City Research Online

City, University of London Institutional Repository

Citation: Zhang, N., Yan, S. & Ma, Q. (2025). Hybrid ISPH_GNN method for simulating violent wave-structure interactions using wave-only data for training. Journal of Computational Physics, 540, 114277. doi: 10.1016/j.jcp.2025.114277

This is the published version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/35674/

Link to published version: https://doi.org/10.1016/j.jcp.2025.114277

Copyright: City Research Online aims to make research outputs of City, University of London available to a wider audience. Copyright and Moral Rights remain with the author(s) and/or copyright holders. URLs from City Research Online may be freely distributed and linked to.

Reuse: Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

City Research Online: http://openaccess.city.ac.uk/ publications@city.ac.uk

Journal of Computational Physics 540 (2025) 114277

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Hybrid ISPH_GNN method for simulating violent wave-structure interactions using wave-only data for training

Ningbo Zhang, Shiqiang Yan, Qingwei Ma [*]

School of Science & Technology, City St George's, University of London, London, UK

ARTICLE INFO

Keywords:
graph neural network (GNN)
ISPH
PPE
violent WSIs
wave-only data
transfer learning

ABSTRACT

It has been well-known that the incompressible Smoothed Particle Hydrodynamics (ISPH) is a powerful method for simulating violent wave-structure interactions (WSIs) concerned in marine engineering. However it is time consuming, primarily due to the need of solving pressure Poisson's equation (PPE) involved in this method. In our previous publications, we are first to propose a hybrid approach embedding the graph neural network (GNN) into ISPH method to form the hybrid ISPH_GNN method for simulating free-surface problems, where the GNN is employed to replace solving the PPE. We demonstrated that the computational time for evaluating the pressure using GNN can be of one order less than that spent by directly solving PPE to achieve similar level of accuracy. More importantly, we also demonstrated in our previous publications that the GNN trained only on data for wave-only (referring to no structure or obstacles in wave fields) cases can be satisfactorily applied to the cases for wave-floater interactions. However, what we have not previously studied is if the GNN trained only by using wave-only cases can be used for simulating violent WSIs. One of the original contributions of this paper is to answer this question. In addition, transfer learning has been proved to be a machine learning (ML) technique that can significantly enhance efficiency and improve the performance in other fields but has not been explored in the hybrid ISPH_GNN method. Another original contribution of this paper is to explore the potential of integrating transfer learning with the ISPH_GNN for simulating violent WSIs. Specifically, we will demonstrate that the GNN trained by using data from sloshing and dam-breaking cases without any structure (termed as wave-only data in this paper) can be employed to simulate more complex cases, such as water entry of an object, wave impact on a trapezoidal structure and wave interaction with an oscillating wave surge converter, all of which involve violent WSIs. We will also demonstrate that the transfer learning technique with use of a small volume of additional data has a potential in enhancing the prediction accuracy of the ISPH_GNN. Furthermore, we will show that the ISPH_GNN significantly reduces computational time for pressure evaluation in violent WSI cases, even with a more significant reduction compared to wave-floater interaction cases studied in our previous work. These highlight the strong potential of the ISPH_GNN for broad applications in marine engineering, opening a novel route to employ ML without need of generating data for very complex cases of violent WSIs.

[*] Corresponding author.
E-mail address: q.ma@city.ac.uk (Q. Ma).

https://doi.org/10.1016/j.jcp.2025.114277
Received 3 December 2024; Received in revised form 6 July 2025; Accepted 4 August 2025
Available online 6 August 2025
0021-9991/© 2025 The Authors. Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, researchers have increasingly explored the applications of machine learning (ML) techniques in the field of computational fluid dynamics (CFD) [1,2]. For instance, ML-based data-driven methods have been employed as alternatives to traditional CFD approaches, which typically involve directly solving governing equations for fluid simulations [3]. These methods have been used for tasks such as predicting the positions and velocities of fluid particles [4], and modeling wave–structure interactions for moored floating structures [5]. Additionally, some physics-informed machine learning models have been developed to solve fluid dynamics problems, including the use of physics-informed neural networks (PINNs) for incompressible [6,7] and compressible flows [8], as well as a physics-informed Long Short-Term Memory network (LSTM-PINN) for predicting wave–structure interactions [9]. These method does not involve solving any physics equations. Moreover, researchers have explored the potential of ML models to replace complex and computationally intensive steps in traditional CFD workflows with other parts solved from physics equations, which as termed as hybrid PE-ML method in this paper. In this context, Tompson et al. [10] utilized a convolutional neural network (CNN) to perform pressure projection in Eulerian-based solvers, thereby enhancing the efficiency of Eulerian fluid simulations. Ling et al. [11] introduced a novel neural network architecture as an alternative tool for the prediction of Reynolds stress anisotropy tensor for turbulence modeling. Furthermore, Kochkov et al. [12] developed ML-based algorithms capable of accurately solving equations on coarser grids by substituting components of conventional CFD solvers affected by resolution loss with high-performing learned alternatives, thus speeding up fluid simulations.

The meshfree particle method is a significant category within CFD approaches. As a popular Lagrangian meshless particle method [13–18], the incompressible Smoothed Particle Hydrodynamics (ISPH) has demonstrated considerable promise as a numerical tool for simulating fluid dynamics problems, including breaking waves and wave–structure interactions [19–23]. The ISPH method relies on a projection approach that involves solving a pressure Poisson equation (PPE) for pressure evaluation [24], which is the most computationally intensive step in ISPH simulations. To address this challenge, Zhang et al. [25] firstly introduced a hybrid PE-ML method in the ISPH model using a convolutional neural network (CNN) to replace the traditional PPE solution for pressure evaluation in free-surface flow simulations. However, since CNNs operate through convolutions on a uniform Cartesian grid, the input data—such as field information and domain geometry, must be formatted on regular grids. This requirement poses inherent limitations for CNNs, making it challenging to apply them to particle-based problems where particles are irregularly distributed.

Graph Neural Networks (GNNs) [26,27], as part of the geometric deep learning framework, are built on graph structures where information at nodes can be organized in either regular or irregular formats. This flexibility gives GNNs a significant advantage for particle-based simulations, such as Lagrangian ISPH simulations, where data are typically represented by irregularly distributed particles. Several studies have employed GNNs to model fluid dynamics within particle-based (mesh-free) frameworks. Kumar and Vantassel [28] developed the Graph Network Simulator (GNS), which learns the underlying physics and predicts the flow of particle and fluid systems by discretizing the domain into nodes that represent particle groups, with edges indicating local particle interactions. Li and Farimani [29] proposed a GNN-based, data-driven method for fluid simulations using a Lagrangian particle representation. Bult on success of GNN in other fields, Zhang et al. [30] further developed the hybrid PE-ML method based on ISPH, in which the GNN replaces the pressure Poisson equation (PPE) solver in conventional ISPH, named as ISPH_GNN in this paper for convenience. In these studies, the cases generating data for training GNN remain within a similar range of cases as those they studied using the trained GNN.

Moreover, we have applied ISPH_GNN with a GNN model trained only on data from simple wave propagation cases without any structure in wave fields to satisfactorily simulate wave-floater interactions in our previous publication [31]. However, one area that remains unexplored is whether the ISPH_GNN with a GNN model trained only by wave-only data (referring to the data for cases where there are no any structures in wave fields) can be used for simulating violent WSIs. The study of violent WSIs is particularly challenging due to the complex and highly nonlinear nature of these phenomena. Accurately understanding violent WSIs is essential for the safe and efficient design of marine infrastructure. Thus, it is critical to study violent WSIs for advancing the field of ocean and coastal engineering. To the best of our knowledge, no existing research has employed the GNN-supported ISPH method for simulating violent WSIs.

In this paper, we adopt the ISPH_GNN to simulate violent WSI problems. A key contribution of this paper is demonstrating that ISPH_GNN with the GNN trained on data from dam-breaking and sloshing cases without any structure can effectively simulate a broad range of violent WSI problems, including water entry of an object, the solitary wave impact on a trapezoidal structure and the regular wave interaction with an oscillating wave surge converter (OWSC). Additionally, by utilizing training data from dam-breaking and sloshing scenarios, which are characterized by considerably smaller computational domains compared to the regular and solitary wave propagation cases used to generate data in our prior work [31], both data generation and model training times are significantly reduced. This suggests that a well-performed ISPH_GNN model can be achieved at a low training cost by selecting the appropriate training data. These findings highlight the robust potential of the ISPH_GNN model for violent WSIs simulations.

Another key contribution of this paper is the investigation of transfer learning potential in enhancing the accuracy of the GNN model when combined with the ISPH method for violent WSIs simulations. Transfer learning is a well-established technique for improving both predictive accuracy in machine learning models by leveraging knowledge from pre-trained models on large datasets to enhance performance on target tasks with limited data availability [32–34]. This is especially valuable in CFD applications, where generating extensive training datasets is often costly and time-consuming. By transferring learned features and patterns from related problems, transfer learning can minimize the need for extensive retraining on target problems. Additionally, it can accelerate the training process, allowing the fine-tuned model to build upon prior knowledge rather than starting from scratch, leading to faster convergence and reduced computational costs. While several studies have demonstrated the effectiveness of transfer learning for improving the performance of GNN models [35,36], there has been no research on its application in the context of a hybrid ISPH_GNN method. Here, we demonstrate the potential of applying transfer learning to simulating violent WSIs.

2. Mathematical and numerical formulation of ISPH_GNN

2.1. Lagrangian ISPH

In the current ISPH framework, the fluid is treated as incompressible and its behaviour is governed by the Navier-Stokes and continuity equations, which are expressed as follows:

$$\frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho}\nabla p + \mathbf{g} + \nu\nabla^2\mathbf{u} \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{2}$$

where $\mathbf{u}$, p and ρ represent the fluid's velocity, pressure, and density, respectively, while $\mathbf{g}$ denotes gravitational acceleration and ν is the fluid's kinematic viscosity. Eq. (1) is expressed in a Lagrangian framework, where D/Dt signifies the material derivative that follows the movement of the fluid particles.

On the solid boundaries, the velocity and pressure are subject to the following Neumann boundary conditions:

$$\mathbf{u} \cdot \mathbf{n} = \mathbf{U} \cdot \mathbf{n} \tag{3}$$

$$\mathbf{n} \cdot \nabla p = \rho(\mathbf{n} \cdot \mathbf{g} - \mathbf{n} \cdot \dot{\mathbf{U}}), \tag{4}$$

where $\mathbf{n}$ represents the unit normal vector of the solid boundary, $\mathbf{U}$ and $\dot{\mathbf{U}}$ denote the velocity and acceleration of the solid boundary, respectively.

At the free surface, the following pressure condition is applied:

$$p = 0 \tag{5}$$

A projection method is employed to address the governing equations and boundary conditions, consisting of three primary steps:

Prediction Step: This involves forecasting the intermediate velocity and position of particles using

$$\mathbf{u}^* = \mathbf{u}_t + \Delta\mathbf{u}^* \tag{6}$$

$$\Delta\mathbf{u}^* = \left(\mathbf{g} + \nu\nabla^2\mathbf{u}\right)\Delta t \tag{7}$$

$$\mathbf{r}^* = \mathbf{r}_t + \mathbf{u}^*\Delta t \tag{8}$$

where $\mathbf{u}_t$ and $\mathbf{r}_t$ represent the velocity and position of a particle at time t, respectively; Δt is the time step size; $\mathbf{u}^*$ and $\mathbf{r}^*$ denote the intermediate velocity and position of the particle.

Pressure Projection: By substituting Eqs. (6) and (7) into Eqs. (1) and (2), the pressure Poisson equation (PPE) can be derived as $\nabla^2 p_{t+\Delta t} = \frac{\rho\nabla\cdot\mathbf{u}^*}{\Delta t}$. Based on the work in [19,21], an alternative formulation [21,23] is applied to all internal fluid particles.

$$\nabla^2 p_{t+\Delta t} = \frac{\Psi}{\Delta t} \tag{9}$$

where $\Psi = \alpha\frac{\rho-\rho^*}{\Delta t} + (1-\alpha)\rho\nabla\cdot\mathbf{u}^*$, with α being the blending coefficient, typically chosen as a small value like 0.01. ρ^* represents the particle density at the intermediate time step, estimated as $\sum_{j=1}^{N} m_j W\left(\mathbf{r}_{ij}^*\right)$, where N is the number of particles within the influence domain of particle i, m_j is the mass of the neighbouring particle j. In this study, the cubic B-spline kernel introduced by Monaghan and Lattanzio [37] is employed as the kernel function $W\left(\mathbf{r}_{ij}^*\right)$.

Correction step: Once the pressure $p_{t+\Delta t}$ at time $t+\Delta t$ is predicted, the velocity adjustment during the correction step is computed as follows:

$$\mathbf{u}^{**} = -\frac{\Delta t}{\rho}\nabla p_{t+\Delta t} \tag{10}$$

The particle's velocity and position at $t+\Delta t$ are subsequently updated using:

$$\mathbf{u}_{t+\Delta t} = \mathbf{u}^* + \mathbf{u}^{**} \tag{11}$$

$$\mathbf{r}_{t+\Delta t} = \mathbf{r}_t + \frac{\mathbf{u}_t + \mathbf{u}_{t+\Delta t}}{2}\Delta t \tag{12}$$

In each time step, the velocity divergence and viscous terms are estimated using classic SPH formulations [14,37], while the pressure gradient is determined by the semi-analytical finite difference interpolation scheme (SFDI) [38].

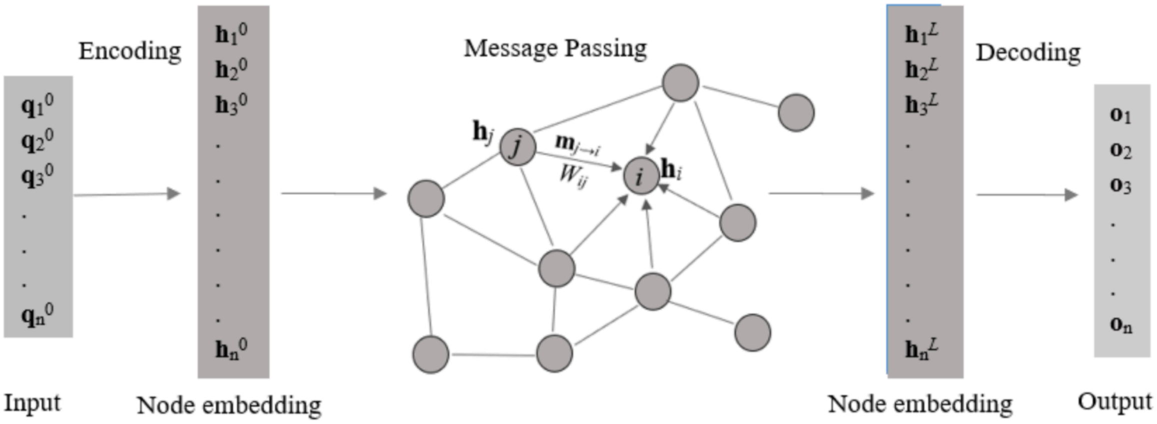


Fig. 1. Illustration of the GNN architecture.

$$\nabla \cdot \boldsymbol{u}_i = -\frac{1}{\rho_i}\sum_{j=1}^{N} m_j\left(\boldsymbol{u}_i - \boldsymbol{u}_j\right)\cdot \nabla_i W\left(\mathbf{r}_{ij}\right) \tag{13}$$

$$\nabla \cdot (\nu_i \nabla \boldsymbol{u}_i) = \sum_{j=1}^{N} 8m_j\left(\frac{\nu_i+\nu_j}{\rho_i+\rho_j}\frac{\boldsymbol{u}_{ij}\cdot \boldsymbol{r}_{ij}}{r_{ij}^2+\eta^2}\right)\cdot \nabla_i W\left(\mathbf{r}_{ij}\right) \tag{14}$$

$$\nabla P_i = \sum_{j=1,j\neq i}^{N} \frac{n_i^{x_k}B_{ij}^{x_m} - n_i^{xy}B_{ij}^{x_k}}{n_i^x n_i^y - n_i^{xy}n_i^{xy}}\left(P_j - P_i\right) \tag{15}$$

where m is the particle mass, η is a small number to prevent singularity, $\mathbf{u}_{ij} = \mathbf{u}_i - \mathbf{u}_j$, $n_i^{xy} = \sum_{j=1,j\neq i}^{N}\frac{\left(\mathbf{r}_j^{x_m}-\mathbf{r}_i^{x_m}\right)\left(\mathbf{r}_j^{x_k}-\mathbf{r}_i^{x_k}\right)}{|\mathbf{r}_{ij}|^2}W\left(\mathbf{r}_{ij}\right)$, $n_i^{x_m} = \sum_{j=1,j\neq i}^{N}\frac{\left(\mathbf{r}_j^{x_m}-\mathbf{r}_i^{x_m}\right)^2}{|\mathbf{r}_{ij}|^2}W\left(\mathbf{r}_{ij}\right)$ and $B_i^{x_m} = \sum_{j=1,j\neq i}^{N}\frac{\left(\mathbf{r}_j^{x_m}-\mathbf{r}_i^{x_m}\right)}{|\mathbf{r}_{ij}|^2}W\left(\mathbf{r}_{ij}\right)$, in which $x_m = x$ when $x_k = y$ or $x_m = y$ when $x_k = x$, and $\mathbf{r}^{x_m}$represents the position vector component along the x_m direction.

