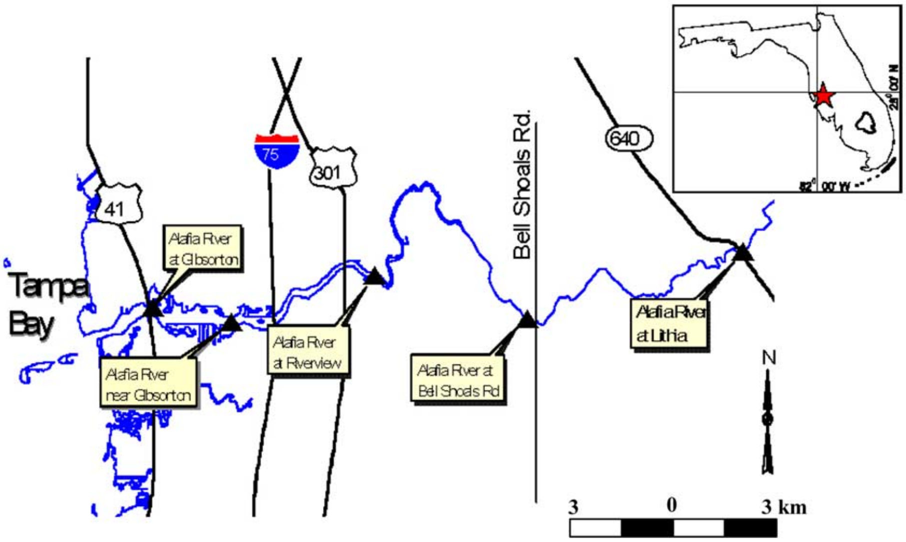
28° 00' N
82° 00' W
640
75
301
41
Bell Shoals Rd.
Alafia River at Gibsonton
Tampa Bay
Alafia River at Riverview
Alafia River near Gibsonton
Alafia River at Bell Shoals Rd
Alafia River at Lithia
N
3
0
3 km

Fig. 13. The Alafia River is a major tributary to Tampa Bay in Florida, USA.

flow was measured, because no tidal signal can be detected there. Normally, tide can be observed at about 18 km upstream from the mouth, although saline water is usually limited only to the downstream 12 km.

Due to the physical configuration of the river, hydrodynamics in the river was first simulated with a laterally averaged 2D model [11] for the entire reach of 24 km from the mouth (Alafia River at Gibsonton) to Alafia River at Lithia. The 3D model LESS3D was used for the most downstream 4.5 km of the river, from the mouth to about 850 m downstream from Highway 301 (Fig. 13) to test the Cartesian method for fitting the bed variation and tracking the dynamic position of the shoreline. Measured surface elevation, velocity, and salinity at four locations shown in Fig. 13 were used for the boundary conditions and for calibrations/verifications of the 2D and 3D models. While the downstream boundary conditions for 3D runs are the same as those for the 2D runs, the upstream boundary conditions used in 3D runs are simulated 2D model results.

