Kirill D. Nikitin, Maxim A. Olshanskii, Kirill M. Terekhov, and Yuri V. Vassilevski*

A splitting method for free surface flows over partially submerged obstacles

https://doi.org/10.1515/rnam-2018-0009
Received January 11, 2018; accepted January 25, 2018

Abstract: The paper proposes a stable time-splitting method for the numerical simulation of free-surface viscous flows. The key features of the method are a semi-Lagrangian scheme for the level-set function transport improved with MacCormack predictor–corrector step with limiting strategy and an adaptive volume-correction procedure. The spatial discretization is done by a hybrid finite volume/finite difference method on dynamically adaptive hexahedral meshes. Numerical verification is done by comparing full-scale 3D numerical simulations of the sloshing tank and the coastal wave run-up with other numerical and experimental results known from the literature.

Keywords: Free surface, incompressible Navier–Stokes equations, time-splitting, level-set method, semi-Lagrangian method.

MSC 2010: 65F10, 65N22, 65F50

Free surface flows over submerged and partially submerged obstacles are of great practical interest. Examples of such flows in naval engineering include the baffled oil tanks and run-up of the sea waves over oil ridge or coastal constructions. Numerical simulations play an important role in predicting fluid motion and its interaction with structures in different situations. For successful numerical simulations one has to use free surface capturing techniques combined with local mesh adaptivity in order to resolve different length scales in fluid solution and geometry. For realistic large-scale 3D problems, free surface capturing and local mesh adaptivity cause serious challenge in designing an accurate and reliable numerical method. This work continues our efforts in developing such a method for 3D free surface flows using dynamically adapted octree hexahedral meshes (see [13–16, 18, 23, 24]).

In the present paper we propose a simple yet accurate and stable time-splitting scheme for the system of fluid flow equations coupled with the transport equation for the indicator function of the free surface. The presented approach features the semi-Lagrangian method for the surface transport enhanced with the MacCormack predictor–corrector step and a limiting strategy. We apply the level-set method to the free-surface transport description. Semi-Lagrangian method is also used for the re-initialization of the level-set function. A particular attention is paid here to the volume correction method applied after re-initialization and semi-Lagrangian advection steps. The volume correction proposed here is based on the *local* adjustments of the discrete level-set function.

In our earlier work [15] we proved the stability of the splitting semi-discrete scheme similar to the one studied here, but with only the first order semi-Lagrangian method for the fluid and level-set transport. An improvement for better accuracy of this approach on adaptive octree meshes was suggested in [24], where the semi-Lagrangian method was used with a higher order interpolation, a limiting strategy and a back and forth correction of the numerical solution. The resulting method was verified on several benchmarks with excellent results and used in realistic applications. For the spatial discretization we apply a hybrid finite volume/finite

Kirill D. Nikitin, Institute of Numerical Mathematics of the Russian Academy of Sciences, Moscow 119333 and Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences, Moscow 125047, Russia
Maxim A. Olshanskii, Department of Mathematics, University of Houston, Houston, TX 77204, USA
Kirill M. Terekhov, Institute of Numerical Mathematics of the Russian Academy of Sciences, Moscow 119333, Russia
***Corresponding author: Yuri V. Vassilevski,** Institute of Numerical Mathematics of the Russian Academy of Sciences, Moscow 119333, Moscow Institute of Physics and Technology, Dolgoprudny 141701, and Sechenov University, Moscow 119991, Russia.
E-mail: yuri.vassilevski@gmail.com

difference method on octree hexahedral meshes. The discretization includes a higher order interpolation with limiting strategy for the semi-Lagrangian transport (see details in [16, 24]). In this paper, we replace the computationally expensive semi-Lagrangian step with back and forth correction for transport by a cheaper yet accurate semi-Lagrangian MacCormack predictor–corrector method. Another new contribution here is an improved volume correction scheme applied after the transport and re-initialization steps. We avoid the redundant details about spatial discretization which can be found in [16, 24]. Only those details about spatial discretization that are important for the presentation of the limiting and volume correction procedures are included in this paper.

Let us comment on the related papers on this topic. The volume preserving variants of the level-set method based on localized mass correction were previously addressed in [1, 2, 25], the back and forth error compensation and correction methods were suggested to improve the accuracy of the level-set function transport in [6, 7], also with a limiting strategy [9], an idea to replace the more expensive back and forth error compensation with MacCormack correction in the context of the semi-Lagrangian approach was proposed in [22], semi-Lagrangian method for the re-initialization of an indicator function was used in [26].

The rest of the paper is organized into three sections. Section 1 introduces the mathematical model. Section 2 presents the time stepping scheme with an emphasis on new findings which provide higher order approximation in time and improve monotonicity and conservation properties of the overall method. Section 3 collects the results of numerical experiments which validate our approach for numerical simulation of free surface flows over partially submerged obstacles.

1 Mathematical model

A motion of viscous incompressible fluid is driven by the Navier-Stokes equations for unknown fluid velocity vector field $\mathbf{u}$ and pressure p

$$\begin{cases} \rho\left(\dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) - \operatorname{div}\boldsymbol{\sigma}(\mathbf{u}, \mathrm{p}) = \mathbf{g} \\ \nabla\cdot\mathbf{u} = 0 \end{cases} \quad \text{in } \Omega(t), \quad t \in (0, T] \tag{1.1}$$

where $\boldsymbol{\sigma}(\mathbf{u}, p) = \nu[\nabla\mathbf{u} + (\nabla\mathbf{u})^T] - p\,\mathbf{I}$ is the stress tensor of the fluid; $\mathbf{g}$ is the external force (e.g., gravity), ρ is the density, and ν is the kinematic viscosity, all are assumed to be given. The equations are posed in the time-dependent fluid domain $\Omega(t)$, $t \in [0, T]$. At the initial time $t = 0$ the domain and the velocity field are known:

$$\Omega(0) = \Omega_0, \qquad \mathbf{u}|_{t=0} = \mathbf{u}_0, \qquad \nabla\cdot\mathbf{u}_0 = 0. \tag{1.2}$$

Finding the evolution of the domain $\Omega(t)$ for $t > 0$ is a part of the problem. For the definition of the domain evolution we use the level-set approach. This is an Eulerian approach using the indicator function for the implicit definition of the domain $\Omega(t) \in \mathbb{R}^3$ occupied by the fluid:

$$\varphi(t, \mathbf{x}) = \begin{cases} < 0, & \mathbf{x} \in \Omega(t) \\ > 0, & \mathbf{x} \in \mathbb{R}^3 \setminus \overline{\Omega(t)} \\ = 0, & \mathbf{x} \in \Gamma(t), \quad \forall t \in [0, T]. \end{cases} \tag{1.3}$$

The *level-set* function $\varphi(t, \mathbf{x})$ is assumed to be at least Lipschitz continuous. Then for the free-surface flow, one can show that $\varphi(t, \mathbf{x})$ satisfies the transport equation [19]:

$$\frac{\partial\varphi}{\partial t} + \tilde{\mathbf{u}}\cdot\nabla\varphi = 0 \quad \text{in } \mathbb{R}^3 \times (0, T]. \tag{1.4}$$

The transport field $\tilde{\mathbf{u}}$ coincides with the velocity $\mathbf{u}$ in $\Omega(t)$ and is extended to $\mathbb{R}^3$. The initial condition (1.2) defines $\varphi(0, \mathbf{x})$. Equations (1.4) and (1.1) are coupled through the boundary equations for (1.1) and definitions of $\tilde{\mathbf{u}}$ and $\Omega(t)$.