To solve the PPE (Eq. (9)), various discretization schemes have been devised for the Laplacian operator, with the following classic approach being widely used.

$$\nabla \cdot \left(\frac{1}{\rho}\nabla P_i\right) = \sum_{j=1}^{N}\frac{8m_j}{\left(\rho_i+\rho_j\right)^2}\frac{\left(P_i - P_j\right)\mathbf{r}_{ij}}{r_{ij}^2+\eta^2}\cdot \nabla_i W\left(\mathbf{r}_{ij}\right) \tag{16}$$

Further information on the numerical implementation, such as the discretization of the PPE, the methods for handling boundary conditions, and the approach for addressing the gradient and divergence operators in the conventional ISPH method, is provided in [21,23].

2.2. GNN model

The GNN model [29] utilized in this study operates by performing convolutions on graphs characterized by nodes and their spatial relationships [39]. As illustrated in Fig. 1, the GNN framework consists of three main phases: encoding, message-passing and decoding [30,33].

The function of the encoding phase is to transform the input data into network embeddings suitable for graph convolutions. For instance, in Fig. 2, initial input $\mathbf{q}_i^0$ for particle i is converted into node embedding $\mathbf{h}_i^0$ through a learnable encoder $en()$, which is built on a multi-layer perceptron (MLP) with N layers:

$$en\left(\mathbf{q}_i^l\right) = \sigma\left(\mathbf{w}_l\mathbf{q}_i^{l-1} + b_l\right) \; with \; l = 1, 2, 3...., n \tag{17a}$$

$$\mathbf{h}_i^0 = en\left(\mathbf{q}_i^n\right) \tag{17b}$$

where $\sigma()$ represents the activation function, while $\mathbf{w}_l$ and b_l refer to the learnable weight vector and bias term for the L-th layer of the MLP, respectively. The feature of the node at the L-th layer is denoted as $\mathbf{q}_i^l$. The resulting $\mathbf{h}_i^0$serves as the input for the graph convolution process described in the following sections.

Within the message-passing block, the effect of neighbouring node j on node i during the L-th layer convolution is represented by $\mathbf{m}_{j\to i}$, which is defined as follows.

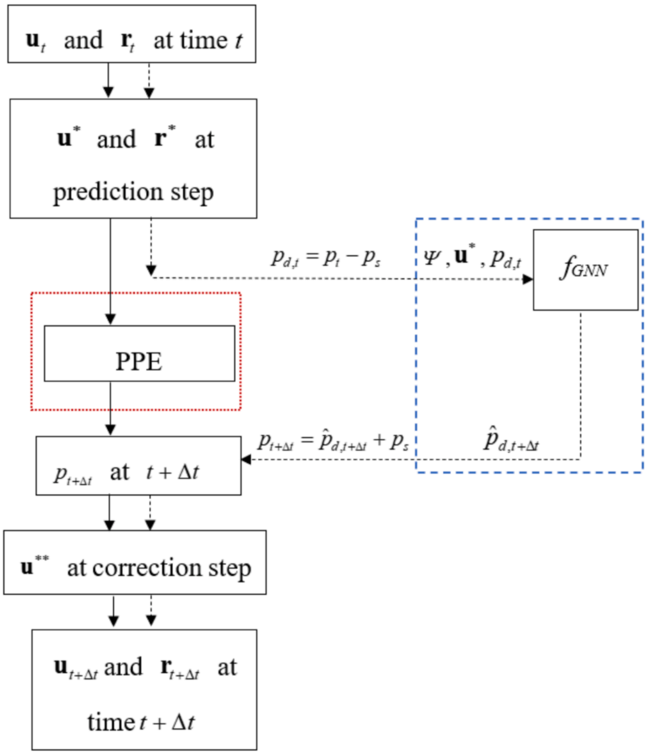

$\mathbf{u}_t$ and $\mathbf{r}_t$ at time t
$\mathbf{u}^*$ and $\mathbf{r}^*$ at prediction step
$p_{d,t} = p_t - p_s$
$\Psi, \mathbf{u}^*, p_{d,t}$
f_{GNN}
PPE
$p_{t+\Delta t} = \hat{p}_{d,t+\Delta t} + p_s$
$\hat{p}_{d,t+\Delta t}$
$p_{t+\Delta t}$ at $t+\Delta t$
$\mathbf{u}^{**}$ at correction step
$\mathbf{u}_{t+\Delta t}$ and $\mathbf{r}_{t+\Delta t}$ at time $t+\Delta t$

Fig. 2. Diagram of the ISPH integrated with GNN (the conventional ISPH flowchart shown with solid arrows, and the ISPH_GNN flowchart with dashed arrows).

$$\mathbf{m}^l_{j\to i} = \mathbf{h}^{l-1}_j W(r_{ij}, r_0) \tag{18}$$

where$\mathbf{h}^{l-1}_j$ denotes the node embedding of neighbor node j in the L-th layer of convolution, and the weight is given by $W(r_{ij},\ r_0) = \{ \begin{matrix} r_0/r_{ij} - 1 & 0 < r_{ij} \le r_0 \\ 0 & r_{ij} > r_0 \end{matrix}$, where r_0 is the radius of the influence domain, which is taken as a value of $1.7dx$ and dx is the initial particle spacing. The influence from all the neighbouring nodes is aggregated and given by

$$\mathbf{M}^l_i = \frac{\sum_{j=1}^{N} \mathbf{m}^l_{j\to i}}{\sum_{j=1}^{N} W(r_{ij}, r_0)} \tag{19}$$

In the L-th layer of message passing, the node embedding of the node i is calculated by

$$\mathbf{h}^l_i = \sigma\left(\mathbf{w}'_l \mathbf{M}^l_i + b'_l \mathbf{h}^{l-1}_i\right) \ \mathit{with}\ l = 1, 2, ..., L, \tag{20}$$

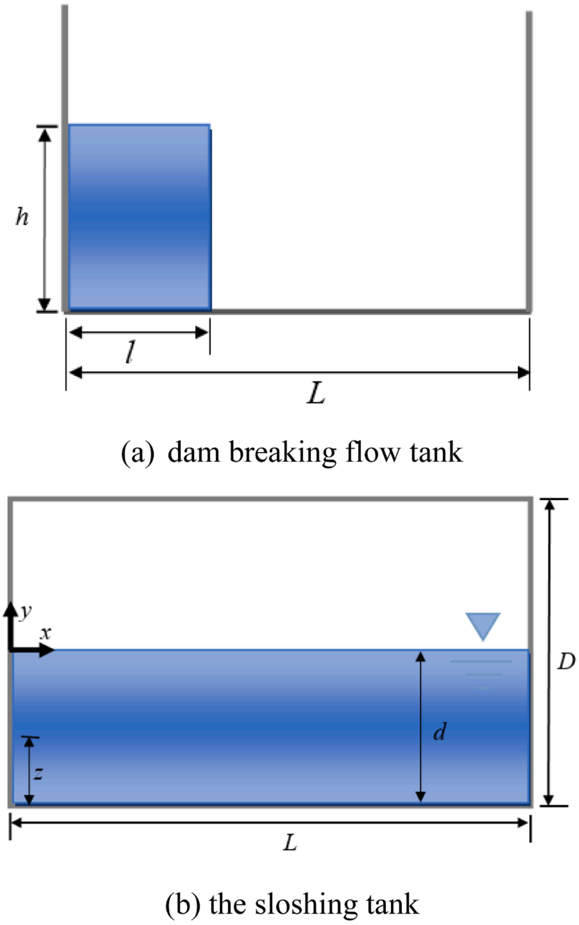
where $\mathbf{w}'_l$ and b'_l are the learnable weight vector and bias term in the l-th layer of message passing.

After completing the message passing through L layers, the output $\mathbf{o}_i$ (representing pressure in this paper) is obtained by decoding the final embedding $\mathbf{h}^L_i$using a learnable decoder *den*(), which is structured as a multi-layer perceptron (MLP) with N layers. This process follows a similar approach to Eq. (21) but uses different learnable weights $\mathbf{w}''_l$ and bias terms b''_l:

$$den\left(\mathbf{h}^l_i\right) = \sigma\left(\mathbf{w}''_l\ \mathbf{h}^{l-1}_i + b''_l\right) \ \mathit{with}\ l = L+1, L+2, ..., L+N \tag{21a}$$

$$\mathbf{o}_i = den\left(\mathbf{h}^{L+N}_i\right) \tag{21b}$$

In this study, both the encoder and decoder MLPs consist of three layers, and the number of recursive message-passing layers is set to two, following the studies in [30,34] and the results discussed in Sections 3 and 4. The intermediate message-passing block is implemented as an MLP with a structure of (64, 64). Gaussian Error Linear Units (GeLUs) [40] are used as the activation function for all layers to introduce non-linearity.



(a) dam breaking flow tank

(b) the sloshing tank

Fig. 3. Sketch of the numerical water tank for generating training data.

2.3. *Incorporating GNN with ISPH*

In the ISPH_GNN approach, a Graph Neural Network (GNN) is used to substitute the traditional PPE solver in conventional ISPH method. Fig. 2 depicts the flowchart of the ISPH_GNN process, in which solving the PPE in the conventional ISPH is replaced by the trained GNN model, represented by f_{GNN} in the figure with the details given below. This figure provides an overview of the GNN model, its integration with ISPH, and the training and implementation of the GNN. For more comprehensive information, refer to Zhang et al. [30].

2.3.1. *Input parameters*

To ensure accurate predictions, the input variables for the GNN model must effectively capture key hydrodynamic properties. According to the study in [30], the term Ψ in the right hand of PPE, the intermediate velocity $\mathbf{u}^*$ and the pressure $p_{d,t} = p_t - p_s$ at the previous time step are adopted as the input parameters to train the GNN model for predicting the pressure $\widehat{p}_{d,t+\Delta t}$. It's important to note that, during both training and prediction, the total pressure p is decomposed into two components: the reference pressure p_s, and the rest part of pressure p_d, i.e., $p = p_d + p_s$. The reference pressure p_s can be directly calculated by $\rho g\widetilde{z}$, where $\widetilde{z}$ is the vertical distance from the fluid particle to its nearest free-surface particle. More details about evaluating p_s can be found in [30]. The GNN model, represented by the function f_{GNN}, takes these inputs and predicts the pressure at the next time step $\widehat{p}_{d,t+\Delta t}$. Mathematically, the relationship is expressed as:

$$\widehat{p}_{d,t+\Delta t} = f_{GNN}\left(\Psi, \mathbf{u}^*, p_{d,t}, c_p\right) \tag{22}$$

here, c_p represents the network coefficients—specifically, all the weights and biases involved in Eqs. (17), (20), and (21). These coefficients are optimized during the training process, as detailed in Section 2.2.

Once the GNN model is trained, the pressure $\widehat{p}_{d,t+\Delta t}$ can be predicted using the input data Ψ, $\mathbf{u}^*$ and $p_{d,t}$ through the GNN model f_{GNN} with the network coefficients c_p. After making predictions with the GNN model f_{GNN}, the total pressure of fluid particles, required for updating the velocity field in Eq. (1), can be determined by:

$$p_{t+\Delta t} = \widehat{p}_{d,t+\Delta t} + p_s \tag{23}$$

Further discussions regarding the selection of input parameters can be found in [30].

2.3.2. Training data

In this study, the GNN model is trained using data generated from the cases for dam breaking and sloshing, which are conducted using the consistent second-order ISPH method (ISPH_CQ) [41]. The governing equations and boundary conditions in ISPH_CQ are identical to those in the conventional ISPH. However, ISPH_CQ employs the quadratic SFDI method (QSFDI) [42] for numerical approximations of the Laplacian operator, velocity divergence, associated Neumann boundary conditions, and pressure gradient. Some formulas for evaluating Laplacian operator and divergence by QSFDI are given in Appendix. More detailed information on ISPH_CQ can be referred to [41,42].

The dam breaking cases are simulated within a water tank with a horizontal length L, containing a rectangular water column of width l and height h, as illustrated in Fig. 3(a). For the violent sloshing wave scenarios, the setup is depicted in Fig. 3(b), where L represents the total length of the tank, D is the tank height, and d indicates the mean water depth. The tank undergoes periodic sway motion described by $X_s = a\sin(\omega t)$, where a and ω are the amplitude and frequency of the motion, respectively.

The parameters used to generate the training dataset are selected randomly. For the dam breaking cases, the height h and width l of the water column are chosen from the ranges $h/L = 0.2$ to 0.5 and $l/L = 0.2$ to 0.5, with L fixed at 2.0 m. The ISPH_CQ simulations use an initial particle size of 0.01 m and a time step of 0.001 s. A total of 60 dam breaking cases are generated, with each case consisting of 100 frames captured at 0.006 s intervals, covering the entire process from the initial state to the impact of the dam breaking flow on the right wall of the tank. For the violent sloshing cases, parameters are set as $d = 0.2L$, $D = 0.5L$, and $L = 0.6$ m. The amplitude a is randomly selected between $0.05L$ and $0.1L$, while the sloshing periods T are assigned values of 1.4, 1.5, and 1.6 s. An initial particle size of 0.01 m and a time step of 0.001 s are also used to generate the training data. In total, 60 sloshing cases are simulated, with each case containing 100 frames captured at 0.1 s intervals. The training dataset is composed of 30 dam breaking cases and 30 violent sloshing cases, randomly selected from the total simulations. The remaining cases, not included in the training dataset, are utilized as the test set.

2.3.3. Training and implementing the GNN

Using the provided training and testing datasets, the GNN is trained by minimizing the loss function, defined in this study as the sum of squared errors between the predicted and actual pressure values.

$$f_{obj} = \sum_{i}^{N} s_i \left(\widehat{p}_{d,i} - p_{d,i}\right)^2 \tag{24}$$

where $\widehat{p}_{d,i}$ and $p_{d,i}$ represent the predicted and actual pressures at particle i, respectively. The weight s_i is introduced to account for boundary effects. This weighting term s_i is defined as follows:

$$s_i = \begin{cases} 3 \; d_b \le 3.0 \cdot dx \\ 2 \; 3.0 \cdot dx < d_b \le 6.0 \cdot dx \\ 1 \; d_b > 6.0 \cdot dx \\ 0 \; \textit{free surface} \end{cases} \tag{25}$$

During the training and application of the GNN model, two types of boundary conditions: i.e. free-surface and solid boundaries need to be handled. For the free-surface condition, the discrepancy between predicted and actual pressures at the free surface is not considered during training. Instead, the GNN model ensures that the pressure at all free-surface particles is zero, in accordance with the free-surface condition described by Eq. (5). In practice, only interior fluid particles are considered during both training and testing stages, while boundary particles are excluded according to the physic boundary conditions given in Eq. (4). The pressure p_b at boundary particles is calculated directly from neighbouring fluid particles, following the method outlined in [43].

$$p_b = \frac{\sum_{f\in fluid}^{N} p_f W_{bf} + (\mathbf{g} - \dot{\mathbf{U}}) \cdot \sum_{f\in fluid}^{N} \rho_f \left(\mathbf{r}_f - \mathbf{r}_b\right) W_{bf}}{\sum_{f\in fluid}^{N} W_{bf}} \tag{26}$$

where p_f is the pressure of the fluid particle, ρ_f is the fluid density, W_{bf} is the kernel weight function, and N is the total number of fluid particles in the vicinity of the boundary particle. The position vectors of the fluid and boundary particles are represented by $\mathbf{r}_\mathrm{f}$ and $\mathbf{r}_\mathrm{b}$, respectively.

Once the model is trained, it is integrated into the numerical framework of the ISPH_GNN, as illustrated in Fig. 2.