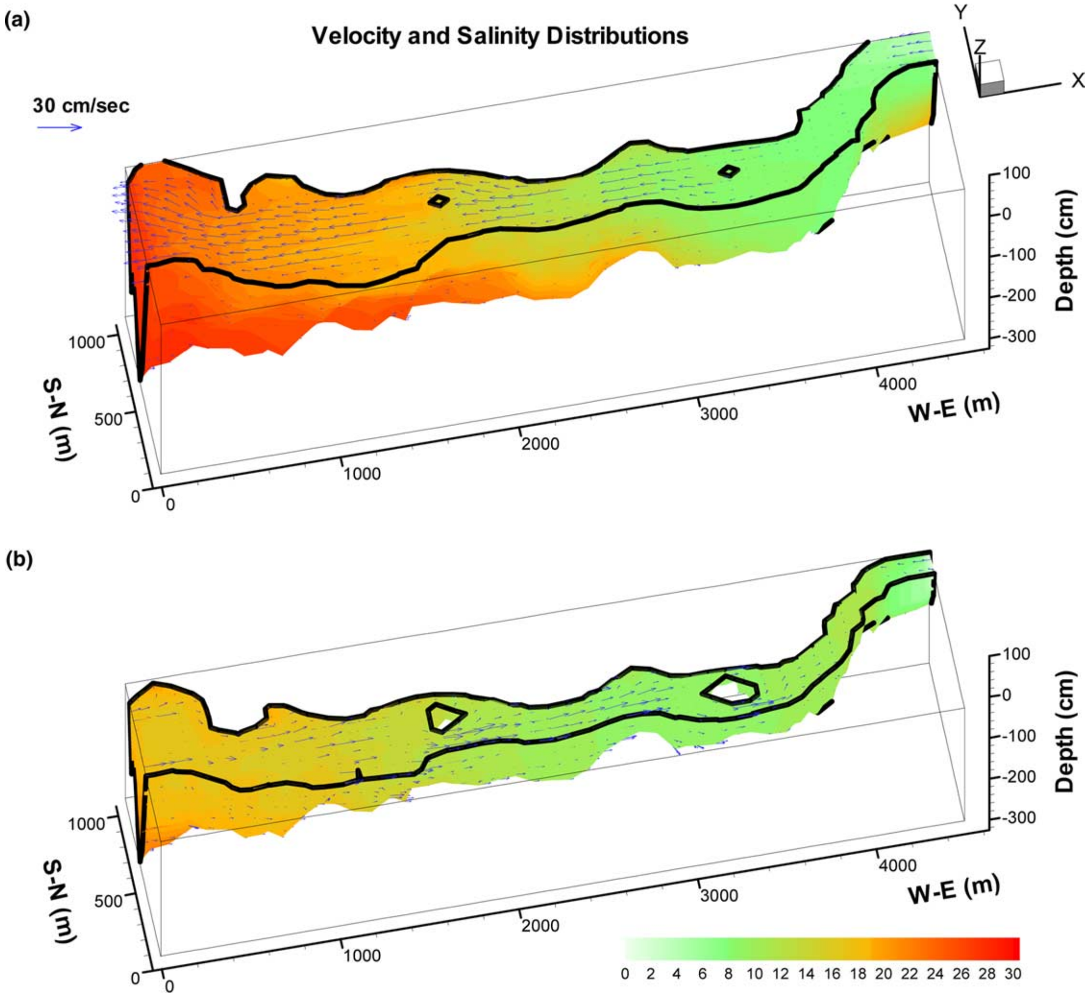

Fig. 14. Simulated velocity and salinity distributions in the Lower Alafia River at a high tide (a) and a low tide (b). Shorelines at the low tide are quite different from those at the high tide. The presence of two islands is more evident at the low tide.

The computational domain of the Lower Alafia River was disrectized with a horizontal mesh of $\Delta x = 100$ m and $\Delta y = 60$ m and a vertical spacing $\Delta\zeta$ varying between 0.3 and 0.5 m. Numbers of grids in the x-, y-, and z-directions are 45, 18, and 12, respectively, resulting in a total of 9720 grid cells. The length of the simulation was a 320-day period from May 10, 1999 to March 24, 2000. The time step used for the 3D model runs was $\Delta t = 150$ s. Predicted salinity and surface elevation at the USGS Alafia River near Gibsonton station were compared with measured data. The LESS3D model was able to yield reasonably good model results that agree well with measured field data. Details on the comparisons of model results and measured 15-min data are reported in [16] and are omitted. To demonstrate the robustness of the method presented here, 3D plots of simulated velocity and salinity distributions at a high tide and a low tide are shown in Fig. 14. Different colors (gray scales in the printed issue) in the figure represent different salinity ranges. Velocity distributions are plotted with vectors. Closed areas inside the river are three small islands that are partially submerged when the water level is high and emerged when the water level is low. A comparison of shorelines at the high tide with those at the low tide indicates that the wetting/drying process is well handled by the model. Because of a low water level, emerged areas of the islands in Fig. 14(b) are larger than those in Fig. 14(a). Near the south bank at about 1 km from the downstream boundary, a shallow area is submerged in Fig. 14(a), but exposed to the air in Fig. 14(b). Fig. 14 indicates that flow fields and salinity distributions in the lower portion of the Alafia River pattern are three-dimensional due to the existence of islands and some shallow areas.

6. Conclusions

A Cartesian method for fitting the bed variation and tracking the dynamic position of the shoreline has been developed and implemented in a three-dimensional, Cartesian grid model for free-surface flows named LESS3D. Although the model uses the so-called z-level discretization in the vertical direction, it can fit the topography nicely due to the use of the bilinear bottom. The resulting grid cells are hybrid grid cells, including regular rectangular boxes and boundary cells whose side faces are generally not rectangular. The model can fit both the bed and the dynamic shoreline by bookkeeping volumes and side face areas (including wetted surface areas) of grid cells. Therefore, it has the potential to be an attractive alternative to curvilinear grid models.

The model presented in the paper takes advantage of the finite difference method and the finite volume method by treating each computational cell as a control volume. Flux-based finite difference equations are derived from the mass and momentum balances for each control volume. For internal cells, these flux-based finite difference equations are the same as the finite difference equations derived directly from the RANS equations. Because the LESS3D model solves the flux-based finite difference equations, it conserves both mass and momentum.

The Cartesian method was verified with a laboratory experiment before it was applied to a real estuary in Florida, USA. Model results agree well with both the laboratory experiment of a wave runup on a circular island and the tidal flow in the Lower Alafia River in Florida where the shoreline variation is more significant. By accurately following the shoreline variations, the wetting/drying phenomenon can be handled automatically. Although many previous models (e.g. [3,18,19]) monitor the wetting and drying of the computational domain, these models cannot accurately follow the shoreline change because they do not include the sub-grid variation of the shoreline. In these models, a grid is either entirely included or entirely excluded in the computation, depending on the criterion used for the decision. For example, in [19], the averaged water depth of the grid that is calculated by dividing the actual water volume by the entire grid area (instead of the wet surface area) is compared with a user-defined small number (ε) to decide whether the grid is included in the computation or not. Therefore, although these models can handle the wetting and drying, the resulting shorelines are normally composed of zigzag lines that give a less accurate representation of the actual coastline than LESS3D.