To set up suitable boundary conditions, let $\overline{\partial\Omega(t)} = \overline{\Gamma_W} \cup \overline{\Gamma(t)} \cup \overline{\Gamma}_{\text{out}} \cup \overline{\Gamma}_{\text{in}}$, where Γ_W is the static boundary (walls), $\Gamma(t)$ is the free surface of fluid, Γ_{in} and Γ_{out} are inflow and outflow parts of the boundary, respectively. In general, the sets Γ_W, Γ_{in}, and Γ_{out} are time-dependent. On the static part of the flow boundary, the boundary condition $\mathcal{B}\mathbf{u}|_{\Gamma_W}$ for the velocity is given by either no-slip boundary condition

$$\mathbf{u} = \mathbf{0} \quad \text{on } \Gamma_W \tag{1.5}$$

or no-penetration and free-slip boundary conditions:

$$\mathbf{u} \cdot \mathbf{n} = \mathbf{0}, \qquad \frac{\partial(\mathbf{u} \cdot \mathbf{t}_i)}{\partial \mathbf{n}} = 0, \quad i = 1, 2, \quad \text{on } \Gamma_W \tag{1.6}$$

where $\mathbf{t}_i$ and $\mathbf{n}$ are tangential and normal vectors on Γ_W. We assume that $\mathbf{u}$ is given on Γ_{in} and $\boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n} = \mathbf{0}$ on Γ_{out}. The flows of our interest have large Weber numbers and so we ignore the capillary forces. Hence, on the free surface boundary $\Gamma(t)$ one has vanishing normal stress,

$$\boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n} = \mathbf{0} \quad \text{on } \Gamma(t) \tag{1.7}$$

where $\mathbf{n}$ is the normal vector to $\Gamma(t)$. The normal vector $\mathbf{n}$ to the implicitly defined $\Gamma(t)$ can be computed as $\mathbf{n} = \nabla\varphi/|\nabla\varphi|$ or simply $\mathbf{n} = \nabla\varphi$, if the level-set function φ possesses the signed distance property, i.e., satisfies the Eikonal equation

$$|\nabla\varphi| = 1. \tag{1.8}$$

2 Discretization in time

In this section, we describe the time-stepping procedure. The numerical method splits the coupled fluid–level-set system of equations into the transport problem for φ, convection–diffusion problem for velocity, and the Poisson problem for the pressure. The divergence-free constrain is enforced by the projection as in the classical projection schemes by Chorin, Yanenko, Pironneau and others (see, for example, [4, 20]).

At each time step $t = t_n$ we use the notation $\mathbf{u}^n$, p^n, and φ^n for approximations to the velocity field $\mathbf{u}(t_n)$, the pressure $p(t_n)$, and the level-set function $\varphi(t_n)$. The initial condition (1.2) provides $\mathbf{u}^0 = \mathbf{u}(t_0)$ and $\varphi^0 = \varphi(t_0)$. The fluid domain is implicitly given by φ^n through $\Omega_n := \{\mathbf{x} \in \mathbb{R}^3 \;:\; \varphi^n(\mathbf{x}) < 0\}$. The time-stepping scheme is as follows.

Given $\mathbf{u}^n$ and φ^n such that $\operatorname{div} \mathbf{u}^n = 0$, $|\nabla\varphi^n| = 1$, $n = 0, 1, \dots$, we find $\mathbf{u}^{n+1}$, p^{n+1}, and φ^{n+1} by performing the following steps:

1: The mesh is refined according to the prediction of the new position of the zero level set. The refinement is important for large time steps when Step 2 may advect the zero level set into coarse cells with a loss of well-resolved features.
2: Semi-Lagrangian step, $\Omega_n \to \Omega_{n+1}$. Given φ^n and $\mathbf{u}^n$, compute φ^{n+1} by a semi-Lagrangian MacCormack predictor–corrector scheme (see Section 2.1).
3: Volume correction. Update the level-set function φ^{n+1} to conserve the volume of fluid (see Section 2.2).
4: Remeshing. Locally update the octree mesh by adapting it to $\partial\Omega_{n+1}$;
5: Re-interpolation. Map all discrete variables to the new grid (see Section 2.4).
6: Re-initialization. Update the level-set function φ^{n+1} to satisfy the Eikonal equation (1.8) and make the volume correction again (see Section 2.3).
7: Convection–diffusion solve. Compute the new velocity field $\widetilde{\mathbf{u}^{n+1}}$ in Ω_{n+1} by solving the convection–diffusion equation (see Section 2.5).
8: Projection step. Project the vector field $\widetilde{\mathbf{u}^{n+1}}$ onto the discrete divergence-free subspace. Compute the new velocity $\mathbf{u}^{n+1}$ and pressure p^{n+1} (see Section 2.6).

In the following sections we discuss these steps in more detail.

The stability of a simpler semi-discrete scheme was studied in [15]. The simple method is built of semi-Lagrangian steps for both φ^n and $\mathbf{u}^n$, diffusion step for velocity $\widetilde{\mathbf{u}^{n+1}}$; it does not include re-initialization, volume correction, remeshing, or re-interpolation. This scheme is shown to conserve global momentum and angular momentum, and to satisfy an energy inequality.

2.1 Semi-Lagrangian step

The physical velocity field $\mathbf{u}^n$ is defined in Ω_n. To evolve φ, we need to extend $\mathbf{u}^n$ to $\mathbb{R}^3$ (in practice, to a bulk computational domain). We use the closest-point extension of the velocity at the boundary to the exterior of fluid $\mathbf{u}^n|_{\Omega_n} \to \widetilde{\mathbf{u}}^n|_{\mathbb{R}^3}$. The semi-Lagrangian step is the solution of the characteristic equation backward in time

$$\frac{\partial \mathbf{x}(\tau)}{\partial \tau} = \widetilde{\mathbf{u}}^n(\mathbf{x}(\tau)), \qquad \mathbf{x}(t_{n+1}) = \mathbf{y}, \qquad \mathbf{y} \in \mathbb{R}^3, \quad \tau \in [t_{n+1}, t_n]. \tag{2.1}$$

Equation (2.1) defines an isomorphism $\mathbf{X} : \mathbf{y} \to \mathbf{x}(t_n)$ on $\mathbb{R}^3$; one finds φ^{n+1} from $\varphi^{n+1}(\mathbf{y}) = \varphi^n(\mathbf{X}(\mathbf{y}))$. The numerical integration of (2.1) is based on the trapezoidal rule

$$\mathbf{x}\Big(t_n + \frac{\Delta t}{2}\Big) = \mathbf{x}_0 - \frac{\Delta t}{2}\mathbf{u}(\mathbf{x}_0, t_n), \qquad \mathbf{x}(t_n) = \mathbf{x}_0 - \Delta t \widetilde{\mathbf{u}}^{n+1/2} \tag{2.2}$$

where $\widetilde{\mathbf{u}}^{n+1/2}$ is extrapolated linearly in time:

$$\widetilde{\mathbf{u}}^{n+1/2} = (1+\eta)\mathbf{u}\left(\mathbf{x}\left(t_n + \frac{\Delta t}{2}\right), t_n\right) - \eta \mathbf{u}\left(\mathbf{x}\left(t_n + \frac{\Delta t}{2}\right), t_{n-1}\right), \qquad \eta = \frac{t_{n+1} - t_n}{t_n - t_{n-1}}, \qquad \Delta t = t_n - t_{n+1}.$$

The tri-cubic interpolation combined with a limiter to enforce monotonicity [24] is used to define the values of φ in $\mathbf{x}(t_n)$. The same operations are performed for the 'missing' velocity values in $\mathbf{x}(t_n + \Delta t/2)$.

The semi-Lagrangian method as described above produces an error which can be reduced by the MacCormack predictor–corrector scheme with a limiter. We describe this correction procedure below. Backward integrating in time, interpolating and limiting define a nonlinear operator $\mathcal{B}(\varphi^{n+1})$. In the same way the numerical integration can be performed forward in time. It defines another nonlinear operator $\mathcal{F}(\varphi^n)$. A combination of $\mathcal{B}$ and $\mathcal{F}$ in a MacCormack predictor–corrector type of method and using 'minmod' limiting procedure gives the following algorithm.