2.4. ISPH_ GNN with transfer learning

2.4.1. Transfer learning

Transfer learning is a machine learning technique where a model developed for source tasks is reused as the starting point for a model on related target tasks. Instead of training a model from scratch on the target tasks, transfer learning leverages the knowledge gained from the pre-trained model on source tasks, demonstrated in other fields [32–35]. Mathematically, we can outline the concept

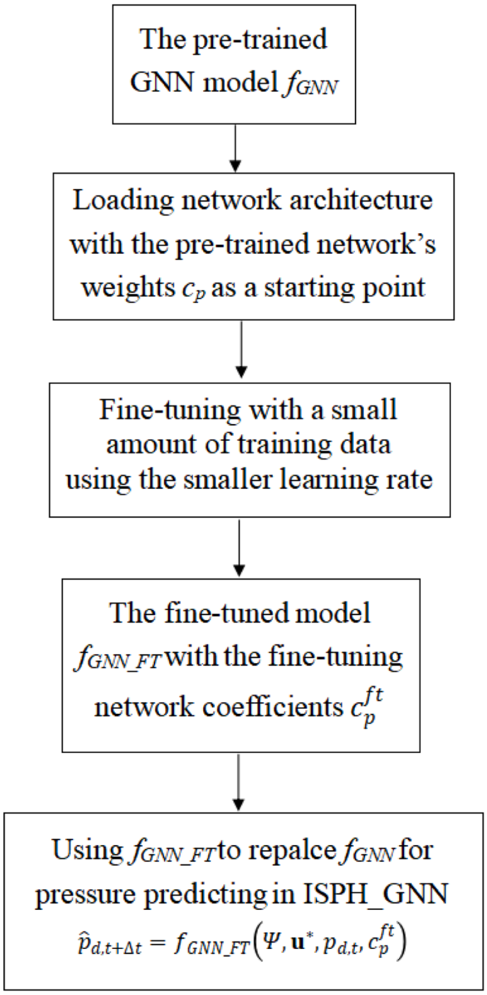

The pre-trained GNN model f_{GNN}
Loading network architecture with the pre-trained network's weights c_p as a starting point
Fine-tuning with a small amount of training data using the smaller learning rate
The fine-tuned model f_{GNN_FT} with the fine-tuning network coefficients c_p^{ft}
Using f_{GNN_FT} to repalce f_{GNN} for pressure predicting in ISPH_GNN
$\hat{p}_{d,t+\Delta t} = f_{GNN\_FT}\left(\Psi, \mathbf{u}^*, p_{d,t}, c_p^{ft}\right)$

Fig. 4. Illustration of ISPH_ GNN with transfer learning.

of transfer learning as follows:

Let a source task be defined by:

$$T_s = \{\mathbf{q}_s, \mathbf{o}_s, f_s(\cdot)\} \tag{27a}$$

where $\mathbf{q}_s$ is the input feature, as discussed in Section 2.3, $\mathbf{o}_s$ is the output feature (the pressure in this work) and f_s: $\mathbf{q}_s \rightarrow \mathbf{o}_s$ is the predictive function learned from the source data and it also represents the pre-trained GNN model f_{GNN} discussed in above section.

The corresponding target task can be defined by:

$$T_t = \{\mathbf{q}_t, \mathbf{o}_t, f_t(\cdot)\} \tag{27b}$$

where $\mathbf{q}_t$ is the input feature, $\mathbf{o}_t$ is the output feature and f_t: $\mathbf{q}_t \rightarrow \mathbf{o}_t$ is the predictive function for the target task.

The objective of transfer learning is to learn the target predictive function f_t for the target task T_t by leveraging knowledge from the source task T_s. This can be written as:

$$f_t(\mathbf{q}_t) = \text{Transfer}(f_s, \mathbf{q}_t), \tag{28}$$

where Transfer is a mechanism, e.g., fine-tuning, that incorporates information from the source function f_s to improve the prediction accuracy on the target task T_t.

2.4.2. Implementing transfer learning in ISPH_ GNN

A common transfer learning mechanism is fine-tuning, where the target model f_t can be initialized with the parameters of the pre-trained source model f_s and then update the model parameters using a small target dataset [32–35]. In this paper, the fine-tuning strategy is employed for transfer learning in combination with the ISPH_GNN. In the context of GNNs, a common transfer learning

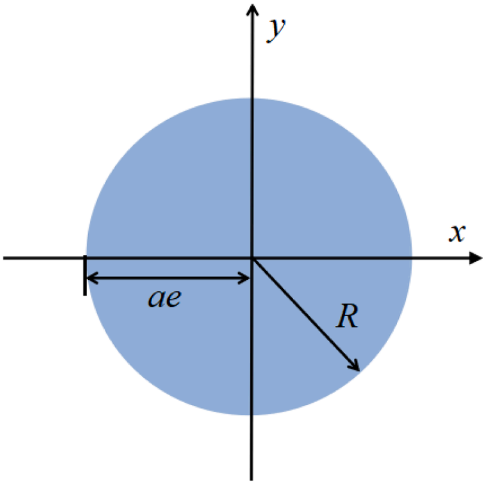
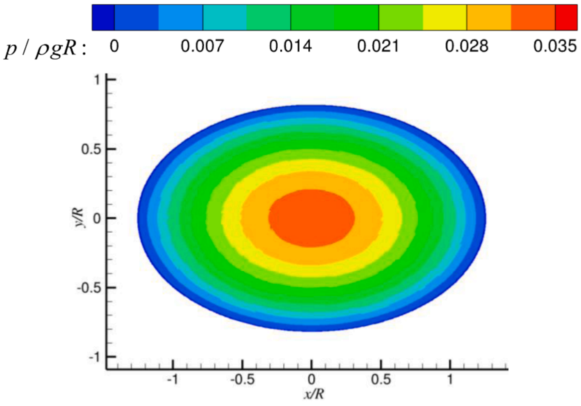
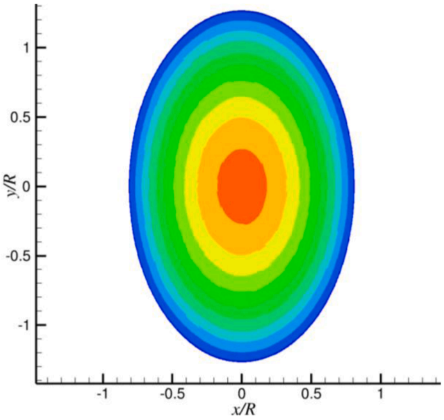


Fig. 5. Illustration of an oscillating droplet under a central force.



(a) $\tau = 40.0$



(b) $\tau = 47.26$

Fig. 6. The particle distributions and pressure contour from ISPH_GNN for the oscillating droplet at two different instants.

strategy involves keeping the network architecture unchanged while fine-tuning the weight parameters with a smaller learning rate. This approach takes the advantage of the knowledge learned by the pre-trained GNN model, which has already captured general patterns in the graph data, and adapts it to a new related task.

In this paper, the network architecture with employing transfer learning, which includes the encoding, message-passing, and

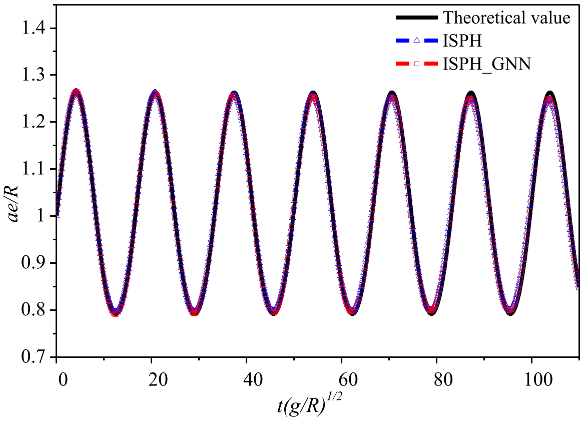


Fig. 7. Time histories of semi-major axis *ae* of the oscillating droplet with the initial particle spacing of 0.01).

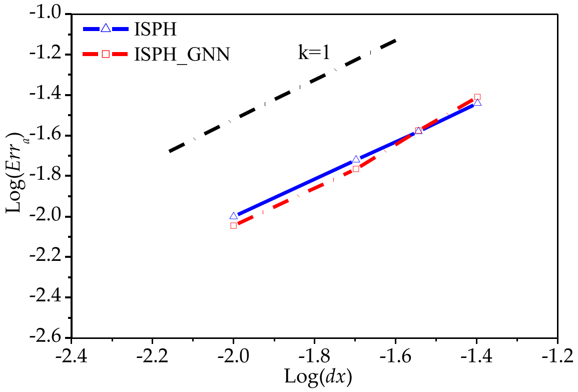


Fig. 8. Numerical errors in modeling an oscillating droplet with different particle spacing (*dx* here just indicate the initial particle spacing).

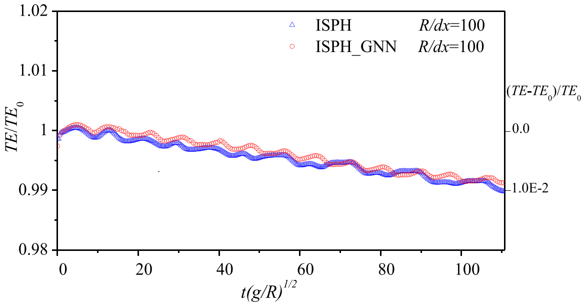


Fig. 9. Time histories of total energy (*TE*) in the case of oscillating droplet.

decoding layers, remains consistent with the architecture of the pre-trained GNN model. This is based on the rationale that these layers have already learned general features that can be transferred across tasks. Transfer learning in neural networks is implemented by reusing the weights from the pre-trained GNN model as a starting point and fine-tuning them with a small amount of target data. The primary goal of this fine-tuning process is to adjust the GNN model's weights to improve its performance on the new task, while preserving the knowledge gained from pre-training. To achieve this, a smaller learning rate is used, which prevents drastic changes to the pre-trained weights while still allowing the model to adapt to the new task. In the fine-tuning process described in this paper, an initial learning rate of 10^{-6} is used for the encoding and message-passing modules, while an initial learning rate of 10^{-5} is applied to the decoding module of the GNN. The flowchart for performing transfer learning based on the pre-trained GNN model is shown in Fig. 4. The function f_{GNN_FT}, which maps inputs to outputs in the fine-tuned GNN model utilizing transfer learning, can be expressed as:

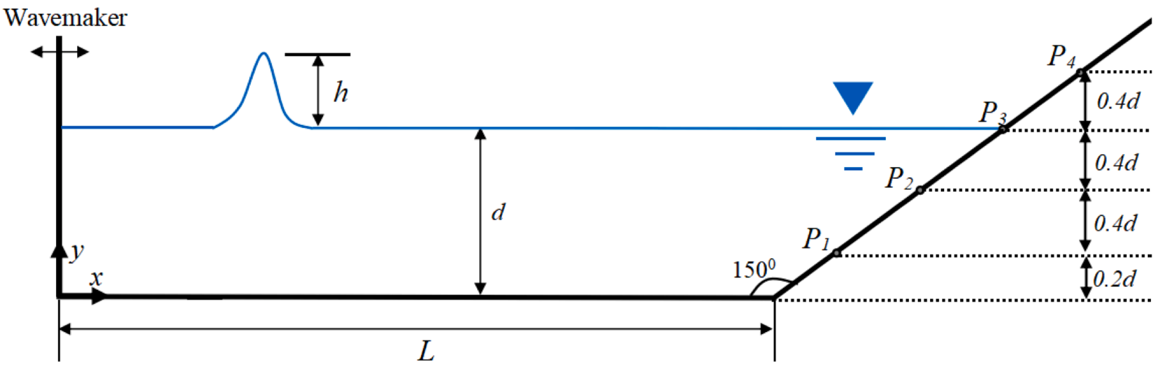
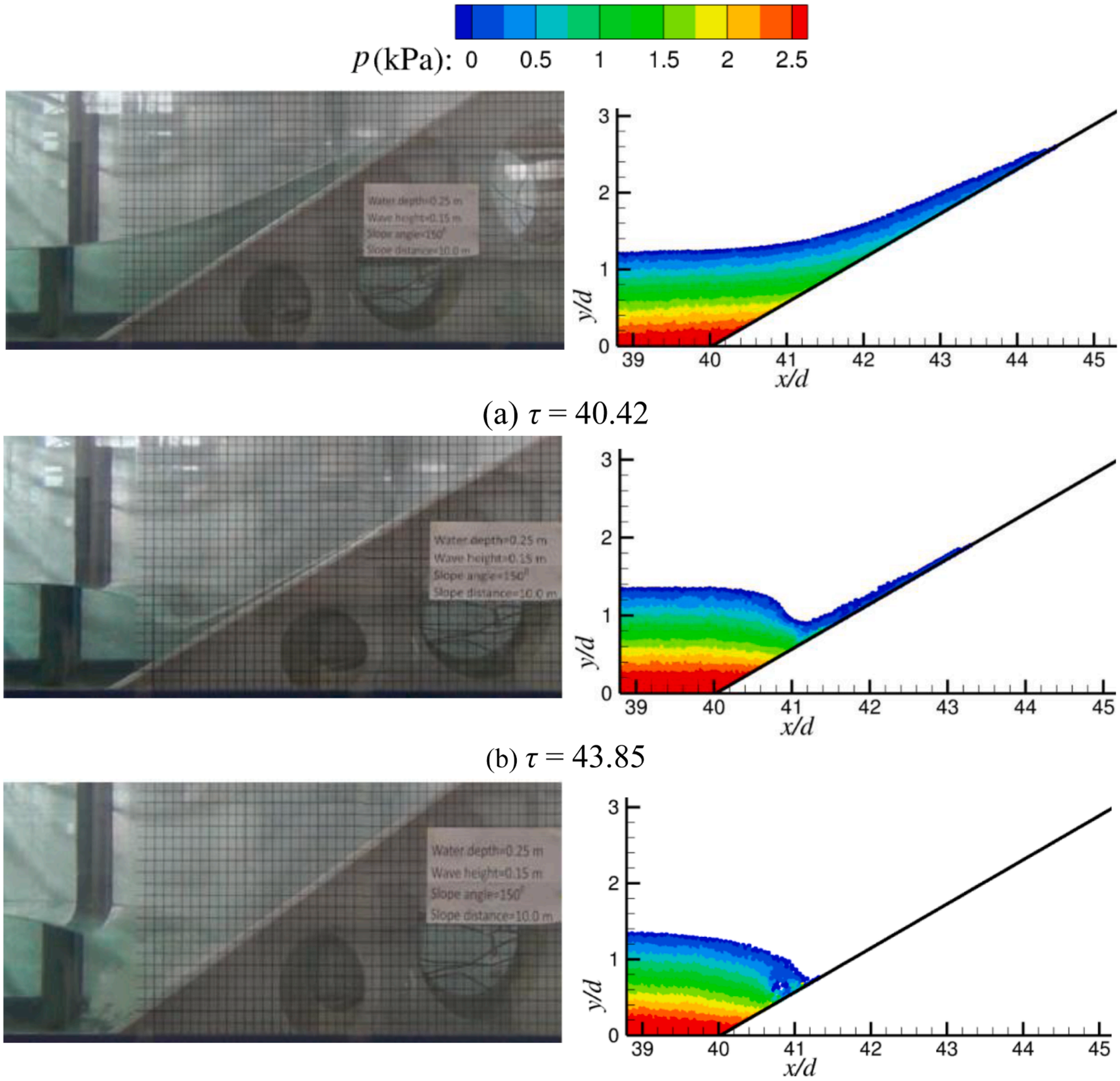


Fig. 10. Schematic wave tank for solitary wave impact on a slope.



(a) $\tau = 40.42$

(b) $\tau = 43.85$

(c) $\tau = 45.1$

Fig. 11. Comparisons of wave uprush and backwash between laboratory wave photograph (left) and ISPH_GNN particle snapshots (right) at different time instants.