It is worth pointing out that the Cartesian method presented in this paper allows the water depth to go to zero near the shoreline. This property is very useful in shallow water applications where water quality and ecological processes are significant near shorelines where water depth is shallow. In such cases, the present model offers an advantage over σ-level models that require a minimum water depth to avoid numerical stability issues due to very thin σ-layers.

References

[1] P.B. Crean, T.S. Murty, J.A. Stronach, in: Mathematical Modeling of Tides and Estuarine Circulation, Lecture Notes on Coastal and Estuarine Studies, vol. 30, Springer, New York, 1988, 471pp.

[2] V. Casulli, R.T. Cheng, Semi-implicit finite difference methods for three-dimensional shallow water flow, Int. J. Numer. Methods Fluids 15 (1992) 629–648.

[3] Y.P. Sheng, A three-dimensional mathematical model of coastal, estuarine and lake currents using boundary fitted grid, Report No. 585, A.R.A.P. Group of Titan Systems, New Jersey, Princeton, NJ, 1986.

[4] A.F. Blumberg, G.L. Mellor, A description of a three-dimensional coastal ocean circulation model, in: N.S. Heaps (Ed.), Three-Dimensional Coastal Ocean Models, AGU Coastal and Estuarine Series, vol. 4, American Geophysical Union, Washington, DC, 1987.

[5] G.L. Mellor, L.Y. Oey, T. Ezer, Sigma coordinate pressure gradient errors and the seamount problem, J. Atmos. Oceanic Technol. 15 (1998) 1122–1131.

[6] R.C. Pacanowski, A. Gnanadesikan, Transient response in a Z-level ocean model that resolves topography with partial cells, Monthly Weather Rev. 126 (1998) 3248–3270.

[7] X. Chen, Responses of a hybrid z-level model to various topography treatment methods for a boundary value problem and an initial value problem, in: M.L. Spaulding (Ed.), Estuarine and Coastal Modeling, Proceedings of the 7th International Conference, ASCE, 2001, pp. 614–627.

[8] M.D.J.P. Bijvelds, J. van Kester, G.S. Stelling, A comparison to two 3D shallow water models using sigma-coordinates and z-coordinates in the vertical direction, in: M.L. Spaulding, H. Lee Butler (Eds.), Estuarine and Coastal Modeling, Proceedings of the 6th International Conference, ASCE, 1999, pp. 130–147.

[9] X. Chen, M.S. Flannery, Use of a hydrodynamic model for establishing a minimum freshwater flow to the lower hillsborough river, in: M.L. Spaulding, A.F. Blumberg (Eds.), Estuarine and Coastal Modeling, Proceedings of the 5th International Conference, ASCE, 1997, pp. 663–678.

[10] X. Chen, M.S. Flannery, D.L. Moore, Response times of salinity in relation to changes in freshwater inflows in the lower Hillsborough river, Florida, Estuaries 23 (2000) 735–742.

[11] X. Chen, Using a piecewise linear bottom to fit the bed variation in a laterally averaged, z-coordinate hydrodynamic model, Int. J. Numer. Methods Fluids 44 (2004) 1185–1205.

[12] A. Adcroft, C. Hill, J. Marshall, Representation of topography by shaved cells in a height coordinate ocean model, Monthly Weather Rev. 125 (1997) 2293–2315.

[13] V. Casulli, G.S. Stelling, Numerical simulation of 3D quasi-hydrostatic free-surface flows, J. Hydraulic Eng. 124 (1998) 678–686.

[14] X. Chen, A fully hydrodynamic model for three-dimensional, free-surface flows, Int. J. Numer. Meth. Fluids 42 (2003) 929–952.

[15] UNESCO, Algorithms for computation of fundamental properties of seawater, UNESCO Technical Papers in Marine Science, Number 44, UNESCO, Paris, 1983, 53pp.

[16] X. Chen, A free-surface correction method for simulating shallow water flows, J. Comput. Phys. 189 (2003) 557–578.

[17] M.J. Briggs, C.E. Synolakis, G.S. Harkins, D.R. Green, Laboratory experiment of Tsunami runup on a circular island, Pure Appl. Geophys. 144 (1995) 569–593.

[18] P.L.-F. Liu, Y.-S. Cho, M.J. Briggs, U. Kanoglu, C.E. Synolakis, Runup of solitary waves on a circular island, J. Fluid Mech. 302 (1995) 259–295.

[19] S.F Bradford, B.F. Sanders, Finite-volume model for shallow-water flooding of arbitrary topography, J. Hydraulic Eng. 128 (2002) 289–298.