Algorithm 1 Semi-Lagrangian MacCormack method with limiting.

```
1: Perform forward semi-Lagrangian step $\widehat{\varphi}^{n+1} = \mathcal{F}\varphi^n$.
2: Perform backward semi-Lagrangian step $\widehat{\varphi}^n = \mathcal{B}\widehat{\varphi}^{n+1} = \mathcal{B}\mathcal{F}\varphi^n$.
3: Compute error estimate $e = \frac{1}{2}(\varphi^n - \widehat{\varphi}^n) = \frac{1}{2}(\mathcal{I} - \mathcal{B}\mathcal{F})\varphi^n$ and correct $\widetilde{\varphi}^{n+1} = \widehat{\varphi}^{n+1} + e = (\mathcal{F} + \frac{1}{2}(\mathcal{I} - \mathcal{B}\mathcal{F}))\varphi^n$.
4: Perform backward semi-Lagrangian step $\widetilde{\varphi}^n = \mathcal{B}\widetilde{\varphi}^{n+1}$ and compute error estimate $\hat{e} = \varphi^n - \widetilde{\varphi}^n - e = (\mathcal{I} - \mathcal{B})e$.
5: Compute $\tilde{e}$ by limiting $e$ at nodes where $|\hat{e}| > |e|$ using (2.3).
6: Compute $\varphi^{n+1} = \widehat{\varphi}^{n+1} + \tilde{e}$.
```

We note that if at node $\mathbf{x}_0$ one detects $|\hat{e}(\mathbf{x}_0)| > |e(\mathbf{x}_0)|$, the limiting of e is applied at all nodes $\mathbf{x}_i$ involved in the interpolation procedure for $\mathbf{x}_0$. This is done by inspecting a row $r_{\mathcal{F}}(\mathbf{x}_0)$ and a row $r_{\mathcal{B}}(\mathbf{x}_0)$ of the discrete operators $\mathcal{F}$ and $\mathcal{B}$ as shown below:

```
for all $\mathbf{x}_0$ initialize $\tilde{e}(\mathbf{x}_0) = e(\mathbf{x}_0)$
for all $\mathbf{x}_0$ s.t. $|\hat{e}(\mathbf{x}_0)| > |e(\mathbf{x}_0)|$
    for all $\mathbf{x}_i$ contributing to $r_{\mathcal{F}}(\mathbf{x}_0)$ or $r_{\mathcal{B}}(\mathbf{x}_0)$
        $\tilde{e}(\mathbf{x}_i) = \text{minmod}(e(\mathbf{x}_0), \tilde{e}(\mathbf{x}_i))$.        (2.3)
```

The function minmod(a, b) is:

$$\operatorname{minmod}(a,b) = \begin{cases} \min(a,b), & a, b > 0 \\ \max(a,b), & a, b < 0 \\ 0, & \text{otherwise.} \end{cases}$$

The method presented here is a simpler analogue of the semi-Lagrangian BFECC method with the limiter [9, 24].

2.2 Volume correction

Advection and re-initialization steps for the level-set function lead to the divergence (loss or gain) of the fluid volume. The higher accuracy of the MacCormack predictor-corrector method ameliorates the situation, but does not eliminate the volume loss/gain, which becomes perceptible over long-time simulations. Therefore, we apply volume correction after advection and re-initialization sub-steps.

The volume correction is performed through the following steps.

Algorithm 2 Volume correction.

```
1: Compute the level set correction constants $\varphi_C$ in each surface computational cell (see Section 2.2.2).
2: Average out the cell-wise level set corrections $\varphi_C$ to nodes adjacent to surface computational cells. This
   gives the nodal function $\varphi_N$. Extension of the nodal correction level-set function $\varphi_N$ to the whole domain
   (see Sections 2.2.2–2.2.3).
3: Solve a nonlinear problem for a constant $Q$ such that the correction of level-set function $\varphi^{n+1}$ by $Q\,\varphi_N$
   minimizes the difference between the volume of $|\Omega_n|$ and $|\Omega_{n+1}|$ (see Section 2.2.4).
```

2.2.1 An estimate of error in fluid volume

From the semi-Lagrangian method we know backward characteristics. Using this information, we consider a cell $C(t^{n+1})$ intersected by the free surface at time t^{n+1} (further we call any such cell 'surface cell') and track its original $C(t^n)$, which is a polyhedron. In the nodes of $C(t^n)$ we interpolate the level-set function $\widehat{\varphi}^n$ with the third order method of interpolation with limiter (see [24]) and calculate the volume of the subset of $C(t^n)$ where $\widehat{\varphi}^n$ is negative, fluid volume for $C(t^n)$ denoted further by $V(\widehat{\varphi}^n, C(t^n))$.

The error at the cell is computed as a weighted difference between the fluid volume for $C(t^n)$ and the fluid volume for the cell at initial position $C(t^{n+1})$ with the advected level set φ^{n+1} as follows:

$$\mathrm{Err} = \left| V\left(\varphi^{n+1}, C(t^{n+1})\right) V\left(C(t^n)\right) - V\left(\widehat{\varphi}^n, C(t^n)\right) V\left(C(t^{n+1})\right)\right| . \tag{2.4}$$

Note that here we also account for possible compression or expansion of the cell due to the violation of the incompressibility condition for interpolated velocity. If we ignore this effect, Err is computed as the error indicator based on the interpolations of different orders, as it is done in embedded Runge–Kutta time-stepping methods [5].

The volume is calculated by splitting each computational cell into tetrahedra and extracting an isosurface of the nodal level-set function on each tetrahedron. The isosurface splits the tetrahedron into either two prisms or into a prism and a tetrahedron. Each prism can be split into three tetrahedra. Therefore, the space occupied by liquid can be split into a set of tetrahedra, which provides the cell fluid volume as the sum of the volumes of all tetrahedra. Note, that a computational cell is not necessarily a hexagon but may contain hanging nodes on octree mesh. We account these nodes when we split the cell into tetrahedra.

2.2.2 Local correction

For each given surface cell C we use a combination of the secant algorithm and the conjugate gradient algorithm as in [2] to find such a constant φ_C that

$$\mathrm{Err}(\varphi_C) = \left| V\left(\varphi^{n+1} + \varphi_C, C(t^{n+1})\right) V\left(C(t^n)\right) - V\left(\hat{\varphi}^n, C(t^n)\right) V\left(C(t^{n+1})\right)\right| = 0. \tag{2.5}$$

Further for each node $\mathbf{x}$ adjacent to any surface cell C we average nodal volume correction according to

$$\varphi_N(\mathbf{x}) = \sum_k \varphi_{C_k} V(C_k) \Big/ \sum_k V(C_k) \tag{2.6}$$

where the summation runs over all cells sharing $\mathbf{x}$.

2.2.3 Smooth extension

To avoid non-smooth artifacts on the surface after the volume correction, we first iteratively project φ_N at nodes $\mathbf{x}$ adjacent to the surface. At the kth iteration of the algorithm for each node $\mathbf{x}_n$ adjacent to a surface cell C we find the closest-point projection $\mathbf{x}_s = \mathbf{x}_n - \varphi^{n+1}\nabla\varphi^{n+1}$ on the surface and assign $\varphi_N^k(\mathbf{x}_n) = \tilde{\varphi}_N^{k-1}(\mathbf{x}_s)$ where $\tilde{\varphi}$ is the trilinear interpolation. The algorithm stops once either a critical number of iterations or a steady state is reached.

For the rest of the nodes, φ_N is also iteratively extended by similar procedure with

$$\mathbf{x}_s = \mathbf{x}_n - \mathrm{sign}(\varphi^{n+1})\nabla\varphi^{n+1}\triangle x.$$

Here $\triangle x$ is twice as big as the size of the smallest mesh cell. This results in extension of φ_N to the whole computational domain. In practice, φ_N is extended to a narrow band around the surface consisting of only few layers of computational cells.

2.2.4 Minimization problem

Finally, we solve the following nonlinear problem for the unknown factor Q using the secant algorithm

$$\left| V\left(\varphi^{n+1} + Q\varphi_N, \Omega^{\mathrm{comp}}\right) - V\left(\varphi^n, \Omega^{\mathrm{comp}}\right)\right| = 0 \tag{2.7}$$

where Ω^{comp} is the computational domain. Addition of $Q\varphi_N$ with the optimal factor Q to the level-set function φ^{n+1} completes the volume correction algorithm.

2.3 Re-initialization method

After application of the semi-Lagrangian method to the transport of the level-set function φ^{n+1}, the latter does not satisfy equation (1.8). The level-set function φ^{n+1} is corrected to satisfy Eikonal equation by the following algorithm.

Algorithm 3 Re-initialization.

```
1: Re-initialize the level-set function in cells adjacent to the surface.
2: Estimate volume loss near surface.
3: Correct volume loss with the algorithm from the previous section.
4: Correct the level-set function in the rest of the domain.
```

2.3.1 Re-initialization near surface

We first perform re-initialization in the set of cells closest to the free surface. For each such cell we extract the triangulated isosurface of $\varphi^{n+1} = 0$. To achieve this, we split each computational cell into a set of tetrahedra accounting for the presence of hanging nodes in the octree mesh. On each tetrahedron we compute the surface corresponding to $\varphi^{n+1} = 0$.

At each node k shared by a surface cell with the center $\mathbf{x}_k$ we seek for the closest triangle and obtain the initial position for the closest point $\hat{\mathbf{x}}_s$ on the surface. We further improve the position of the point $\hat{\mathbf{x}}_s$ to $\mathbf{x}_s$ using the gradient descent algorithm $\mathbf{x}_s = \hat{\mathbf{x}}_s - \varphi^{n+1}\nabla\varphi^{n+1}$ to find the zero of the level-set function. The latter is computed with the 4-th order interpolation method with the limiter. Since the level-set function does not satisfy the signed distance property, one may iteratively refine the position of $\mathbf{x}_s$.