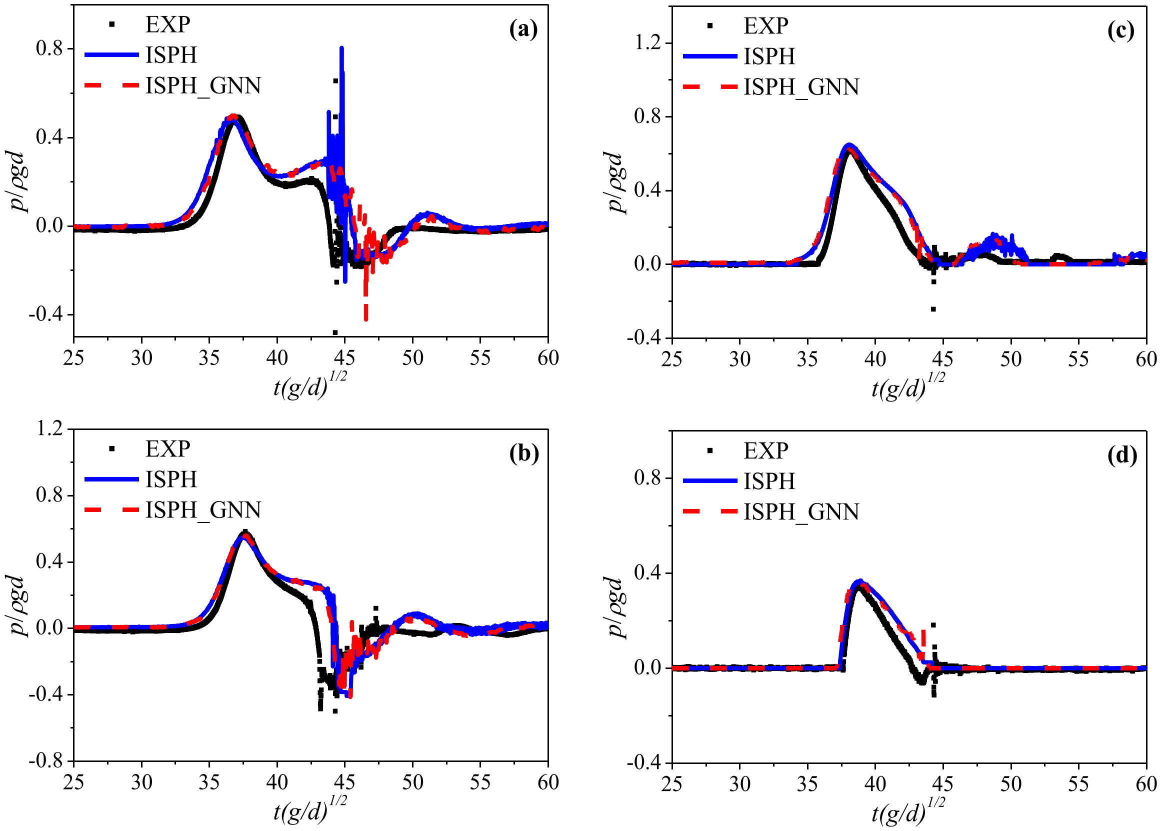


Fig. 12. Comparisons of wave impact pressure between experimental data and different ISPH results at measuring point: (a) P_1; (b) P_2; (c): P_3; and (d) $P_{4.}$

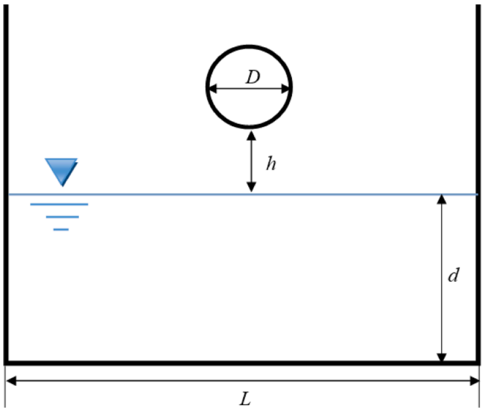


Fig. 13. Sketch of the numerical water tank for water entry of a circular cylinder.

$$\widehat{p}_{d,t+\Delta t} = f_{GNN\_FT}\left(\Psi, \mathbf{u}^{*}, p_{d,t}, c_p^{ft}\right) \tag{29}$$

where c_p^{ft} represents the fine-tuning network coefficients, which are adjusted based on the pre-trained network's weights c_p in Eq. (22).

This is noted that we attempt only to explore the potential of transfer learning technique in improving the robustness of modelling WSI problems. For this purpose, the transfer leaning technique will be tested on some cases, and the corresponding results are compared with those without it to demonstrate its potential in the sections below.

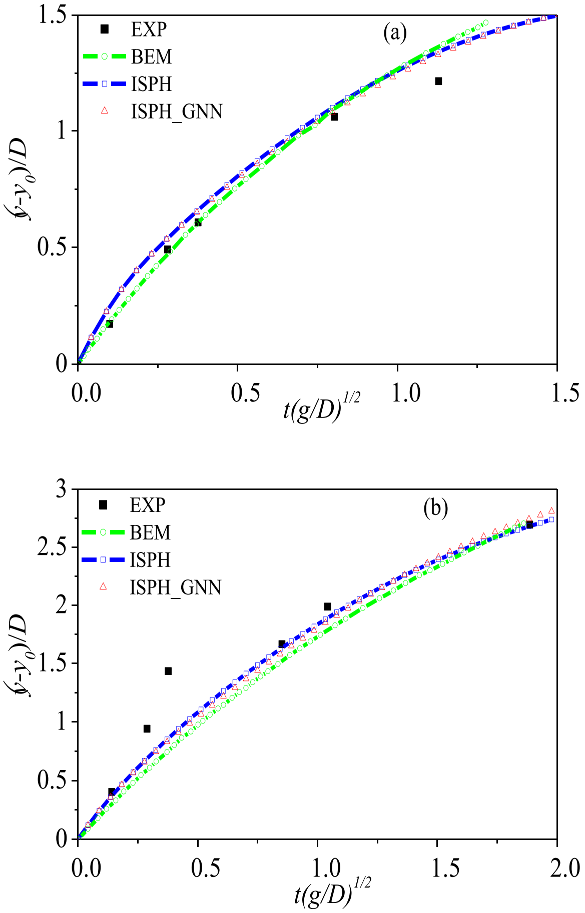


Fig. 14. Comparisons of time histories of the penetration depths: (a) the half buoyant cylinder; and (b) the neutrally buoyant cylinder.

3. Numerical investigations using ISPH_GNN

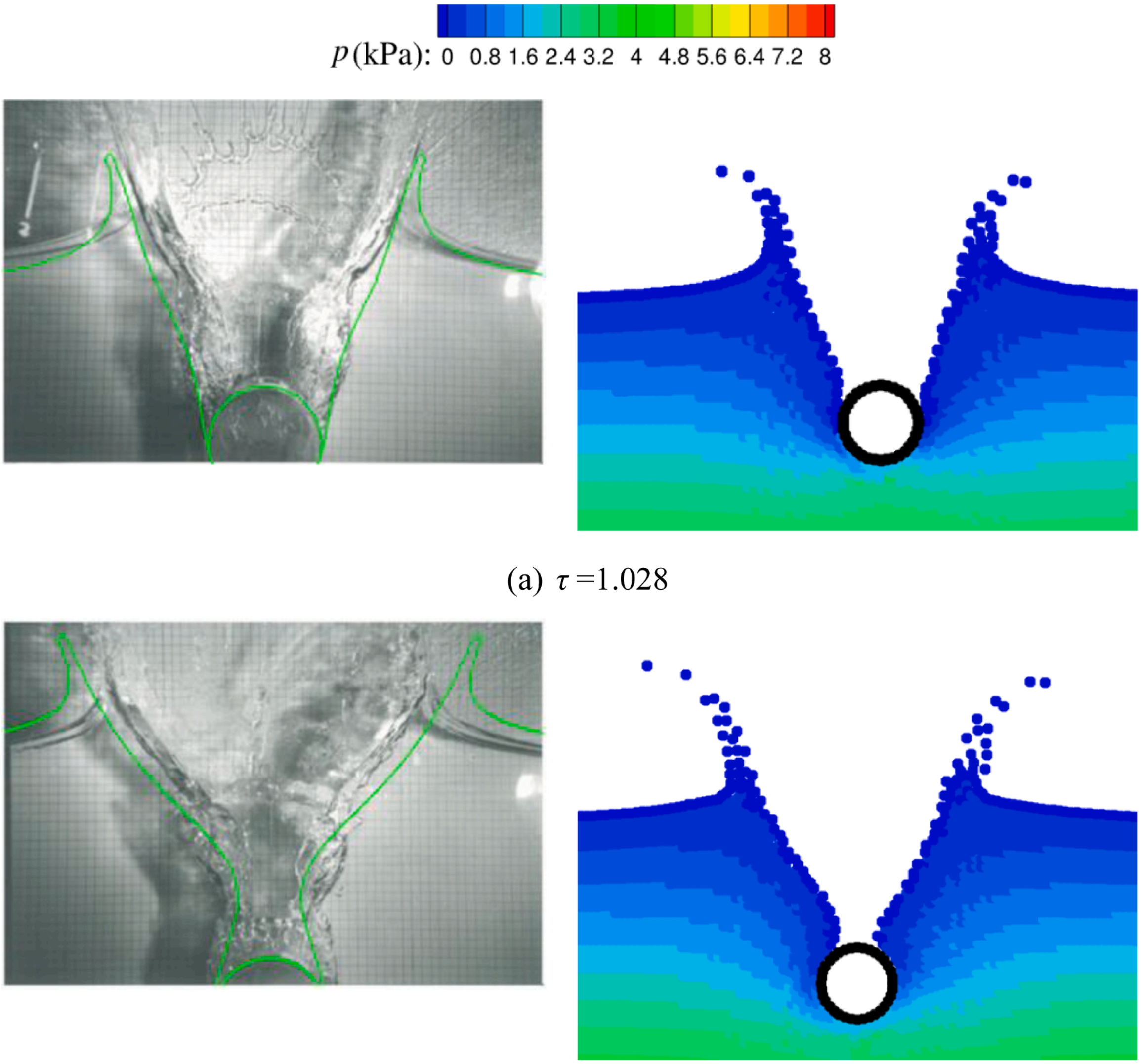
In this part, the performance of the ISPH_GNN, using the GNN model trained on data from dam breaking and sloshing cases, is examined by using two types of cases: (1) these with quite different configurations from training cases (dam breaking and sloshing), such as oscillating drop and solitary wave impact on a slope; (2) these involving a structure in water/wave field such as water entry of a circular cylinder, wave impact on a structure with complex geometries and wave-OWSC interactions. For comparison, results from the conventional ISPH method [23] are also presented.

3.1. Oscillating drop under a central force

In this section, the oscillating droplet subjected to a central force field is considered to show the precision, convergence behaviour, and energy conservation properties of ISPH_GNN. This configuration, very different from training cases, involves a non-viscous and incompressible fluid evolving under a conservative force field characterized by $-\Omega^2\mathbf{r}$, where $\Omega^2 = 1.44$ N/m is used in this study. As depicted in Fig. 5, initial conditions feature a perfectly circular droplet of radius $R = 0.5$ m, with particle position vectors $\mathbf{r}$ (x, y) defined relative to the droplet's centre in Cartesian coordinates. Initial particle velocity field is prescribed as $(\sigma x, -\sigma y)$ with $\sigma = 0.4$ s^{-1}. This configuration requires only free-surface boundary treatment. Its exact theoretical solution [44] will be used for quantitative error assessment. For analytical convenience, length scales are nondimensionalized by R and time by the characteristic scale $\tau = t\sqrt{g/R}$ in this subsection unless explicitly stated otherwise.

Fig. 6 shows the particle configurations and pressure distribution contours of the oscillating droplet at two different instants with the initial particle spacing of 0.01. Depicted in Fig. 7 are the temporal evolutions of the oscillating drop's semi-major axis *ae* obtained using ISPH and ISPH_GNN methods. For reference, the analytical solution from [44] is also included to facilitate direct comparison. Notably, the numerical results from ISPH_GNN demonstrate close agreement with the theoretical prediction, maintaining both the amplitude and phase characteristics over time, although there is a slight underestimation of the peak value of *ae* and some phase shifts.

To quantitatively assess the accuracy and convergence of ISPH_GNN, Fig. 8 shows the numerical errors in the cases with different initial particle spacing, ranging from 0.04 to 0.01, in which the averaged errors Err_a (L-2 norm) between the simulated semi-major axis



(a) τ =1.028

(b) τ =1.878

Fig. 15. Comparisons of the free surface penetration of the neutrally buoyant cylinder between experimental photos [47] (left) and ISPH_GNN simulations (right) at different instants.

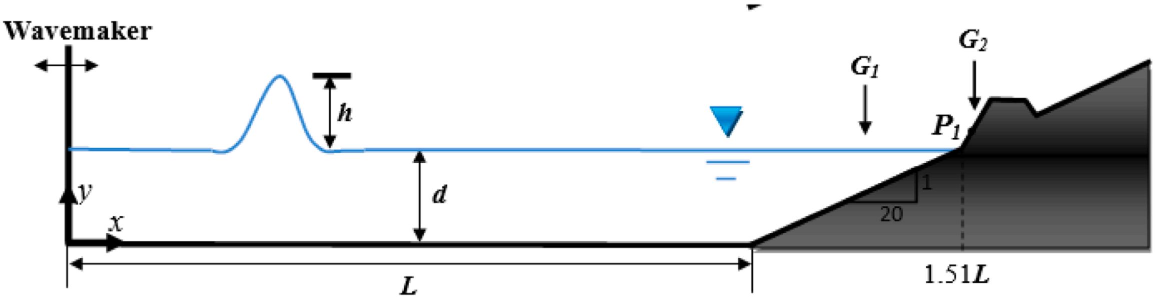


Fig. 16. Schematic setup of solitary wave impact on a trapezoidal seawall.

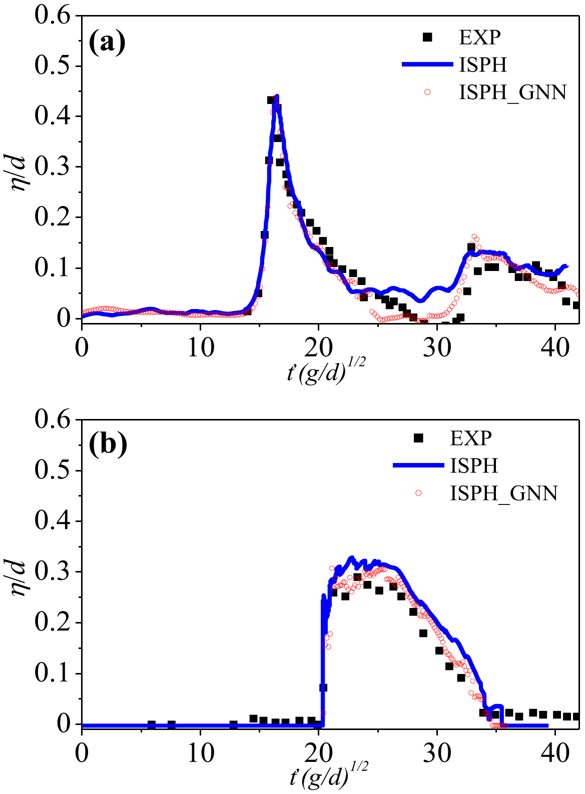


Fig. 17. Time histories of wave profile measured at (a) G_1 ($x = 48.22d$) and (b) G_2 ($x = 53.66d$), with experimental data sourced from Hsiao and Lin [49].

ae and the analytical solution are computed as

$$Err_a = \frac{\sqrt{\sum_{i=1}^{N}\left(ae_{i,n} - ae_{i,a}\right)^2}}{\sqrt{\sum_{i=1}^{N} ae_{i,a}^2}} \tag{30}$$

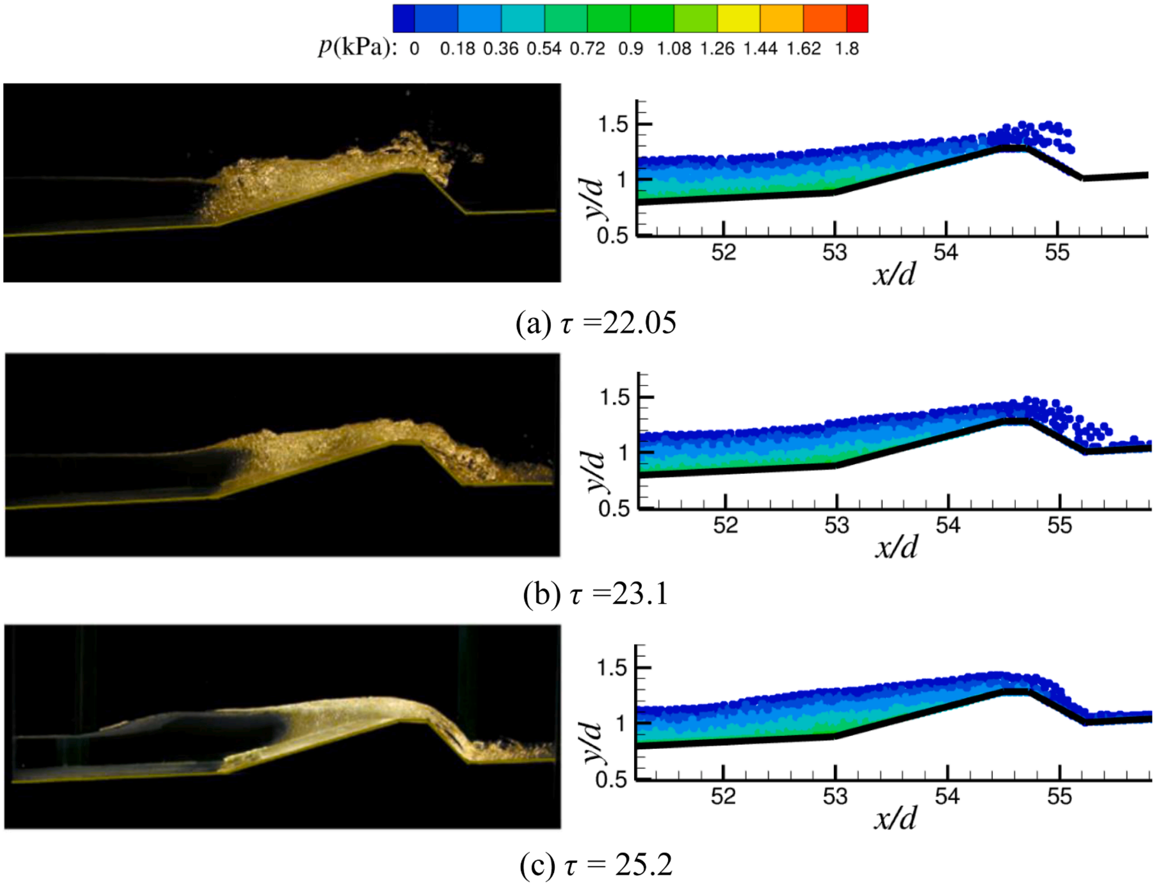
where $ae_{i,a}$ denotes the semi-major axis value from the theoretical solution at the i th time step, and $ae_{i,n}$ represents the corresponding numerical result. Here, N signifies the total number of time steps employed for error assessment. For convenience, the dash lines representing the first-order ($k = 1$) convergence rate is included in the figure. As seen in Fig. 8, the error profile of ISPH_GNN exhibits a linear reduction trend with decreasing particle size, demonstrating a linear convergence behaviour consistent with the conventional ISPH method. It is also seen from Fig. 8 that the ISPH_GNN exhibits smaller errors than the conventional ISPH method at smaller particle spacing.