The corrected value for level set at node k becomes $\widetilde{\varphi}_k^{n+1} = \operatorname{sign}(\varphi_k^{n+1})\,|\mathbf{x}_s - \mathbf{x}_k|$.

2.3.2 Volume loss estimation and correction

The algorithm for the volume correction is similar to the one described in Section 2.2.2, except for the estimation of the volume error indicator. For each surface cell C we estimate the volume divergence as follows:

$$\mathrm{Err} = \left| V\left(\varphi^{n+1}, C\right) - V\left(\widetilde{\varphi}^{n+1}, C\right)\right|. \tag{2.8}$$

The volume correction for the re-initialization algorithm proceeds similarly to Algorithm 2.

2.3.3 Semi-Lagrangian re-initialization

The semi-Lagrangian method is used for the extension of the corrected level-set function to the rest of the domain. In other words, the method is applied to all the nodes except for those, where $\widetilde{\varphi}^{n+1}$ was defined already in Section 2.3.1. Therefore, the extension does not shift the interface and does not change the sign of the level-set function. The re-initialization step presented below is an iterative process and is close to a method from [26]. At the kth iteration, for each node i with vector of coordinates $\mathbf{x}_i$ and level set value $\widetilde{\varphi}_i^{n+1,k-1}$ we compute $\mathbf{x}_s = \mathbf{x}_i - \operatorname{sign}\left(\widetilde{\varphi}_i^{n+1,k-1}\right)\nabla\widetilde{\varphi}^{n+1,k-1}/|\nabla\widetilde{\varphi}^{n+1,k-1}|\triangle x$. Then we interpolate the level-set function at the point $\mathbf{x}_s$ with the 4-th order method with limiter. Next we check the sign of $\widetilde{\varphi}_i^{n+1,k}\widetilde{\varphi}^{n+1,k-1}(\mathbf{x}_s)$. If it is positive, then we do not cross the zero level set and we set $\widetilde{\varphi}_i^{n+1,k} = \widetilde{\varphi}^{n+1,k-1}(\mathbf{x}_s) + \operatorname{sign}\left(\widetilde{\varphi}_i^{n+1,k-1}\right)\triangle x$ with account of the distance passed by the characteristic. Otherwise, we search for a point $\mathbf{x}_z$ on the segment between $\mathbf{x}_i$ and $\mathbf{x}_s$, such that $\widetilde{\varphi}^{n+1,k-1}(\mathbf{x}_z) = 0$ and define $\widetilde{\varphi}_i^{n+1,k} = \operatorname{sign}(\varphi_k^{n+1,k-1})\,|\mathbf{x}_z - \mathbf{x}_k|$.

Usually, the correction of the level-set function is needed just in a band of a few cells away from the surface. The width of the band may increase with larger time steps.

2.4 Remeshing and re-interpolation

Remeshing step 4 is needed for a better resolution of the moving free surface. The grid adaptation accounts for the new position of the zero level set of φ^{n+1}. In principle, the adaptation can be based on the information given by the nodal values of φ^{n+1} and discrete derivatives of φ^{n+1}. In numerical experiments in this paper we use the following indication for grid refinement: the cell is split if its nodal values have both negative and positive signs. This means that the zero isosurface of φ^{n+1} passes through the cell. Such indicator produces a thin layer of highly refined cells near the free surface.

After the remeshing step all discrete variables (level-set function, velocity, pressure) have to be re-interpolated to the new grid. The re-interpolation for the pressure is based on the limited second order least squares reconstruction. For each component of the velocity the re-interpolation is linear along the direction of the component and second-order least squares in the transversal directions. For the level-set function, we use the third-order interpolation.

We note that the time stepping scheme involves two stages of remeshing. At step 1 the mesh is refined prior the advection of the level-set function. Large time steps may result in large shifts of the zero level set and hence some geometrical information about $\partial\Omega_n$ may be lost due to the shift of the free surface to the region with coarser cells. To prevent this, we refine all cells having at least one node where the advected level-set function (predicted by the first order characteristic) changes its sign.

2.5 Convection–diffusion step

Next we handle viscous and inertia terms. We denote $\Gamma_1 = \Gamma_W \cup \Gamma_{\rm in}$, $\Gamma_2 = \Gamma(t_{n+1}) \cup \Gamma_{\rm out}$. The step consists in finding a discrete solution of the following boundary value problem: Find $\widetilde{\mathbf{u}^{n+1}}$ in Ω_{n+1} such that

$$\left\{\begin{aligned} &\frac{\alpha\widetilde{\mathbf{u}^{n+1}} + \beta\mathbf{u}^n + \gamma\mathbf{u}^{n-1}}{\triangle t_n} + (\mathbf{u}^n + \xi(\mathbf{u}^n - \mathbf{u}^{n-1}))\cdot\nabla\widetilde{\mathbf{u}^{n+1}} - \nu\Delta\widetilde{\mathbf{u}^{n+1}} = -\frac{\nabla p^n}{\rho} + \mathbf{g} \\ &\widetilde{\mathbf{u}^{n+1}}|_{\Gamma_{\rm in}} = \mathbf{u}_{\rm in}, \quad \mathcal{B}\widetilde{\mathbf{u}^{n+1}}|_{\Gamma_W} = \mathbf{0}, \quad (\nabla\widetilde{\mathbf{u}^{n+1}} + \nabla\widetilde{\mathbf{u}^{n+1}}^T)\mathbf{n}\Big|_{\Gamma_2} = 0. \end{aligned}\right. \tag{2.9}$$

Here $\xi = \triangle t_n/\triangle t_{n-1}$, $\alpha = 1+\xi/(\xi+1)$, $\beta = -(\xi+1)$, $\gamma = \xi^2/(\xi+1)$. We consider two options for the solution of (2.9). The first option is discretization of (2.9) by the hybrid finite volume/finite difference method from [16]. The discretization is hybrid since a finite volume method is applied to handle the inertia terms, whereas a finite difference method is used for the diffusion terms and the pressure gradient. The method approximates both differential operator and boundary conditions with the second order of accuracy. We refer to [16] for the treatment of the boundary conditions. The second option is to use the operator splitting framework: first we treat the inertia term by the semi-Lagrangian MacCormack method described in Section 2.1, then we apply the finite difference method to the diffusion term. The second option has the second order accuracy as well but may provide slightly smoother velocity fields.

2.6 Projection step

At this step we project the solution of (2.9) $\widetilde{\mathbf{u}^{n+1}}$ onto the subspace of discretely divergence-free functions:

$$\left\{\begin{aligned} \alpha(\mathbf{u}^{n+1} - \widetilde{\mathbf{u}^{n+1}})/\triangle t_n - \frac{\nabla q}{\rho} &= 0 \\ \operatorname{div}\mathbf{u}^{n+1} &= 0 \\ \mathbf{n}\cdot\mathbf{u}^{n+1}|_{\Gamma_1} = 0, \quad q|_{\Gamma_2} &= 0. \end{aligned}\right. \tag{2.10}$$

This equation recovers both divergence-free velocity $\mathbf{u}^{n+1}$ and pressure p^{n+1} due to

$$p^{n+1} = p^n - q + \nu\operatorname{div}\widetilde{\mathbf{u}^{n+1}}. \tag{2.11}$$

The ‘extra’ divergence term in the pressure correction step (2.11) minimizes numerical boundary layers in the pressure [8, 21]. The solution of the problem (2.10) reduces to the solution of the Poisson equation:

$$\left\{\begin{aligned} &-\frac{\Delta q}{\rho} = \frac{\alpha}{\triangle t_n}\operatorname{div}\widetilde{\mathbf{u}^{n+1}} \\ &q|_{\Gamma_2} = 0, \quad \frac{\partial q}{\partial\mathbf{n}}\Big|_{\Gamma_1} = 0. \end{aligned}\right. \tag{2.12}$$

3 Numerical experiments

In this section we consider two benchmark problems, the sloshing tank and run-up of the wave on the inclined bottom and partially submerged barrier. We also apply the method to simulate fluid sloshing in a container with baffles.

In sloshing simulations, we use the semi-Lagrangian MacCormack method both for the level-set function and inertia terms in the momentum equations. In these experiments, the free surface is smooth enough and hence we use the volume correction method in its simplest form (constant shift $\varphi_N \equiv 1$). In wave run-up simulations, however, we need the advanced volume correction presented in Section 2.2 in order to handle properly the complex dynamics of the free surface. Both options for the numerical solution of (2.9) can be considered for these simulations.

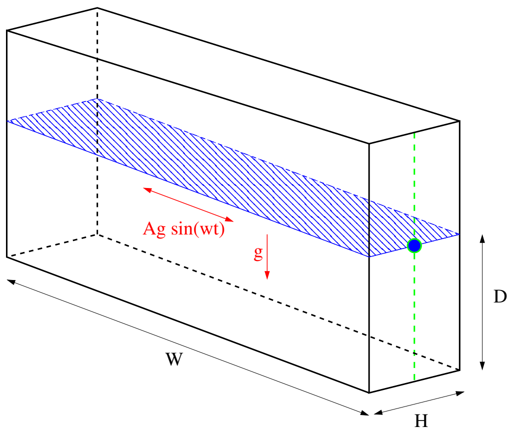
3.1 Sloshing tank: grid alignment

First we consider a benchmark problem [3, 10, 16] with a fluid sloshing in a rectangular tank. The setup of the experiment is illustrated in Fig. 1a. The tank dimensions are $W = 0.8$ m, $H = 0.1$ m, and $D = 0.3$ m. On the bottom and side boundaries the free-slip no-penetration conditions (1.6) are imposed.

The fluid with viscosity $\nu = 10^{-6}$ m^{2}s^{-1} and density $\rho = 10^3$ kg m^{-3} is driven into motion by two forces. In addition to a constant gravity force $g = 9.81$ m^{2}s^{-2}, we apply a horizontal shift with sinusoidal acceleration $Ag \sin \omega t$ with $A = 0.01$ and $\omega = 2\pi f$, $f = 0.89$ Hz. The horizontal shift is applied during first ten periods and terminated after that. The frequency of the shift is chosen to excite the first mode of wave motion, i.e., the motion with a wave length equal to the double width of the tank. The statistic of interest in this experiment is the evolution of free surface contact line along the middle line of the tank side walls. We shall call it the wave height.

The time evolution of the wave height is shown in Fig. 2. These data were computed for the 2D setting of the problem in [10]. These results are supposed to correspond well to the physical observations [17].

It was shown in [16] that the hybrid FV/FD method on octree meshes recovers correctly the time dependence of the water level at the tank walls. To study the accuracy of this method for complex boundaries, we compare the results of the simulation on grids aligned and non-aligned to the tank walls. In the second case we rotate the tank by 15° in the horizontal plane and obtain a stepwise computational domain shown in



(a)

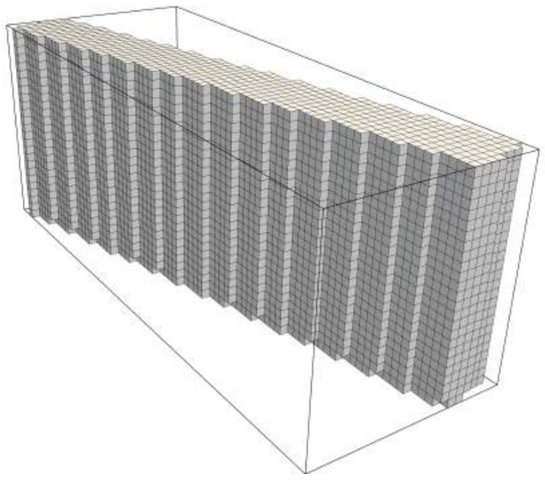

(b)

Fig. 1: (a) Problem setup for sloshing tank test, (b) example of the computational grid for the rotated domain with stepwise boundary.

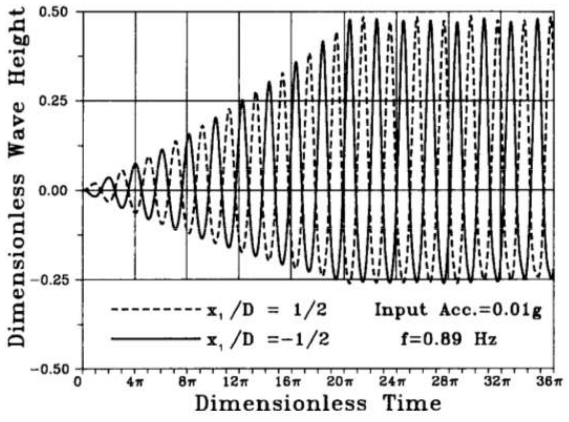


Fig. 2: Time histories of the wave height at two opposite tank walls from [10].

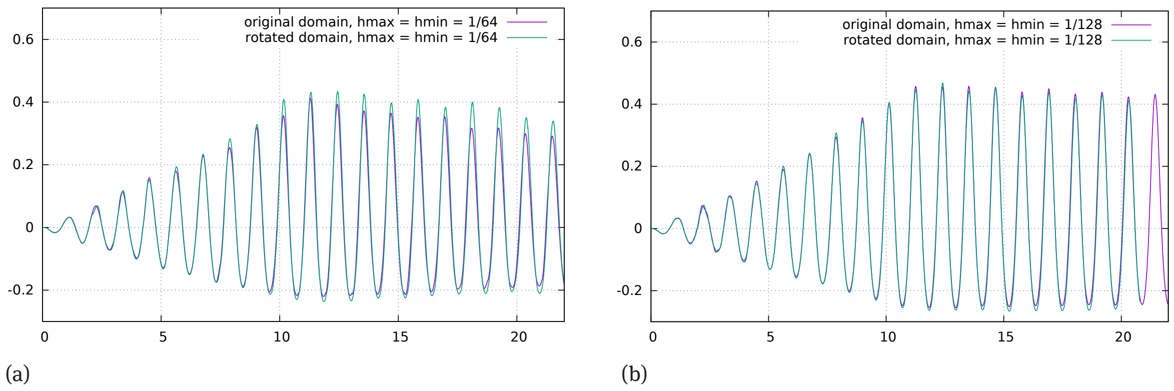


Fig. 3: Time histories of the wave height at the left tank wall for uniform grids: (a) $h_{\max} = h_{\min} = 1/64$, (b) $h_{\max} = h_{\min} = 1/128$.

Fig. 1b. A grid non-aligned to the boundary requires accurate treatment of boundary conditions [16] and may produce undesirable grid orientation impact to fluid flow.

The comparison of the wave height histories on the left tank wall of the original and rotated tank is shown in Fig. 3. On the coarse grid ($h_{\max} = h_{\min} = 1/64$), deviation in the wave heights is clearly observed in Fig. 3a. However, on the fine grid ($h_{\max} = h_{\min} = 1/128$), the wave heights curves are indistinguishable (see Fig. 3b).

The experiment demonstrates that grids non-aligned to the boundary and the axis of excitation do not cause any significant deterioration of accuracy to the numerical solution provided that the grid is fine enough. Therefore, the present method appears to be reliable tool for the numerical study of sloshing in tanks with complex boundaries.

3.2 Sloshing in tank with baffles

Internal baffles are widely used in the design of ocean tankers and liquid carrying trucks. Baffling is known to damp sloshing and thus it makes the container more stable during transportation (see Fig. 4).

In order to assess the damping effect of internal baffles, we consider the sloshing experiment from the previous section with three types of containers: a tank without baffle (see Fig. 5a), a tank with a bottom baffle (see Fig. 5b), a tank with a surface baffle (see Fig. 5c).

The computational meshes are dynamically refined to the free surface up to the meshsize h_{min}, while the coarsest cell size is fixed to $h_{\max}$. We use the following combinations of the mesh refinements: (1) $h_{\max} = h_{\min} = 1/64$, (2) $h_{\max} = h_{\min} = 1/128$, (3) $h_{\max} = 1/32$, $h_{\min} = 1/128$, and (4) $h_{\max} = 1/32$, $h_{\min} = 1/256$. In this simulation we use adaptive time step, $\Delta t_k = \min\{0.0187, \mathrm{CFL} \cdot h_{\min}/\max_{\mathbf{x}} |\mathbf{u}(\mathbf{x}, t_k)|\}$, with CFL = 1.0.