Energy conservation performance is further evaluated and plotted in Fig. 9, which depicts temporal variations of total energy (*TE*) relative to the theoretical baseline *TE₀*. The error from the theoretical value does not exceed 1 %. Similar to Fig. 8, the error in terms of energy of ISPH_GNN is also slightly smaller compared to the conventional ISPH methods. Notably, the total energy profiles exhibit oscillatory behaviour rather than monotonic decay, consistent with previous studies (e.g., [41]). These fluctuations are attributed to the particle shifting algorithm, which inherently introduces energy perturbations [45].

3.2. Solitary wave impact on a slope

This section employs the ISPH_GNN method to model the impact of a solitary wave on a 150° angled slope, also very different configuration from the training cases. The computational domain replicates the experimental setup of Zheng et al. [46], enabling a direct and meaningful comparison between numerical simulations and experimental results. Fig. 10 illustrates the case with a solitary wave having its amplitude of $h = 0.6d$ and in a tank with a water depth of $d = 0.25$ m. The computational setup specifies a horizontal domain length of $L = 40d$ An initial particle spacing of 0.01 m is employed, resulting in approximately 25,000 particles. For measuring the pressure, four pressure sensors (P_1-P_4) are positioned as illustrated.

Fig. 11 depicts solitary wave uprush and backwash on the slope at sequential instants, simulated using the ISPH_GNN method. At $\tau = 40.42$ ($\tau = t\sqrt{\frac{g}{d}}$), Fig. 11(a) reveals the wave front attains its maximum run up elevation, marking the transition to flow recession.



(a) τ =22.05

(b) τ =23.1

(c) τ = 25.2

Fig. 18. Comparisons of the wave impact and overtopping on the trapezoidal seawall: experimental images (Hsiao and Lin [49], left) and ISPH_GNN simulations (right) at different time instants.

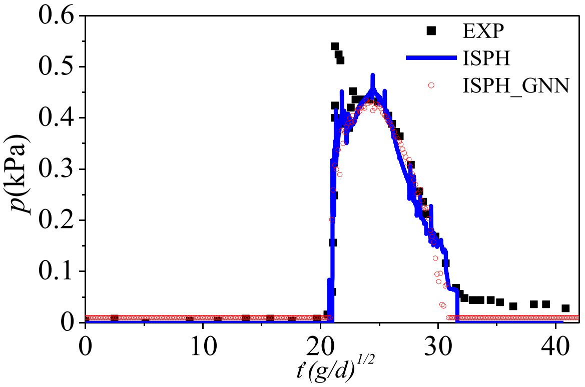


Fig. 19. A comparison of wave impact pressures between different ISPH results and experimental data.

Fig. 11(b) illustrates a vigorous backflow developing near the initial shoreline, a phenomenon that correlates with the sudden pressure drop observed in subsequent analyses (see Fig. 12). Overall, the numerical simulation results exhibit correct correlation with experimental observations.

To quantify the accuracy of ISPH_GNN, Fig. 12(a)-(d) present the gauge pressure at four monitoring locations (P_1-P_4) between experimental measurements and different ISPH methods. Despite some discrepancies, reasonable agreement has been found. Analogous to experimental trends, the impact pressure at submerged sensors P_1 and P_2 exhibit similar temporal evolution patterns. When the wave reaches its maximum run-up height, the impact pressure reaches the largest peak. Subsequently, as the wave recedes, the pressure gradually transitions to the relative negative values until it hits the minimum pressure point. This is understandable from the

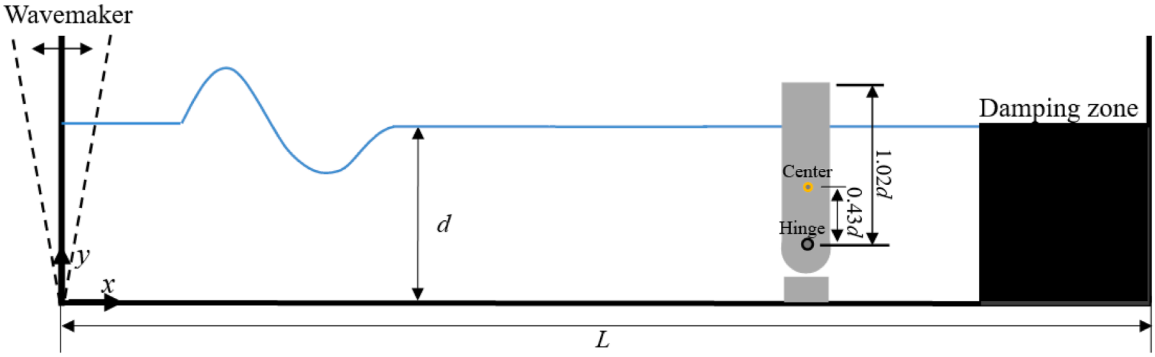


Fig. 20. Sketch of wave interaction with an OWSC.

basic fluid dynamics. As illustrated in Fig. 12(c) and (d), the calculated pressures at sensor points P_3 and P_4 escalate sharply upon solitary wave impact, achieving maximum values before returning to ambient levels without significant oscillating into negative pressure zones.

Given the critical role of peak pressure in marine structure design, we carry out a quantitative error assessment for peak pressure from different ISPH methods. Taking the experimental peak pressure as reference, the relative errors are about 1.1 % for ISPH_GNN and 0.3 % for ISPH in Fig. 12(a), 3.1 % for ISPH_GNN and 5.6 % for ISPH in Fig. 12(b), 3.3 % for ISPH_GNN, and 4.2 % for ISPH in Fig. 12(c), and 6.3 % for ISPH_GNN and 6.8 % for ISPH in Fig. 12(d). Although the specific values of the errors compared to the experimental data are varying, they are all acceptable from engineering point of view. However, it should be indicated that the agreement between numerical and experimental results during the process of wave receding is not that good, though both the numerical ones are not too far from the experimental ones.

3.3. Water entry of a circular cylinder

Next we start to discuss the cases with very different configurations from the train cases. They have a structure in water/wave fields while there is no structure in the training cases. In this sub-section, the ISPH_GNN is first applied for simulating the 2D cylinder water entry problem. This problem has been experimentally studied in [47] and numerically modelled in [48]. For this case, a cylinder with a diameter of $D = 0.11$ m is released from a height of $h = 4.45D$ (measured from the cylinder's centre to the water surface), as show in Fig. 13. Therefore, the initial entry velocity when the cylinder reaches the water surface can be estimated as $\sqrt{2gh} = 2.955$ m /s, where $g = 9.81$ m/s^2. Two cylinders of identical dimensions but with different mass are evaluated: one is half buoyant and the other is neutrally buoyant, as detailed in [47,48].

In this case, the fluid domain has a length of $L = 18.2D$ and a depth of $d = 8.2D$, with the initial particle spacing dx set at 0.01 m, meaning $h/dx = 60$. Fig. 14 compares the penetration depth time histories for the half and neutrally buoyant cases. The comparison includes results from different ISPH methods, BEM solutions [48] and experimental data [47]. It can be observed that the penetration depths obtained from the ISPH_GNN method are similar to those from ISPH, BEM simulations, and experimental data, though there are some visible difference from experimental data. Using the ISPH result from Fig. 14 as a reference, the penetration depth error for ISPH_GNN is approximately 1.2 % in Fig. 14(a). In Fig. 14(b), the error is around 2.0 %. This error is determined using the following formula:

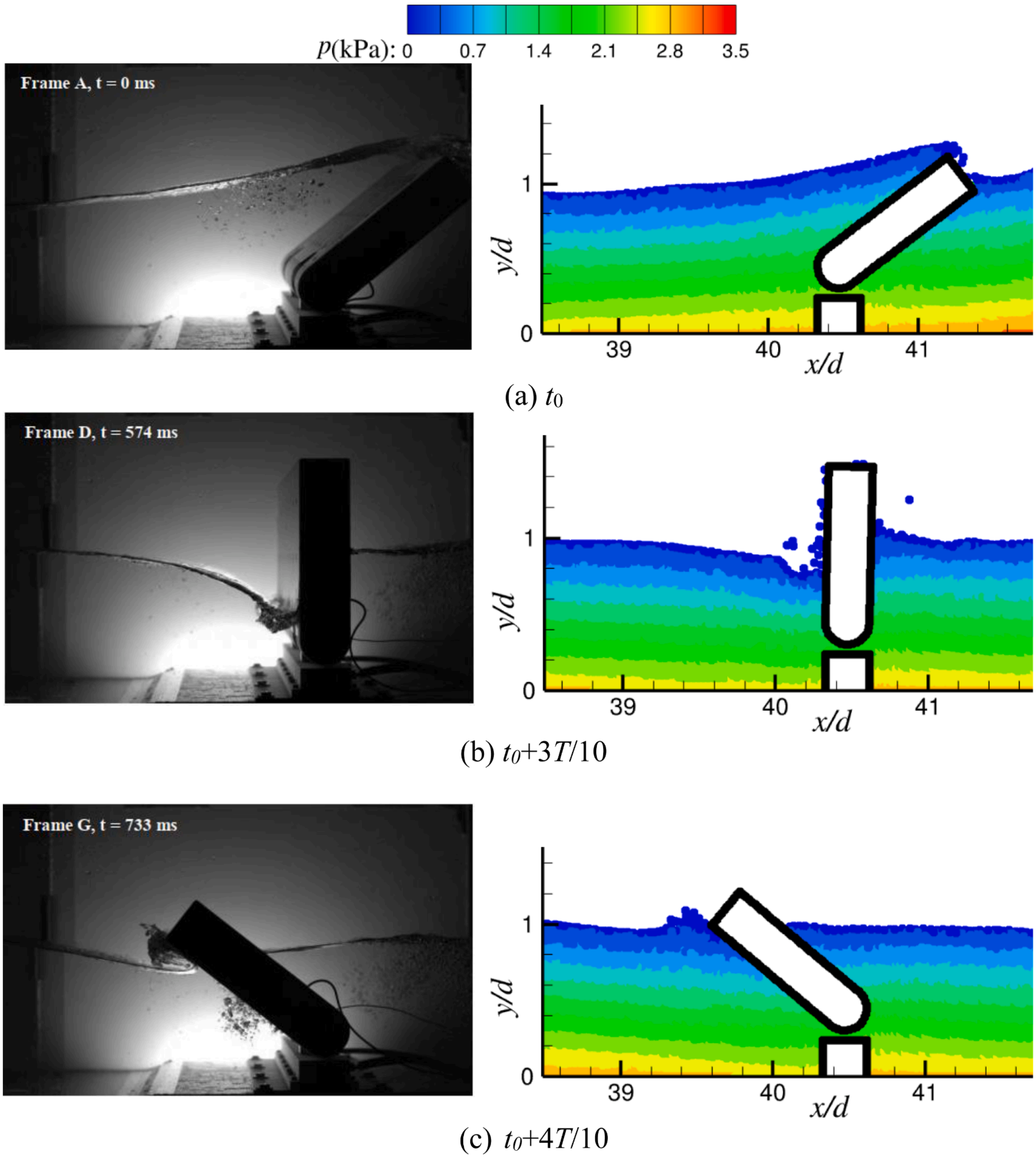
$$Err_a = \frac{\sqrt{\sum_{i=1}^{N}\left(\eta_{i,n} - \eta_{i,f}\right)^2}}{\sqrt{\sum_{i=1}^{N}{\eta_{i,f}}^2}} \tag{31}$$

where $\eta_{i,n}$ represents the penetration depth at the i th time step obtained from the ISPH_GNN, while $\eta_{i,f}$ refers to the corresponding ISPH result. N is the total number of time steps ranging from τ =0.0 to 2 ($\tau = t\sqrt{\frac{g}{D}}$).

In Fig. 15, snapshots from the experimental images and ISPH_GNN simulations are compared for the neutrally buoyant case at two time instants, specifically τ =1.028 and τ =1.878. The free-surface profile evolution and the water jet formation generated by ISPH_GNN clearly resemble the BEM results [48] and experimental images [47]. The ISPH_GNN method also presents reasonable well-structured pressure field.

3.4. Solitary wave impact on a trapezoidal seawall

Another scenario analysed is the impact of a solitary wave on a trapezoidal seawall, which was experimentally studied by Hsiao and Lin [49]. This case is different from that in Section 3.2, mainly with the seawall appearing here. The relevant parameters are illustrated in Fig. 16 in which the wave gauge "G" and pressure probe "P" are also depicted. To be consistent with [49], the solitary wave height $h =$



(a) t_0

(b) $t_0+3T/10$

(c) $t_0+4T/10$

Fig. 21. Comparison of OWSC motions between experimental images [51] (left) and ISPH_GNN results (right) at various time instants.

$0.35d$, $L = 35d$ and the water depth d is 0.2 m. In evaluating the numerical results, relative time $t' = t - t_{MR}$ is used, where t_{MR} marks the point when the wave run-up reaches its peak at the wall first time. The numerical simulations capture 10 s of wave propagation and its interaction with the seawall. The solitary wave is generated by the wavemaker using the approach in Ma and Zhou [50]. In these simulations, the initial particle spacing is set to 0.01 m.

Fig. 17 presents a comparison of the wave profile time histories between the numerical results and the experimental data [49] recorded at G_1 and G_2. As seen, both the numerical methods produce reasonably accurate results. According to the experimental data in Fig. 17, the wave profile errors at G_1 are approximately 16.6 % for ISPH_GNN and 20.1 % for ISPH, as shown in Fig. 17(a). For the wave profile recorded at G_2, the errors are around 13.2 % for ISPH_GNN and 18.7 % for ISPH, as illustrated in Fig. 17(b). The error calculation follows the same method as in Eq. (31) but $\eta_{i,n}$ and $\eta_{i,f}$ are substituted by the wave elevations at i th time step obtained from the numerical and the experimental results for this case. However, it is worth noting that the experimental data were extracted from the publication, which may limit their precision as a reference when assessing quantitative error.

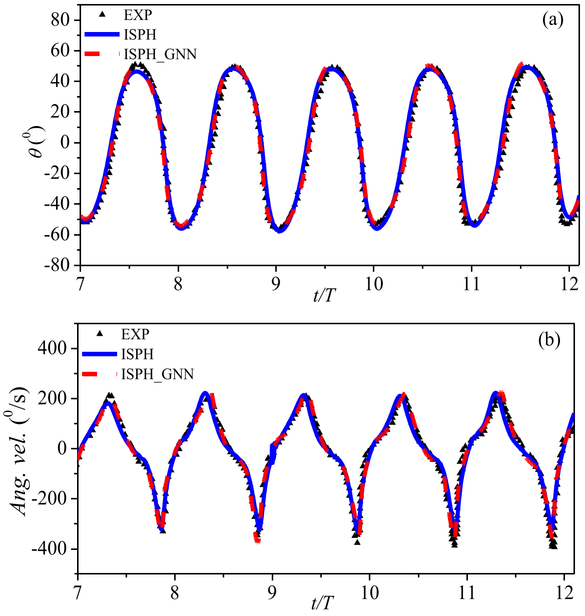


Fig. 22. Comparison of the time histories of OWSC motions: (a) rotation angle and (b) angular velocity of the flap.