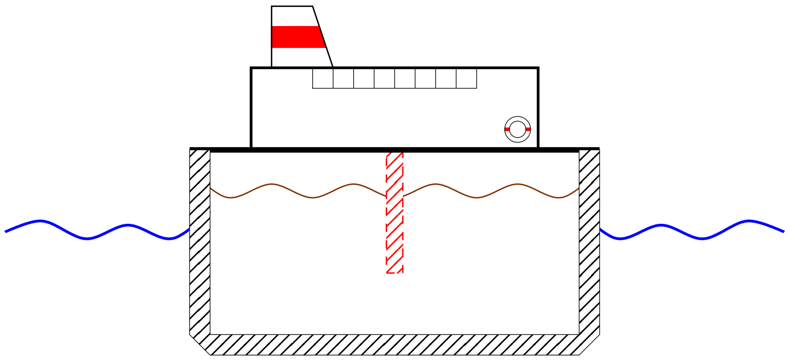


Fig. 4: Container of a tanker.

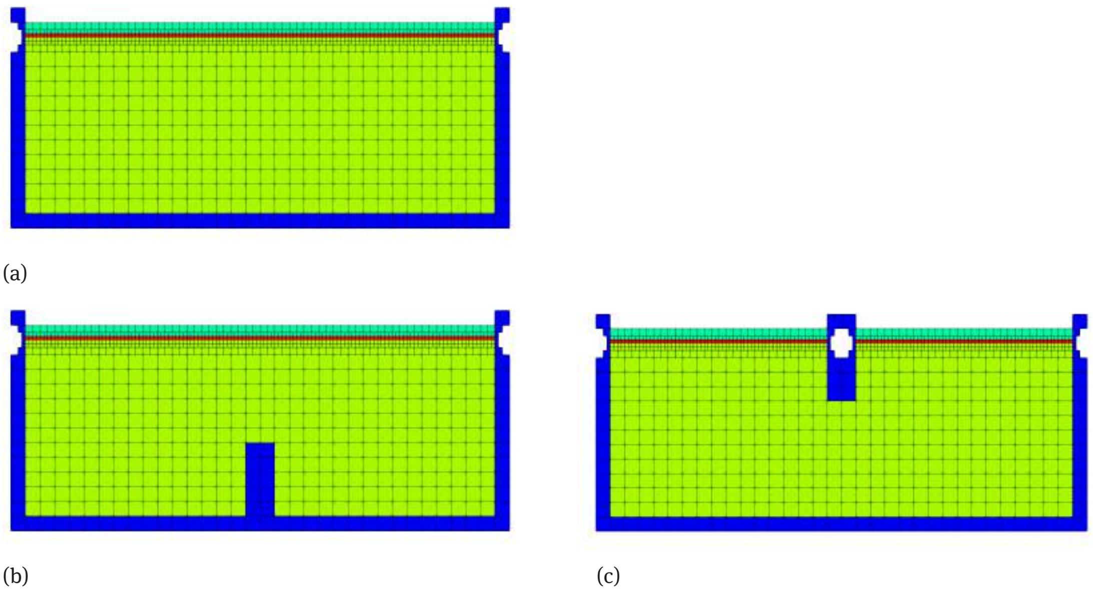


Fig. 5: Cutaway of the computational grid $h_{\max} = 1/32$, $h_{\min} = 1/128$: (a) no baffle, (b) a bottom baffle, (c) a surface baffle. Cell types are marked by colors: walls and baffles (blue), water (green), surface (red), air (light green).

In Fig. 6 we present the numerical sloshing in containers of the above three types on the dynamic adaptive grids with $h_{\max} = 1/32$, $h_{\min} = 1/256$. Waves in the container without baffle preserve their amplitude for many periods since the induced wave length is equal to the double width W of the tank and the computational method has low numerical dissipation [16]. In the container with the bottom baffle, the wave height is damped by a factor of 2.5. In the container with the surface baffle, the wave heights are 10 times smaller compared to the original experiment. The simulation demonstrates that the correct positioning of the internal baffle may reduce the sloshing effect significantly.

The grid convergence of the wave heights on the left wall is demonstrated in Fig. 7. In the container without baffle and in the container with the bottom baffle we observe the grid convergence, the frequencies of sloshing are equal to the excitation frequencies. In the container with the surface baffle finer grids produce a large number of additional modes.

3.3 Propagation of waves behind a semi-submerged barrier

For the final experiment we consider the interaction of run-up waves with a protective barrier. Protective barriers serve to reduce the wave load on floating objects, power units, piers, and platforms behind them. The

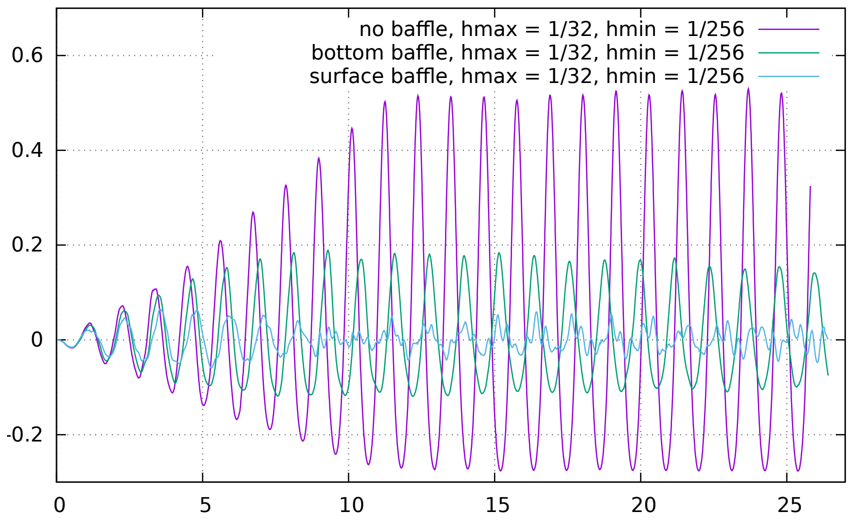


Fig. 6: Wave heights on the left wall of the sloshing tank for the experiment without a baffle, with a bottom baffle and with a surface baffle. All curves are computed on the grid with $h_{\max} = 1/32$, $h_{\min} = 1/256$.

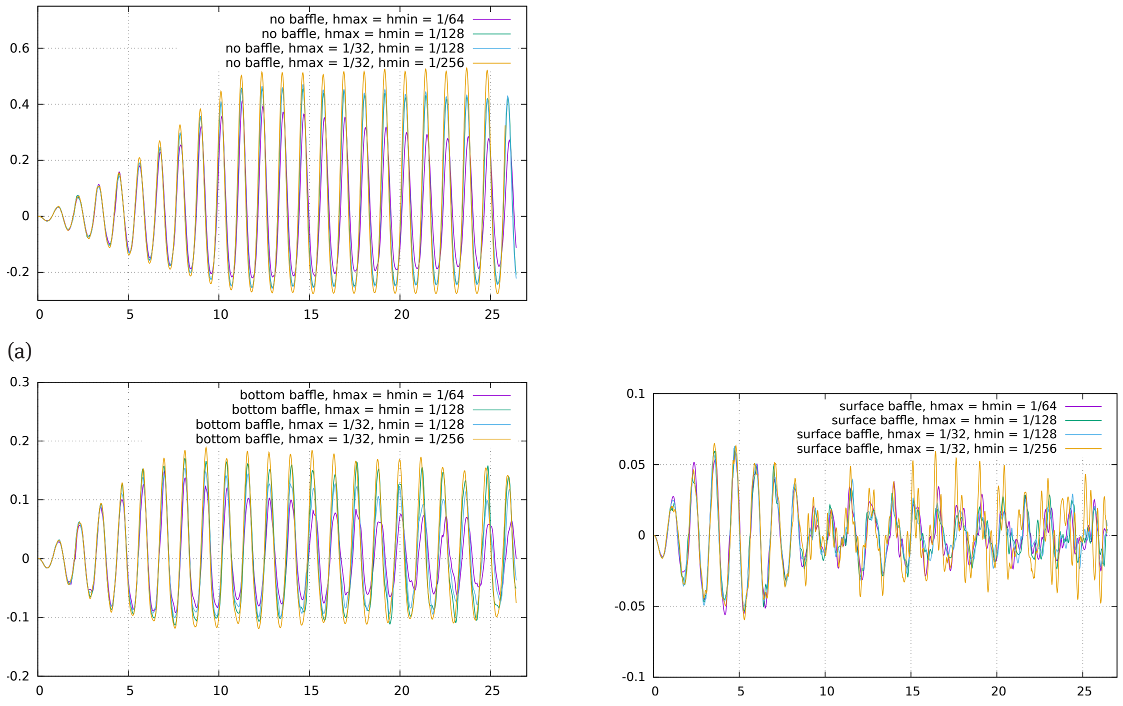


Fig. 7: Grid convergence of the wave heights on the left wall of the sloshing tank for the experiment (a) without baffle, (b) with a bottom baffle, (c) with a surface baffle.

physical experiment is described and compared with the numerical simulation results in [11]. The simulation in [11] was based on the nonlinear planar potential flows model [12] discretized by finite differences on adaptive moving grids.

The experiment setup is presented in Fig. 8. The fluid waves are excited by the wave generator: a vertical column of height h_w filled with water generates the waves upon its release. The wave generator is separated from the main tank by the vertical wall with a gap z_e at the bottom. The first part of tank floor is horizontal and

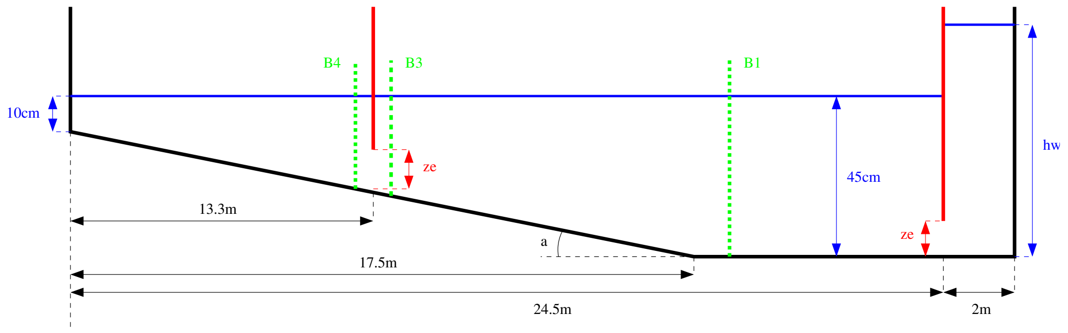
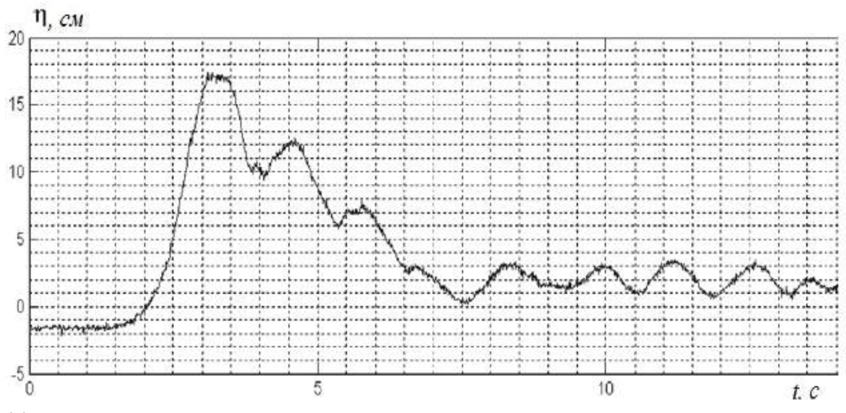
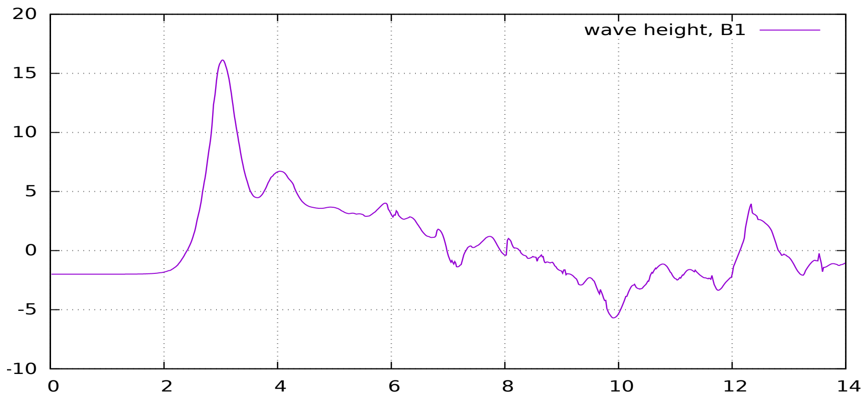


Fig. 8: Scheme of the experiment with propagating wave.



(a)



(b)

Fig. 9: Wave heights in cross-section B_1: (a) physical experiment, (b) present numerical model.

the second one is inclined at angle α, $\tan(\alpha) = 0.02$. We put the vertical barrier above the inclined part with the same gap z_e and study how it affects the wave propagation. In this experiment we neglect the viscosity $\nu = 0$.

For verification of our numerical model we collect data from three height recorders located at the horizontal part (B1), 0.16 m before the barrier (B3), and 3.1 m after the barrier (B4).

The wave generator provides physically meaningful wave front with the main wave and secondary waves. Implementation of the wave generation technique similar to the physical experiment allowed us to reproduce measurements of the wave recorders with good accuracy. Measured and numerical wave heights at cross-section B_1 corresponding to the largest (in time) height 17 cm are shown in Fig. 9. The numerical wave was produced by the wave generator with $h_w = 85$ cm and $z_e = 10$ cm. The wave profile in time is shown in Fig. 10.

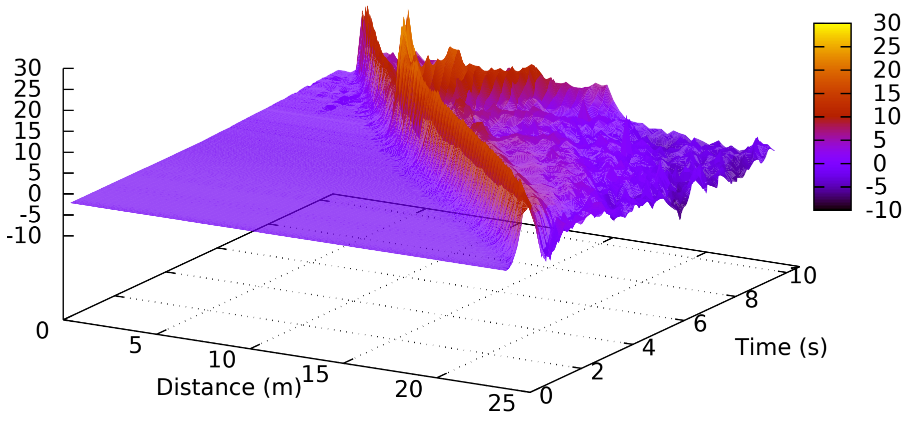


Fig. 10: Wave profile.