Fig. 18 illustrates particle distributions and corresponding pressure field contours at three instants, $\tau = 22.05$, 23.1, and 25.2 ($\tau = t\cdot\sqrt{\frac{g}{d}}$), capturing the wave impact and overtopping process of the trapezoidal structure. As shown in the experimental images, when the wave overtops the trapezoidal seawall, an overtopping tongue forms at the crown and then falls onto the slope behind it. It is evident that the ISPH_GNN method satisfactorily simulates these physical phenomena similar to that in experimental photos and demonstrates a smooth and well-aligned pressure field.

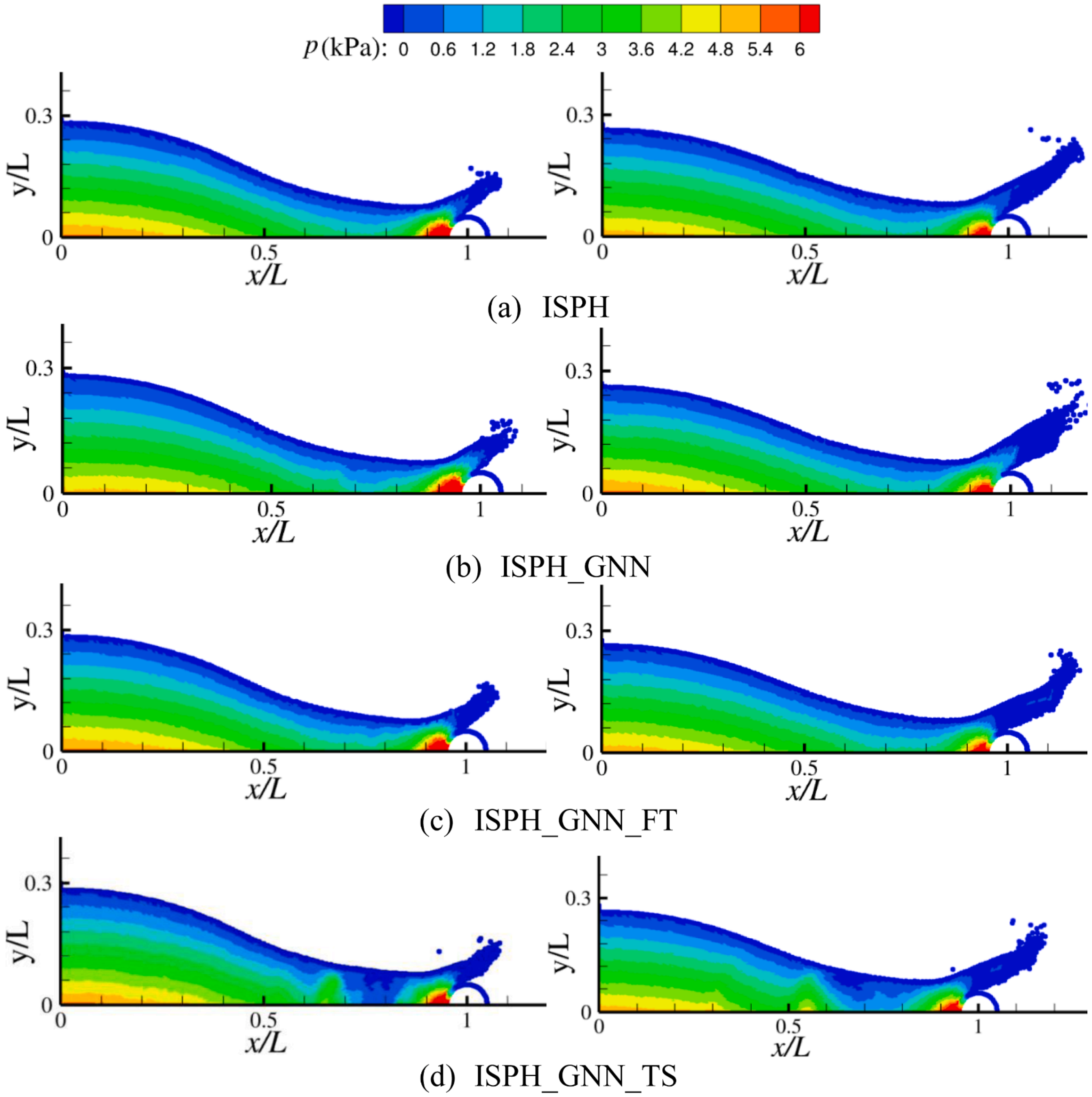
Additionally, Fig. 19 presents a comparison of impact pressures calculated using various ISPH schemes, as measured by pressure sensor P_1 located at coordinate (10.83, 0.249) on the trapezoidal seawall. The results indicate that all ISPH schemes exhibit similar overall trends in impact pressure, closely aligning with the experimental data. However, some differences are observed, which can be attributed to the complexity of the physical problem.

3.5. Regular wave interaction with an oscillating wave surge converter

In this section, the ISPH_GNN is utilized to examine the regular wave interaction with an oscillating wave surge converter (OWSC), which has very different configuration from the training cases, in particular with a moving structures in waves. This was experimentally studied by Henry [51] and Wei et al. [52]. A schematic of the domain is shown in Fig. 20, in which the water depth is $d = 0.305$ m. The OWSC model used in the experiment is a 1:40 scale version with simplified geometry, as illustrated in Fig. 20. It consists of an OWSC measuring $1.02d$ in height and 0.0875 m in thickness, with a semi-circular tube underneath. The OWSC pivots freely around a tubular axle, acting as a hinge. The hinge is positioned $0.44d$ m above the flume floor. The OWSC weighs 4.27 kg, with a pitch rotational inertia of 0.1147 kg·m^2 around the hinge, and its centre of mass is located $0.43d$ above the hinge.

Fig. 21 presents a sequence of snapshots comparing the motion of OWSC from the ISPH_GNN simulations, using an initial particle spacing of $dx = 0.008$ m (i.e., $d/dx = 38$), with experimental images at various time instants. In this case, the wave height is $H = 0.328d$ and the wave period is $T = 1.9$ s, corresponding to the case with the strongest impact. In Fig. 21(a), the OWSC is rotated maximally clockwise, with zero angular velocity. Some water flows over its top edge. After it reaches the maximum clockwise position as shown in Fig. 21(a), it starts to rotate counterclockwise. The fluid particles in its upper left corner are pushed counterclockwise, and begin to fall back counterclockwise. When it returns to its vertical position at $t_0 + 3T/10$ as shown in Fig. 21(b), the fluid particles close to it exhibit a state of falling down along its surface. Then, it continues to rotate counterclockwise and arrives near the maximum rotation counterclockwise as shown in Fig. 21(c). It is evident that the ISPH_GNN results largely matching those seen in experimental images.

It should be noted, however, that there are discrepancies between free surfaces near the structure in the ISPH_GNN simulation results and those in the experimental images, in particular in Fig. 21(c). There may be different reasons. For example, in the experiments, there may be some mechanical viscosity between rotating and fixed parts, which cannot be modelled in numerical simulation as lack of relevant information. In addition, although the experiment was considered as a two-dimensional test, narrow gaps existed between both sides of the structure and the walls of the wave flume, leading to a small amount of water flow though these gaps. This is again not consistent with the two-dimensional numerical simulations. Furthermore, we are not sure how $t_0 = 0$ in experimental image corresponds to the physical time. We just assume the time is when the convertor reaches to the angle in Fig.21(a) in the first time, not other wave cycles. Apart from these, the bottom configuration cannot be modelled due to lack of the information in our hand. Of



(a) ISPH

(b) ISPH_GNN

(c) ISPH_GNN_FT

(d) ISPH_GNN_TS

Fig. 23. Comparison of particle distributions and pressure contours at $\widetilde{t}$= 1.576 (left column) and $\widetilde{t}$ = 1.75 (right column) obtained using different ISPH_GNN models for the case of dam breaking over a semicircle with h = 0.4L , l = 0.4L and L = 2.0 m.

course, numerical error may also contribute to discrepancies, which indicate the more investigations are needed.

Fig. 22 compares the time histories of the rotational angle and the angular velocity of the OWSC from ISPH_GNN simulations and experimental data [51]. Overall, the numerical results including the amplitude of the rotation angle and angular velocity show quite good agreement with the experimental data, despite some visible differences. The rotation angle differences from the experimental data are approximately 14.3 % for ISPH_GNN and 16.7 % for ISPH for the results in Fig. 22(a), which are estimated in the same way as Eq. (31) with the variables changed to either rotation rotational angle and the angular velocity. Meanwhile, the angular velocity differences are around 17.8 % for ISPH_GNN and 19.3 % for ISPH for the results in Fig. 22(b). It should be mentioned that the difference in the peaks of numerical results from those of experimental much smaller, generally not >10 %.

Overall, the above investigations show that the ISPH_GNN with a single GNN model trained on the data from some dam breaking and sloshing scenarios can be applied to simulate a range of cases with different configurations from the training cases, including the violent WSI problems, and yield satisfactory results. While more extensive studies are required, these findings suggest promising potential of ISPH_GNN for broad applications in marine engineering. This potential stems from the fact that the data for training the GNN are generated from relatively simple cases like dam breaking and sloshing scenarios, not from the cases with similar configurations to the targeting cases for which generating data can be very time consuming.

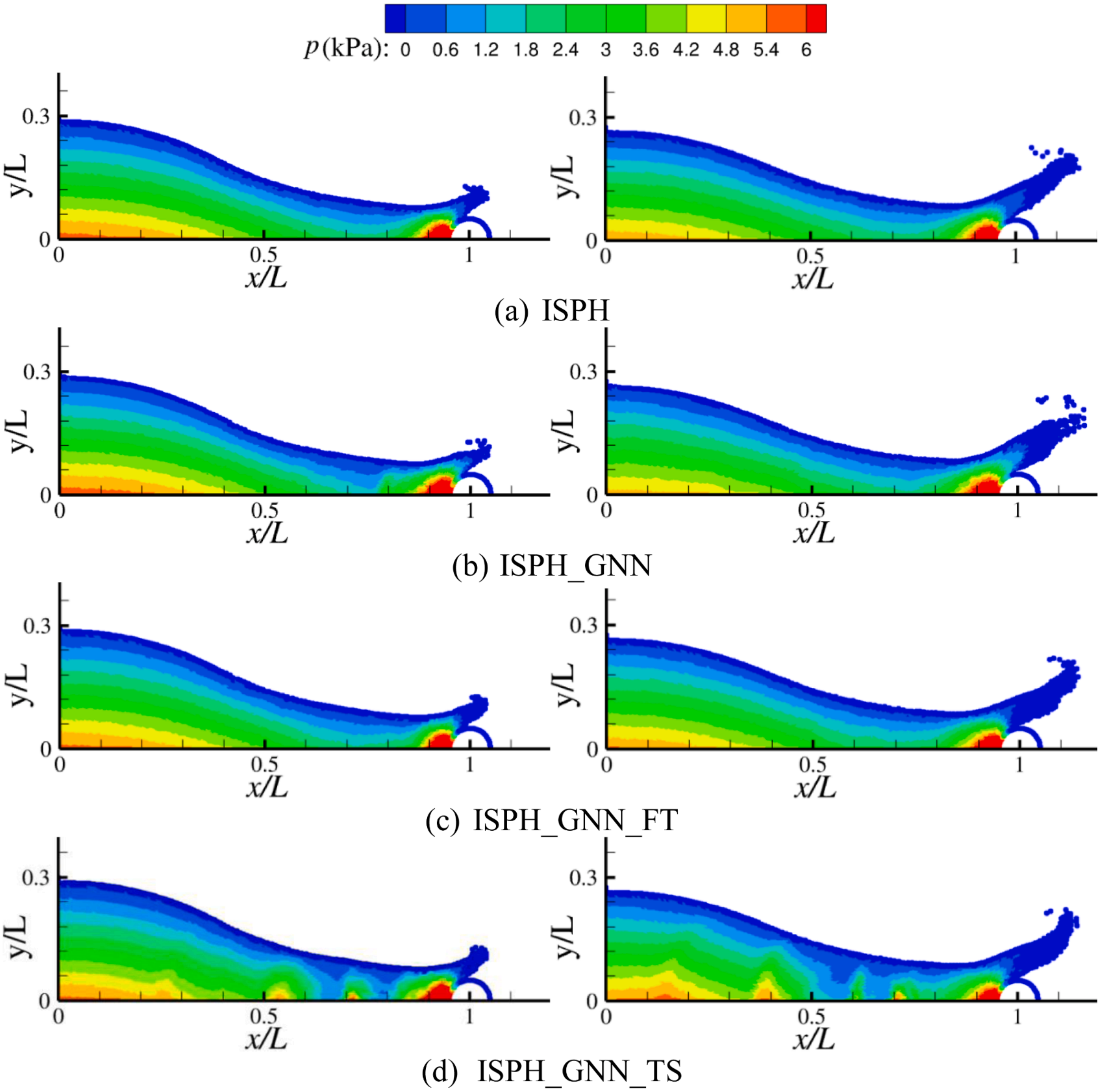


Fig. 24. Comparison of particle distributions and pressure contours at $\tilde{t}$= 1.486 (left column) and $\tilde{t}$ = 1.65 (right column) obtained from different ISPH_GNN models for the case of dam breaking over a semicircle with h = 0.45L , l = 0.35L and L = 2.0 m.

4. Demonstration of the performance of ISPH_GNN with transfer learning

To investigate the performance of the ISPH_GNN model with transfer learning, it is applied to studying some cases in this section. For this purpose, the pre-trained GNN model trained using the cases of dam breaking and sloshing will be fine-tuned through transfer learning using the data for dam breaking over a semicircle structure with a diameter of 0.1L . To generate both the training and testing datasets, the lengths and heights of the water column are randomly selected from the range of h/L = 0.2~0.5 and l/L = 0.2~0.5, while the distance from the centre of semicircle to the left wall is fixed at L = 2.0 m. Total 15 cases are simulated. The GNN is fine-tuned using the procedure given in Fig. 4 with 10 cases for training and 5 cases for testing.

The fine-tuned GNN model is then tested on two cases with dimensions h/L = 0.4, l/L = 0.4 and L = 2.0 m, as well as h/L = 0.45L , l/L = 0.35 and L = 2.0 m, neither of which coincide with any cases during the fine-tuning process. For comparison, Figs. 23 and 24 present the spatial pressure distributions generated by three models: ISPH_GNN, ISPH_GNN_FT (ISPH_GNN with the fine-tuned model), and ISPH_GNN_TS (ISPH_GNN with the model re-trained from scratch using the same small training dataset). The results are shown at two key moments, corresponding to instances of strong wave impacts on the semicircular structure.

As can be seen from Fig. 23 and Fig. 24, the numerical results from ISPH_GNN_TS exhibit obvious unreasonable pressure distribution in front of the semicircle in both cases. In contrast, the pressure distribution from ISPH_GNN_FT is much smoother and closely matches the results from the ISPH method. Fig. 25 compares the horizontal impact force exerted on the semicircle by fluid from the

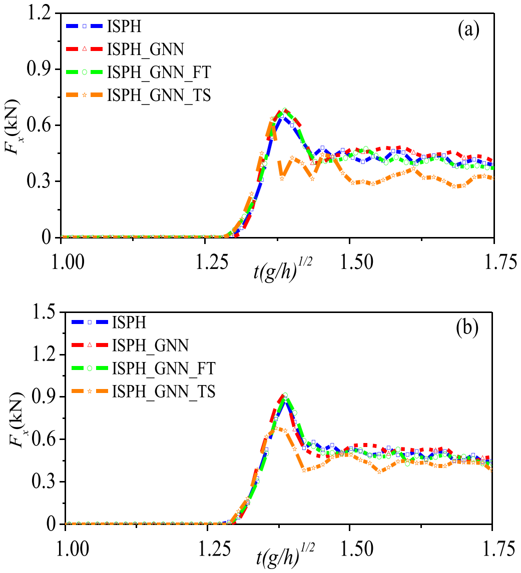


Fig. 25. Comparison of the horizontal impact force exerted on the semicircle obtained from different ISPH models: (a) $h = 0.4L$, $l = 0.4L$ and $L =$ 2.0 m, (b) $h = 0.45L$, $l = 0.35L$ and $L = 2.0$ m.

different models. Using the ISPH results from Fig. 25 as a reference, the horizontal impact force errors in Fig. 25(a) are approximately 10.9 % for ISPH_GNN, 8.5 % for ISPH_GNN_FT, and 28.6 % for ISPH_GNN_TS. Similarly, in Fig. 25(b), the horizontal impact force errors are around 11.5 % for ISPH_GNN, 8.8 % for ISPH_GNN_FT, and 20.7 % for ISPH_GNN_TS.