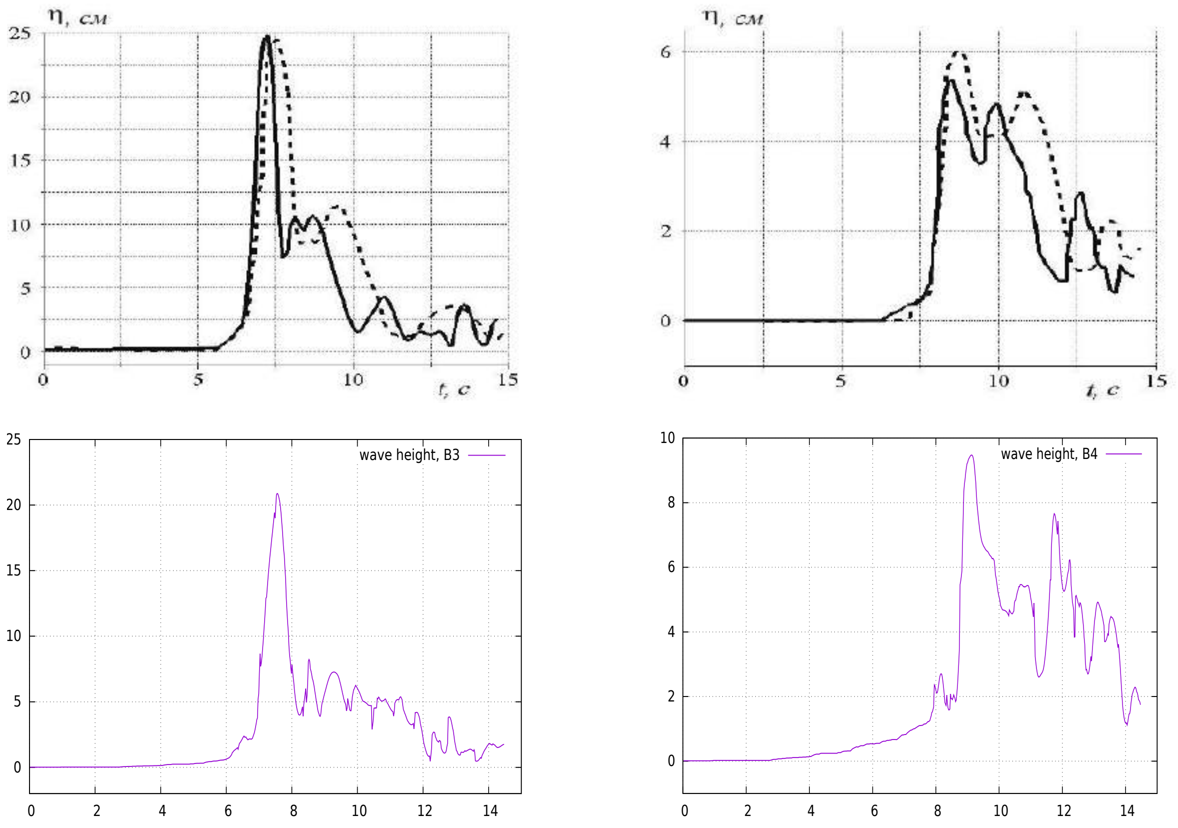


Fig. 11: Wave heights. Top level: experimental data (dashed) and numerical results from [11] (solid). Bottom level: present numerical model. Left: before the barrier (cross-section B_3), right: after the barrier (cross-section B_4).

To produce higher waves in the physical experiment, one pumps more water in the tank of the wave generator. This reduces the initial water level in the basin by 2 cm. The numerical model reproduces this situation as well.

The wave generator with $h_w = 75$ cm and $z_e = 8$ cm produces the maximum wave height ≈ 11cm at cross-section B_1. We compare it to a bit smaller physical wave with height 10.2 cm. Figure 11 presents the comparison of the wave heights before (cross-section B_3) and after (cross-section B_4) the barrier. We observe good matching of the wave arrival time and the number of registered waves. The computed splash before the barrier is slightly lower than the one observed in the experiment, while the wave height after the barrier is somewhat larger than measured.

4 Conclusions

We introduced a time-splitting method for the numerical simulation of free-surface viscous flows. For accurate and stable numerical results, several components of the method needed careful considerations. The critical components include higher order numerical schemes for the discrete level-set function transport and re-initialization. Furthermore, a volume correction method based on local updates and higher order volume error estimators was found important for the overall accuracy of the approach. The numerical method was tested for several benchmark problems, where direct comparison with physical experiment or other numerical results is possible. Such comparison demonstrated that the present method is able to predict statistics of practical interest. Application of the method to the simulation of fluid sloshing in baffled tanks revealed the expected damping properties of various baffling and confirmed the functionality of our numerical approach.

Funding: The work has been supported by the Russian Science Foundation (Grant 14-11-00434). The work was based on the computational technology within RFBR grants 14-01-00830, 17-01-00886.

References

[1] R. Ausas, G. Buscaglia, and E. Dari, A mass-preserving geometry-based reinitialization method for the level set function. *Serie Mécanica Computacional* **27** (2008), 13–32.
[2] R. Ausas, E. Dari, and G. Buscaglia, A geometric mass-preserving redistancing scheme for the level set function. *Int. J. Numer. Methods Fluids* **65** (2011), No. 8, 989–1010.
[3] M. Behr, *Stabilized finite element methods for incompressible flows with emphasis on moving boundaries and interfaces*. Ph.D. Thesis, University of Minnesota, 1992.
[4] A. Chorin, Numerical solution of the Navier-Stokes equations. *Math. Comp.* **22** (1968), 745–762.
[5] J. Dormand and P. Prince, A family of embedded Runge–Kutta formulae. *J. Comput. Appl. Math.* **6** (1980), No. 1, 19–26.
[6] T. Dupont and Y. Liu, Back and forth error compensation and correction methods for removing errors induced by uneven gradients of the level set function. *J. Comput. Phys.* **190** (2003), No. 1, 311–324.
[7] T. Dupont and Y. Liu, Back and forth error compensation and correction methods for semi-Lagrangian schemes with application to level set interface computations. *Math. Comp.* (2007), 647–668.
[8] J. Guermond, P. Minev, and J. Shen, An overview of projection methods for incompressible flows. *Comput. Meth. Appl. Mech. Engrg.* **195** (2006), 6011–6045.
[9] L. Hu, Y. Li, and Y. Liu, A limiting strategy for the back and forth error compensation and correction method for solving advection equations. *Math. Comp.* **85** (2016), No. 299, 1263–1280.
[10] A. Huerta and W. Liu, Viscous flow with large free surface motion. *Comp. Methods Appl. Mech. Engrg.* **69** (1988), No. 3, 277–324.
[11] E. Kamynin, V. Maximov, I. Nudner, K. Semenov, and G. Khakimzyanov, Interaction of surface waves with a partially submerged screen. *Zbornik Radova Konferencije MIT* (2011), 173–178 (in Russian).
[12] G. Khakimzyanov, Yu. Shokin, V. Barakhnin, and N. Shokina, *Numerical Simulation of Fluid Flows with Surface Waves*. Publishing House of SB RAS, Novosibirsk, 2001 (in Russian).
[13] K. Nikitin and Yu. Vassilevski, Free surface flow modelling on dynamically refined hexahedral meshes. *Russ. J. Numer. Anal. Math. Modelling* **23** (2008), No. 5, 469–485.
[14] K. Nikitin, M. Olshanskii, K. Terekhov, and Yu. Vassilevski, A numerical method for the simulation of free surface flows of viscoplastic fluid in 3D. *J. Comput. Math.* **29** (2011), 605–622.
[15] K. Nikitin, M. Olshanskii, K. Terekhov, and Yu. Vassilevski, A splitting method for numerical simulation of free surface flows of incompressible fluids with surface tension. *Comput. Meth. Appl. Math.* **15** (2015), No. 1, 59–78.
[16] K. Nikitin, M. Olshanskii, K. Terekhov, Yu. Vassilevski, and R. Yanbarisov, An adaptive numerical method for free surface flows passing rigidly mounted obstacles. *Comput. & Fluids* **148** (2017), 56–69.
[17] W. Noh, CEL: a time dependent two-space-dimensional coupled Eulerian–Lagrangian code. In: *Methods in Computational Physics* (Eds. B Alder, S. Fernbach, and M. Rotenberg), Vol. 3. Academic Press, New York, 1964, pp. 117–179.

[18] M. Olshanskii, K. Terekhov, and Yu. Vassilevski, An octree-based solver for the incompressible Navier–Stokes equations with enhanced stability and low dissipation. *Comput. & Fluids* **84** (2013), 231–246.
[19] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, New York, 2002.
[20] O. Pironneau, On the transport–diffusion algorithm and its applications to the Navier–Stokes equations. *Numerische Mathematik* **28**, (1982), 309–332.
[21] A. Prohl, *Projection and Quasi-Compressibility Methods for Solving the Incompressible Navier–Stokes Equations*. B. G. Teubner, Stuttgart, 1997.
[22] A. Selle, R. Fedkiw, B. Kim, Y. Liu, and J. Rossignac, An unconditionally stable MacCormack method. *J. Sci. Comput.* **35** (2008), No. 2–3, 350–371.
[23] K. Terekhov, K. Nikitin, M. Olshanskii, and Yu. Vassilevski, CFD technology for 3D simulation of large-scale hydrodynamic events and disasters. *Russ. J. Numer. Anal. Math. Modelling* **27** (2012), No. 4, 399–412.
[24] K. Terekhov, K. Nikitin, M. Olshanskii, and Yu. Vassilevski, A semi-Lagrangian method on dynamically adapted octree meshes. *Russ. J. Numer. Anal. Math. Modelling* **30** (2015), No. 6, 363–380.
[25] S. Van der Pijl, A. Segal, C. Vuik, and P. Wesseling, A mass-conserving level-set method for modelling of multi-phase flows. *Int. J. Numer. Meth. Fluids* **47** (2005), No. 4, 339–361.
[26] Y. Wang, S. Simakhina, and M. Sussman, A hybrid level set-volume constraint method for incompressible two-phase flow. *J. Comput. Phys.* **231** (2012), No. 19, 6438–6471.