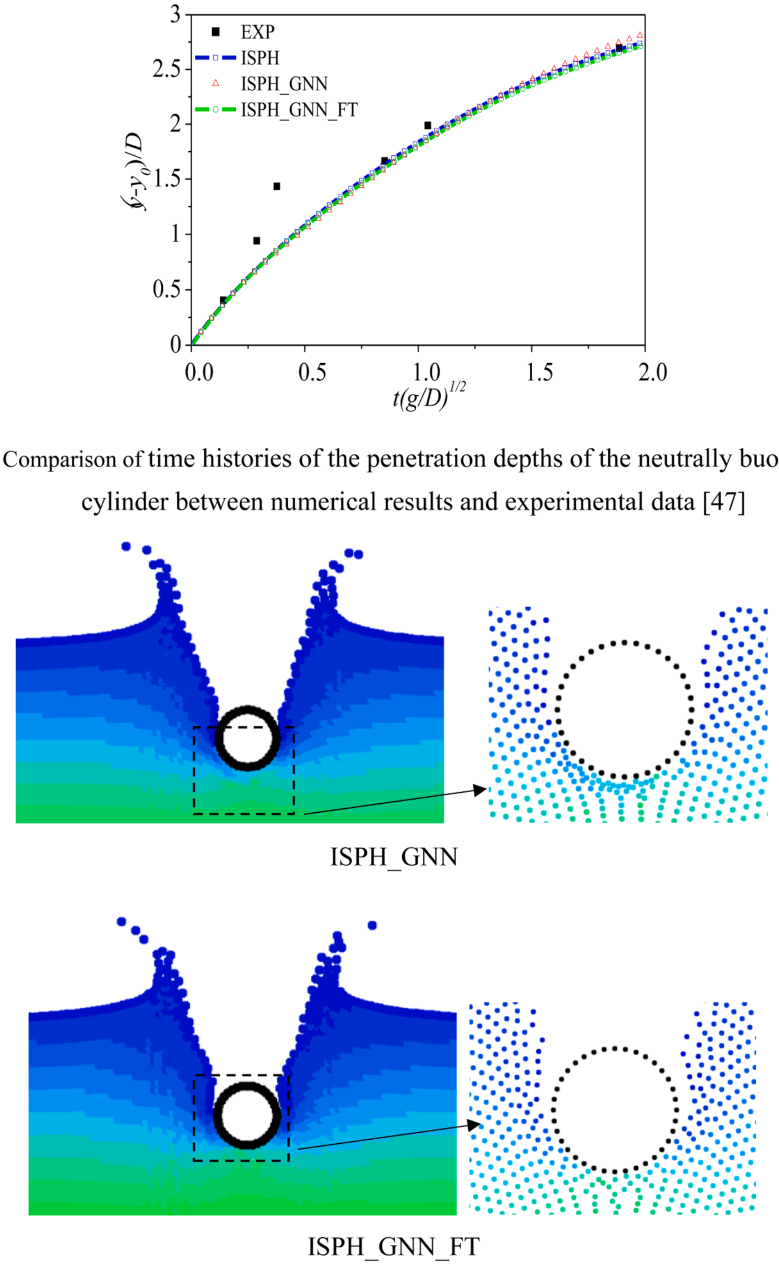
The above comparisons demonstrate that the ISPH_GNN model with the fine-tuned GNN performs much better than ISPH_GNN with the model re-trained using small amount of data from scratch, and also slightly improve the results the pre-trained ISPH_GNN. This suggests that applying transfer learning by fine-tuning the pre-trained GNN model with a small amount of data in the ISPH_GNN approach can be useful for new targeted WSI tasks at least for cases considered here.

To further investigate the performance of the fine-tuned GNN model, the ISPH_GNN_FT, the same as that used for Fig. 24–25, was also applied to study the previously discussed cases: the water entry of a circular cylinder (Section 3.3), the solitary wave impact on a trapezoidal seawall (Section 3.4), and the regular wave interaction with an OSWC (Section 3.5). The setups are kept the same as in previous sections. Fig. 26(a) presents a comparison of the numerical results for the water entry of a neutrally buoyant circular cylinder. Relative to the ISPH results in Fig. 26(a), the errors in penetration depth are approximately 2.0 % for ISPH_GNN and 1.8 % for ISPH_GNN_FT. Additionally, Fig. 26(b) shows that the particle distribution beneath the cylinder from the ISPH_GNN and ISPH_GNN_FT results. It is evident that the ISPH_GNN_FT model produces a bit more uniform particle distribution beneath the cylinder compared to the ISPH_GNN mode.

Fig. 27 gives a comparison of the wave profiles recorded at G_1 for the solitary wave impact on the trapezoidal seawall. Based on the experimental data in Fig. 27, the wave profile errors are about 16.6 % for ISPH_GNN and 15.9 % for ISPH_GNN_FT. The later is slightly less than the former.

Fig. 28 compares the OSWC rotational and angular velocity, using an initial particle spacing of $d/dx = 38$, same as that in Section 3.5. Based on experimental data and estimated using Eq. (26) with the corresponding variables here, the rotational angle errors are about 14.3 % for ISPH_GNN and 14.5 % for ISPH_GNN_FT, as shown in Fig. 28(a). The angular velocity errors are approximately 17.8 % for ISPH_GNN and 17.7 % for ISPH_GNN_FT, as depicted in Fig. 28(b). Additionally, Fig. 29 provides a comparison of the OWSC rotational and angular velocity between ISPH_GNN and ISPH_GNN_FT using a finer particle spacing of $d/dx = 19$, in which the agreements between results are largely similar to that in the above figure. Fig. 30 presents the error trends for rotational angle and angular velocity for both ISPH_GNN and ISPH_GNN_FT across a range of particle spacings, from $d/dx = 38$ to $d/dx = 15$, as part of a convergence study for the case. It is evident from Fig. 30 that ISPH_GNN_FT exhibits a convergence rate similar to ISPH_GNN, but it achieves slightly higher accuracy at coarser particle resolutions before reaching the finest particle spacing.

The above results suggest that ISPH_GNN utilizing a fine-tuned GNN model developed through transfer learning achieves a bit better results but not very significant in the cases studied here. However, it significantly overperforms the model what is trained using a small amount of data. This just highlights that integrating transfer learning with ISPH_GNN can be a way to improve the overall prediction accuracy of ISPH_GNN for violent WSI problems, and also a way to avoid the requirement of large amount of time-consuming data generation for new tasks. However, more extensive studies are need about the performance of transfer learning and how to use it.



(a) Comparison of time histories of the penetration depths of the neutrally buoyant cylinder between numerical results and experimental data [47]

(b) Free surface penetration of the neutrally buoyant cylinder at τ =1.028

Fig. 26. The comparison of numerical results for the water entry of a neutrally buoyant cylinder.

5. Computational efficiency

The computational efficiency is also a crucial factor in evaluating the performance of the ISPH_GNN and ISPH_GNN_FT models for simulating violent WSIs. To assess computational efficiency, the speedup ratio, i.e., the averaged time per step required to solve the PPE in the conventional ISPH to the time per step taken by the GNN model for pressure prediction, is evaluated. The analysis uses the case for regular wave interactions with an OSWC under the same wave conditions in Section 3.5 but with different setups as detailed in Table 1. Simulations were conducted on a workstation equipped with an Intel Xeon Platinum 8268 CPU (2.9 GHz), 256 GB of RAM, and an NVIDIA GeForce RTX 3090 GPU with 24 GB of RAM.

The speedup ratio is presented in Fig. 31. The results clearly demonstrate that the GNN model significantly accelerates pressure prediction compared with the conventional ISPH method. As the number of particles increases, the speedup ratio grows substantially. For example, with 1.4 million particles, the ISPH_GNN achieves a speedup of approximately 136 times over the conventional ISPH for

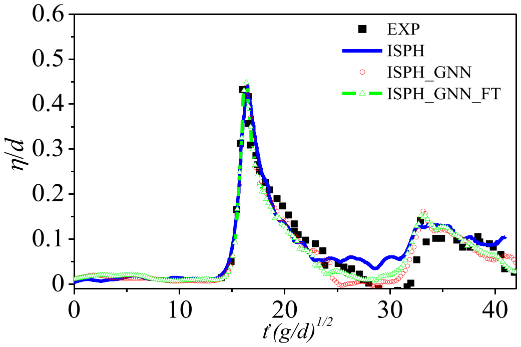


Fig. 27. The comparison of wave profile time history recorded at G_1 in the case of the solitary wave impact on a trapezoidal seawall between numerical results and experimental data [49].

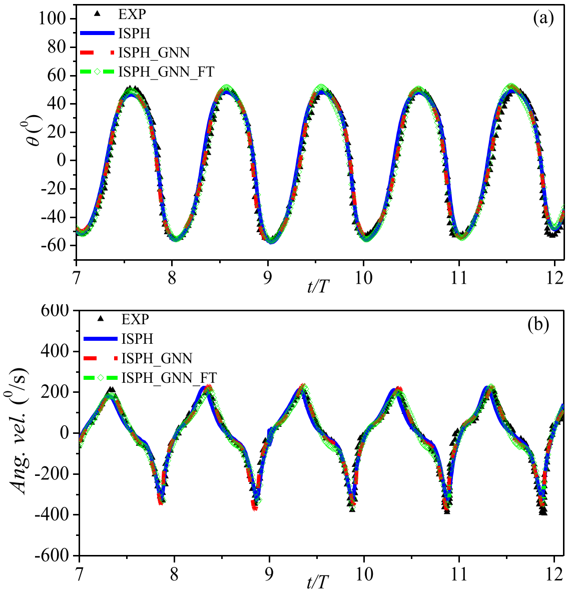


Fig. 28. Comparison of the time histories of flap motions between numerical results and experimental data [51]: (a) rotation angle, (b) flap angular velocity.

the pressure prediction. The speedup achieved in this study is significantly higher than that reported in our previous work [31], where the application of GNN to wave-floater cases resulted in a speedup of 97 times for 1.3 million particles.

This highlights the exceptional efficiency of the ISPH_GNN approach, particularly for large-scale violent WSIs simulations involving a large number of particles. It is also worth noting that the speedup ratio for pressure prediction in both the ISPH_GNN and ISPH_GNN_FT models is essentially identical, as shown in Fig. 31.

6. Conclusion

This paper studies on utilizing a hybrid ISPH_GNN (a kind of hybrid PE-ML method) to simulate violent wave-structure interactions, particularly on training the GNN using data for simple cases and then applying it to more complex WSI problems. Specifically, the GNN integrated with ISPH is trained by using data from sloshing and dam-breaking cases without any structure in wave fields, and then is employed to simulate challenging violent WSIs cases, such as water entry of an object, wave impact on the trapezoidal seawall and wave interaction with OWSC. In most cases, the numerical results have been validated by comparison with experimental data, showing that ISPH_GNN can generate results with satisfactory accuracy. One of the significances of this paper is demonstrating that the ISPH_GNN with the GNN trained solely on wave-only cases (i.e., no any structure inside the wave field) can effectively simulate more complex violent WSIs problems. This means that the data for training the GNN model can be generated on simple cases which may be only a fraction of computational times required for modelling the targeted WSI cases.

Another significance of this paper is to investigate the performance of the ISPH_GNN when integrated with transfer learning. The

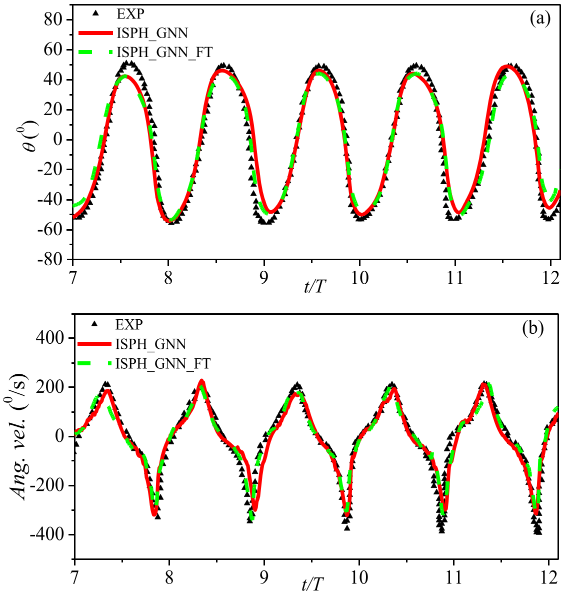


Fig. 29. Comparison of the time histories of OWSC motions between experimental data [51] and numerical results with an initial particle spacing of $d/dx = 19$: (a) rotation angle, (b) flap angular velocity.

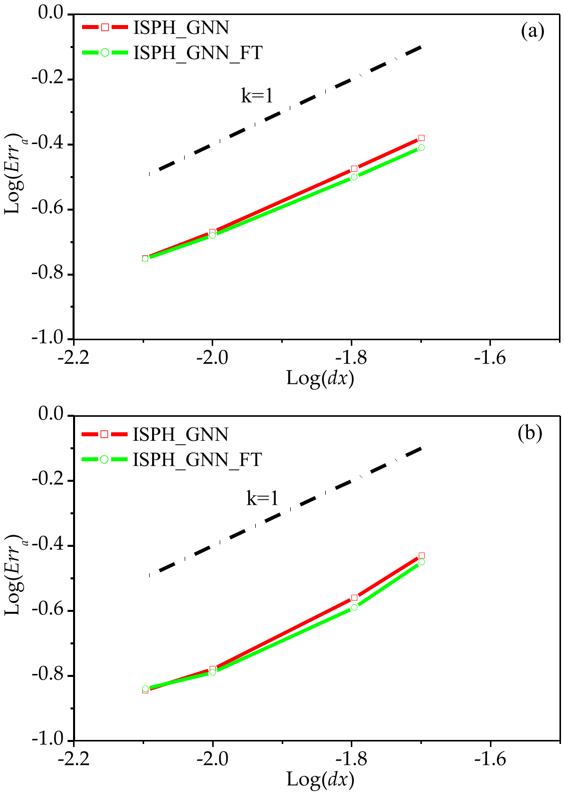


Fig. 30. Errors of numerical results in the wave-OWSC interaction case with different particle resolutions: (a) errors of rotation angle, (b) errors of flap angular velocity.

Table 1
Configurations for the regular wave interaction with an OSWC cases.

Case	*D* (m)	*L* (m)	*dx* (m)	*N*
1	0.305	18	0.008	85,000
2	0.305	40	0.008	189,500
3	0.305	100	0.008	474,500
4	0.6	100	0.008	936,500
5	0.6	150	0.008	1405,000

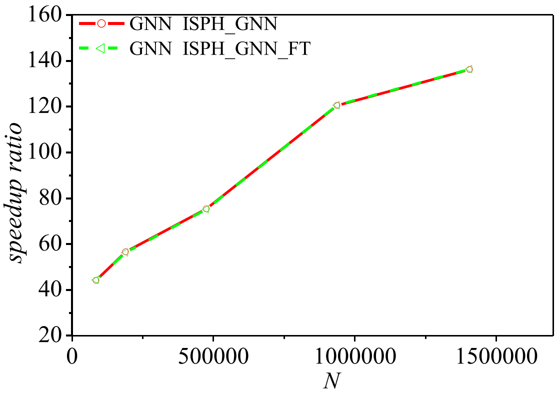


Fig. 31. Speedup ratio of the computational time required for pressure prediction per step in the regular wave interaction with an OSWC cases.

results show that the GNN model, fine-tuned through transfer learning with a very small dataset, can achieve a bit better prediction accuracy than the GNN model without the transfer learning in most cases. In addition, the fine-tuned model significantly overperforms the model what is trained using a small amount of data from scratch. This illustrates that integrating transfer learning with ISPH_GNN can be a way to improve the overall prediction accuracy of ISPH_GNN for violent WSI problems, and also a way to avoid the requirement of large amount of time-consuming data generation for new tasks. This paper just sheds some light of this potential. More extensive studies are need about the performance of transfer learning and how to use it.

Additionally, it is demonstrated that the ISPH_GNN method substantially reduces the computational time required for pressure prediction, achieving a speedup ratio of up to 136 times compared to the conventional ISPH, while maintaining comparable accuracy across the examined cases. This speedup performance achieved in this study for simulating violent WSI cases is considerably higher than that reported in our previous work, where the ISPH_GNN was applied to simulate wave-floater interactions. This indicates that the speedup ratio is case-dependent but the computational efficiency of the ISPH_GNN is very high compared with the conventional ISPH in all the cases we studied so far.

These findings underscore the significant potential of ISPH_GNN for widespread applications in marine engineering. However, the conclusions drawn in this study are based only on the cases analysed in this paper. Although the findings are promising, further validation and investigations with a wider range of scenarios is necessary. In the present study, only single-phase modelling have been considered, meaning air entrainment/entrapment effects were unaccounted for. Apart from these, structural materials and responses (e.g., hydroelasticity) are not considered either. Moreover, this research focuses solely on two-dimensional violent WSIs, which is different from three-dimensional violent WSI in real scenarios. These limitations will be addressed in our future studies.

CRediT authorship contribution statement

Ningbo Zhang: Writing – original draft, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Shiqiang Yan:** Writing – review & editing, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization. **Qingwei Ma:** Writing – review & editing, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors gratefully acknowledge the financial support of EPSRC projects (EP/V040235, EP/T026782 and EP/T00424X).

Appendix. Formulas of QSFDI

The detailed description and derivation of the QSFDI can be found in our previous work, only the final formulas are given here,

$$\nabla P_i = \sum_{j=1}^{N} \boldsymbol{\Phi}_{ji}^{g} (P_j - P_i) \tag{A1}$$

$$\nabla^2 P_i = \boldsymbol{I}^{\mathrm{T}} \sum_{j=1}^{N} \boldsymbol{\Phi}_{ji}^{s} (P_j - P_i) \tag{A2}$$

For convenience, Eq. (A1) and Eq. (A2) are written in a summation form. In the above procedure, $\boldsymbol{I} = [1 \quad 1 \quad 1]^{\mathrm{T}}$ and

$$\boldsymbol{\Phi}_{ji}^{g} = \boldsymbol{M}_{1q,i}^{-1} \left(\frac{W(\mathbf{r}_{ji})}{d_{ji}^{2}} \boldsymbol{q}_{ji} - \sum_{k=1}^{N} \frac{W(\mathbf{r}_{ki})}{d_{ki}^{2}} \boldsymbol{q}_{ki} \left(\mathbf{r}_{ki}^{(2c)}\right)^{\mathrm{T}} \boldsymbol{M}_{2c,i}^{-1} \frac{W(\mathbf{r}_{ji})}{d_{ji}^{4}} \mathbf{r}_{ji}^{(2c)} - \sum_{k=1}^{N} \frac{w_{kI}}{d_{ki}^{2}} \boldsymbol{q}_{ki} \boldsymbol{\Pi}_{ki}^{\mathrm{T}} \boldsymbol{M}_{2s,i}^{-1} \boldsymbol{\Gamma}_{ji} \right) \tag{A3}$$

$$\boldsymbol{\Phi}_{ji}^{s} = 2\boldsymbol{M}_{2s,i}^{-1} \left(\boldsymbol{\Gamma}_{ji} - \sum_{k=1}^{N} \frac{w_{kI}}{d_{ki}^{4}} \boldsymbol{\Pi}_{ki} \boldsymbol{G}_{ki}^{\mathrm{T}} \boldsymbol{\Phi}_{ki}^{g} \right) \tag{A4}$$

Definitions of matrices in Eq. (A3) and (A4) are as follows

$$\boldsymbol{M}_{2c,i} = \sum_{j=1}^{N} \frac{W(\mathbf{r}_{ji})}{d_{ji}^{4}} \mathbf{r}_{ji}^{(2c)} \left(\mathbf{r}_{ji}^{(2c)}\right)^{\mathrm{T}} \tag{A5a}$$

$$\boldsymbol{\Pi}_{ji} = \left\{ \left(\mathbf{r}_{ji}^{(2s)}\right)^{\mathrm{T}} - \left(\mathbf{r}_{ji}^{(2c)}\right)^{\mathrm{T}} \boldsymbol{M}_{2c,i}^{-1} \sum_{k=1}^{N} \frac{W(\mathbf{r}_{ki})}{d_{ki}^{4}} \mathbf{r}_{ki}^{(2c)} \left(\mathbf{r}_{ji}^{(2s)}\right)^{\mathrm{T}} \right\}^{\mathrm{T}} \tag{A5b}$$

$$\boldsymbol{M}_{2s,i} = \sum_{j=1}^{N} \frac{W(\mathbf{r}_{ji})}{d_{ji}^{4}} \boldsymbol{\Pi}_{jI} \boldsymbol{\Pi}_{jI}^{\mathrm{T}} \tag{A5c}$$

$$\boldsymbol{G}_{ji} = \left\{ \mathbf{r}_{ji}^{\mathrm{T}} - \left(\mathbf{r}_{ji}^{(2c)}\right)^{\mathrm{T}} \boldsymbol{M}_{2c,i}^{-1} \sum_{k=1}^{N} \frac{W(\mathbf{r}_{ki})}{d_{ki}^{4}} \mathbf{r}_{ki}^{(2c)} \mathbf{r}_{ki}^{T} \right\}^{\mathrm{T}} \tag{A5d}$$

$$\boldsymbol{q}_{ji} = \left(\boldsymbol{G}_{ji}^{\mathrm{T}} - \boldsymbol{\Pi}_{ji}^{\mathrm{T}} \boldsymbol{M}_{2s,I}^{-1} \sum_{k=1}^{N} \frac{W(\mathbf{r}_{ki})}{d_{ki}^{4}} \boldsymbol{\Pi}_{ki} \boldsymbol{G}_{ki}^{\mathrm{T}} \right)^{\mathrm{T}} \tag{A5e}$$

$$\boldsymbol{\Gamma}_{ji} = \left(\frac{W(\mathbf{r}_{ji})}{d_{ji}^{4}} \boldsymbol{\Pi}_{ji} - \boldsymbol{\Pi}_{ji} \left(\mathbf{r}_{ji}^{(2c)}\right)^{\mathrm{T}} \boldsymbol{M}_{2c,I}^{-1} \sum_{k=1}^{N} \frac{w_{ki}}{d_{ki}^{4}} \mathbf{r}_{ki}^{(2c)} \right) \tag{A5f}$$

$$\boldsymbol{M}_{1q,i} = \sum_{j=1}^{N} \frac{W(\mathbf{r}_{ji})}{d_{ji}^{2}} \boldsymbol{q}_{ji} \boldsymbol{q}_{ji}^{\mathrm{T}} \tag{A5g}$$

In ISPH_CQ, Eq. (A2) is used to discretise the Laplacian in Eq. (9), Eq. (A1) is applied to implementing the pressure boundary condition (Eq. (4)) and to calculating the pressure gradient in Eq. (10). In addition, Eqs. (A1) and (A2) are also extended to finding the velocity divergence and the viscous term, respectively,

$$\nabla \cdot \mathbf{u}_i = \sum_{j=1}^{N} \boldsymbol{\Phi}_{ji}^{g} \cdot (\mathbf{u}_j - \mathbf{u}_i) \tag{A6}$$

$$\nabla \cdot (\nu_i \nabla \mathbf{u}_i) = \sum_{j=1}^{N} \nu_i \boldsymbol{\Phi}_{ji}^{s} \cdot (\mathbf{u}_j - \mathbf{u}_i) \tag{A7}$$

Data availability

Data will be made available on request.

References

[1] S.L. Brunton, B.R. Noack, P. Koumoutsakos, Machine learning for fluid mechanics, Annu Rev. Fluid. Mech. 52 (1) (2020) 477–508.

[2] D. Panchigar, K. Kar, S. Shukla, R.M. Mathew, U. Chadha, S.K. Selvaraj, Machine learning-based CFD simulations: a review, models, open threats, and future tactics, Neural Comput. Appl. 34 (24) (2022) 21677–21700.

[3] J.N. Kutz, Deep learning in fluid dynamics, J. Fluid Mech 814 (2017) 1–4.

[4] L.U. Ladický, S. Jeong, B. Solenthaler, M. Pollefeys, M. Gross, Data-driven fluid simulations using regression forests, ACM Trans. Graph. (TOG) 34 (6) (2015) 1–9.

[5] C. Jiang, Q. Zhang, O. Moctar, P. Xu, T. Iseki, G. Zhang, Data-driven modelling of wave–structure interaction for a moored floating structure, Ocean Eng. 300 (2024) 117522.

[6] S.Cai X.Jin, H. Li, G.E. Karniadakis, NSFnets (Navier-Stokes flow nets): physics-informed neural networks for the incompressible Navier-Stokes equations, J. Comput. Phys. 426 (2021) 109951.

[7] M. Raissi, A. Yazdani, G.E. Karniadakis, Hidden fluid mechanics: learning velocity and pressure fields from flow visualizations, Science 367 (6481) (2020) 1026–1030.

[8] Z. Mao, A.D. Jagtap, G.E. Karniadakis, Physics-informed neural networks for high-speed flows, Comput. Methods Appl. Mech. Eng. 360 (2020) 112789.

[9] R. Halder, M. Damodaran, B.C. Khoo, Deep learning-driven nonlinear reduced-order models for predicting wave-structure interaction, Ocean Eng. 280 (2023) 114511.

[10] J. Tompson, K. Schlachter, P. Sprechmann, K. Perlin, Accelerating eulerian fluid simulation with convolutional networks, in: International Conference on Machine Learning. PMLR, 2017, pp. 3424–3433.

[11] J. Ling, A. Kurzawski, J. Templeton, Reynolds averaged turbulence modelling using deep neural networks with embedded invariance, J. Fluid Mech 807 (2016) 155–166.

[12] D. Kochkov, J.A. Smith, A. Alieva, Q. Wang, M.P. Brenner, S. Hoyer, Machine learning–accelerated computational fluid dynamics, Proc. Natl. Acad. Sci. 118 (21) (2021) e2101784118.

[13] R.A. Gingold, J.J. Monaghan, Smoothed particle hydrodynamics – theory and application to non-spherical stars, Mon. Not. R. Astro. Soc 181 (3) (1977) 375–389.

[14] S.D. Shao, E.Y.M. Lo, Incompressible SPH method for simulating newtonian and non-newtonian flows with a free surface, Adv. Water Resour 26 (7) (2003) 787–800.

[15] V. Sriram, Q.W. Ma, Improved MLPG_R method for simulating 2D interaction between violent waves and elastic structures, J. Comput. Phys. 231 (22) (2012) 7650–7670.

[16] A. Khayyer, N. Tsuruta, Y. Shimizu, H. Gotoh, Multi-resolution MPS for incompressible fluid-elastic structure interactions in ocean engineering, Appl. Ocean Res. 82 (2019) 397–414.

[17] P.N. Sun, C. Pilloton, M. Antuono, A. Colagrossi, Inclusion of an acoustic damper term in weakly-compressible SPH models, J. Comput. Phys. 483 (2023) 112056.

[18] N.B. Zhang, Q.W. Ma, X. Zheng, S.Q. Yan, A two-way coupling method for simulating wave-induced breakup of ice floes based on SPH, J. Comput. Phys. 488 (2023) 112185.

[19] X. Zheng, Q.W. Ma, W.Y. Duan, Incompressible SPH method based on Rankine source solution for violent water wave simulation, J. Comput. Phys 276 (2014) 291–314.

[20] A. Khayyer, H. Gotoh, H. Falahaty, Y. Shimizu, An enhanced ISPH–SPH coupled method for simulation of incompressible fluid–elastic structure interactions, Comput. Phys. Commun 232 (2018) 139–164.

[21] N.B. Zhang, S.Q. Yan, Q.W. Ma, X. Zheng, A QSFDI based laplacian discretisation for modelling wave-structure interaction using ISPH, Appl. Ocean Res 117 (2021) 102954.

[22] N. Tsuruta, A. Khayyer, H. Gotoh, Development of Advective Dynamic Stabilization scheme for ISPH simulations of free-surface fluid flows, Comput. Fluids. 266 (2023) 106048.

[23] N.B. Zhang, X. Zheng, Q.W. Ma, W.Y. Duan, A. Khayyer, X.P. Lv, S.D. Shao, A hybrid stabilization technique for simulating water wave–structure interaction by incompressible smoothed particle hydrodynamics (ISPH) method, J. Hydro-env. Res 18 (2018) 77–94.

[24] Q.W. Ma, Y. Zhou, S.Q. Yan, A review on approaches to solving Poisson's equation in projection-based meshless methods for modelling strongly nonlinear water waves, J. Ocean Eng. Mar. Energy 2 (3) (2016) 279–299.

[25] N.B. Zhang, S.Q. Yan, Q.W. Ma, X.H. Guo, Z.H. Xie, X. Zheng, A CNN-supported lagrangian ISPH model for free surface flow, Appl. Ocean Res 136 (2023) 103587.

[26] M. Gori, G. Monfardini, F. Scarselli, A new model for learning in graph domains, in: Proceedings. International Joint Conference on Neural Networks, IEEE, 2005, pp. 729–734.

[27] Y. Li, D. Tarlow, M. Brockschmidt, R. Zemel, Gated graph sequence neural networks, 2015, arXiv preprint arXiv:1511.05493.

[28] K. Kumar, J. Vantassel, GNS: a generalizable graph Neural network-based simulator for particulate and fluid modeling. arXiv preprint arXiv:2211.10228, (2022).

[29] Z.J. Li, A.B. Farimani, Graph neural network – accelerated lagrangian fluid simulation, Comput. Graph 103 (2022) 201–211.

[30] N.B. Zhang, S.Q. Yan, Q.W. Ma, Q. Li, A hybrid method combining ISPH with graph neural network for simulating free-surface flows, Comput. Phys. Commun. 301 (2024) 109220.

[31] N.B. Zhang, S.Q. Yan, Q.W. Ma, Q. Li, Numerical simulation of wave-floater interactions using ISPH_GNN trained on data for wave-only cases, Ocean Eng. 306 (2024) 118041.

[32] S.J. Pan, Q. Yang, A survey on transfer learning, IEEe Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.

[33] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks? Adv. Neural Inf. Process. Syst. 27 (2014).

[34] F. Zhuang, Z. Qi, K. Duan, D. Xi, H.Zhu Y.Zhu, H. Xiong, Q. He, A comprehensive survey on transfer learning, Proc. IEEE 109 (1) (2020) 43–76.

[35] S. Verma, Z.L. Zhang. Learning universal graph neural network embeddings with aid of transfer learning. arXiv preprint arXiv:1909. (2019) 10086.

[36] N. Kooverjee, S. James, T. Van Zyl, Investigating transfer learning in graph neural networks, Electron. (Basel) 11 (8) (2022) 1202.

[37] J.J. Monaghan, J.C. Lattanzio, A refined method for astrophysical problems, Astron. Astrophys 149 (1) (1985) 135–143.

[38] Q.W. Ma, A new meshless interpolation scheme for MLPG_R method, CMES Comput. Model. Eng. Sci. 23 (2) (2008) 75–89.

[39] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, S.Y. Philip, A comprehensive survey on graph neural networks, IEEE T. Neur. Net. Lear 32 (1) (2020) 4–24.

[40] D. Hendrycks, K. Gimpel, Gaussian error linear units (gelus), 2016, arXiv preprint arXiv:1606.08415.

[41] N.B. Zhang, S.Q. Yan, Q.W. Ma, A. Khayyer, X.H. Guo, X. Zheng, A consistent second order ISPH for free surface flow, Comput. Fluids 274 (2024) 106224.

[42] S.Q. Yan, Q.W. Ma, J.H. Wang, Quadric SFDI for laplacian discretisation in lagrangian meshless methods, J. Mar. Sci. Appl 19.3 (2020) 362–380.

[43] S. Adami, X.Y. Hu, N.A. Adams, A generialised wall boundary condition for smoothed particle hydrodynamics, J. Comput. Phys 231 (2012) 7057–7075.

[44] J.J. Monaghan, A. Rafiee, A simple SPH algorithm for multi-fluid flow with high density ratios, Int. J. Numer. Methods Fluids. 71 (5) (2013) 537–561.

[45] A. Rafiee, S. Cummins, M. Rudman, K. Thiagarajan, Comparative study on the accuracy and stability of SPH schemes in simulating energetic free-surface flows, Eur. J. Mech. - B/Fluids 36 (10) (2012) 1–16.

[46] X. Zheng, Z.H. Hu, Q.W. Ma, W.Y. Duan, Incompressible SPH based on rankine source solution for water wave impact simulation, Procedia Eng. 126 (2015) 650–654.

[47] M. Greenhow, W.M. Lin, Nonlinear-free surface effects: experiments and theory, technical report, in: DTIC Document, 1983.

[48] H. Sun, O.M. Faltinsen, Water impact of horizontal circular cylinders and cylindrical shells, Appl. Ocean Res 28 (2006) 299–311.

[49] S.C. Hsiao, T.C. Lin, Tsunami-like solitary waves impinging and overtopping an impermeable seawall: experiment and RANS modeling, Coast. Eng 57 (1) (2010) 1–18.

[50] Q.W. Ma, J.T. Zhou, MLPG_R method for numerical simulation of 2D breaking waves. CMES-Comp, Model Eng 43 (3) (2009) 277–303.

[51] A. Henry, O. Kimmoun, J. Nicholson, G. Dupont, Y. Wei, F. Dias, A two dimensional experimental investigation of slamming of an oscillating wave surge converter, in: Proceedings of the 24th International Offshore and Polar Engineering Conference, Busan, Korea, 2014, pp. 296–305. June.

[52] Y.J. Wei, T. Abadie, A. Henry, F. Dias, Wave interaction with an oscillating wave surge converter, II: Slamming. Ocean. Eng. 113 (2016) 319–334.