EFFICIENT SEISMIC NUMERICAL MODELING TECHNIQUE USING THE YOLOv2-BASED EXPANDING DOMAIN METHOD

DAWOON LEE[1], SEUNGPYO CHOI[2], JONGHYUN LEE[3] and WOOKEEN CHUNG[4]

[1] *Department of Ocean Energy and Resources Engineering, Korea Maritime and Ocean University, Busan, South Korea.*
[2] *Department of Convergence Study on the Ocean Science and Technology, Ocean Science and Technology (OST) School, Korea Maritime and Ocean University, Busan, South Korea*
[3] *Department of Civil and Environmental Engineering and Water Resources Research Center, University of Hawai'i at Manoa, Honolulu, HI 96822, U.S.A.*
[4] *Department of Ocean Energy and Resources Engineering, Korea Maritime and Ocean University, Busan, South Korea. wkchung@kmou.ac.kr*

(Received April 11, 2022; accepted August 5, 2022)

ABSTRACT

D.W. Lee, S.P. Choi, J.H. Lee and W.K. Chung. Efficient seismic numerical modeling technique using the YOLOv2-based expanding domain method. *Journal of Seismic Exploration,* 31: 425-449.

Wave equation-based seismic modeling has the advantage of simulating the exact full-wave propagation. However, it requires a great amount of computational resources, which becomes prohibitive when both the modeling domain size and the number of the time samples increase. Therefore, much research has been performed to enhance the computational efficiency of seismic numerical modeling. The expanding domain method is such one technique that improves the computational efficiency by identifying the domain extent where the wave propagation has not reached and excluding these domains from the calculation. In this work, we propose a new deep-learning based method that guide the determination of the computational domain. In order to establish the computational domain where the wave propagates from the snapshots, the deep learning-based object detection was employed. The deep learning object detector used has two main components. The first one is a structure for the feature extraction layers based on ResNet-50. The second one is a structure for the detection of the wave propagation domain based on the You Only Look Once method, version 2 (YOLOv2). After the training, validation and test for the YOLOv2 object detector, the computational efficiency of our proposed method was compared with that of the widely used amplitude comparison-based expanding domain method. It was demonstrated that the computational efficiency of the YOLOv2 method was better when the number of modeling grids was large, and the efficiency in the largest number of grids was about 25.1 %.

KEY WORDS: seismic numerical modeling, computational efficiency, expanding domain method, deep learning object detection, YOLOv2.

0963-0651/22/$5.00

INTRODUCTION

Seismic numerical modeling is a technique used to simulate wave propagation in various types of geomedia (Drossaert and Giannopoulos, 2007). This technique has been used not only for predicting physical phenomenon but also for seismic data analysis and synthetic data acquisition (Ramsden et al., 2005; Etgen and O'Brien, 2007). Furthermore, it is used as a forward modeling tool for requiring full-waveform information during the seismic imaging such as full-waveform inversion and reverse-time migration. In seismic modeling, various types of wave equations, such as acoustic and elastic, can be applied depending on the type of data acquired from the seismic exploration and also the characteristics of the area where the exploration was performed. To perform seismic numerical modeling, many numerical methods have been introduced, including the finite difference method (Vireux, 1986; Levander, 1988), staggered-grid finite difference method (Graves, 1996; Özdenvar and McMechan, 1997), finite element method (Santos et al., 1988), and spectral element method (Komatitsch and Tromp, 1999). To simulate wave propagation more accurately, many seismic numerical modeling studies considering various types of parameters have been reported (Carcione et al., 1988; Robertsson et al., 1994; Alkhalifah, 2000; Carcione et al., 2002; Fowler and King, 2011). Although the wave equation-based seismic numerical modeling has the advantage of simulating full-wave information, it requires a great amount of computational resources (Petrov and Khokhlov, 2014). And, when both the spatial extent of modeling and the time sample increase, there is a computational prohibitive. In order to enhance the computational efficiency, various model methods have been proposed such as graphics processing unit (GPU)-based seismic modeling with reverse-time migration (Abdelkhaleck et al., 2009); Laplace-domain modeling and inversion using a logarithm grid technique (Ha and Shin, 2012); efficient wave equation-based seismic modeling using a neural network (Siahkoohi et al., 2019); and the expanding domain method (EDM) (Suh and Wang, 2011).

Among many methods, EDM enhances the computational efficiency by skipping computation of the domain where the wave does not propagate at an early time. Therefore, in the EDM, the determination of the domain extent where the wave propagates is important. Suh and Wang (2011) have reported three different strategies to identify the relevant propagation area in time domain: the method of constant velocity layer, solving the eikonal equation, and amplitude comparison. Among these three methods, it is shown that the method of amplitude comparison can define the computational domain as compact as the eikonal method can do, with a negligible computing overhead (Suh and Wang, 2011). The efficiency of the amplitude comparison method depends on various modeling parameters, including the model size, recording time, velocity model, and source and

receiver geometry (Ryu et al., 2015). Since the seismic modeling is performed on the area where the wave is propagating, one expects that the amplitude comparison method loses its computational efficiency when the downgoing size increases. In other words, If the processing time for amplitude comparison method becomes comparable to the time saving by the domain reduction, the EDM is no longer beneficial.

Recently, deep learning techniques have been applied actively in geophysics to solve various problems (Wang et al., 2019; Yang and Ma, 2019; Jun et al., 2020; Kim et al., 2020; Vrolijk and Blacquière, 2021). Deep learning is a part of machine learning that uses many layers of artificial neural networks (NNs), i.e., successive simple nonlinear functions, performs complex tasks by learning the weights of NNs from the data. (Chassagnon et al., 2020). Various deep learning models and techniques have been developed for various applications in geophysics (Pochet et al., 2018; Li et al., 2019). Object detection is one such technique that separates the object from the background, derives its location, and classifies it (Szegedy et al., 2013). Three types of object detection series have been reported: region-based convolutional neural network (Girshick et al., 2014; Girshick, 2015), single-shot multi-box detector (Liu et al., 2016; Womg et al., 2018), and You Only Look Once (YOLO, Redmon et al., 2016; Redmon and Farhadi, 2017; Redmon and Farhadi, 2018). Among these objective detection methods, we focused on the YOLO series due to its superiority and popularity. The YOLO is the method of predicting the location and class of the object by looking at an image once, like its name. In particular, the YOLO method is the object detector for real-time image processing. In 2017, Redmon and Farhadi proposed YOLO version 2 (v2), which has an improved detection speed and accuracy compared to the previous version.

In this paper, we propose an efficient seismic numerical modeling method by combining both the YOLOv2 method and the EDM. Once trained, the YOLOv2 method was able to define the computational domain much faster than the amplitude comparison method. In particular, the computational efficiency did not decrease even in a spatially large model. The organization of this article is as follows: in the Theory section, YOLOv2 object detection and amplitude comparison method are presented in detail along with computational efficiency analysis. In the Methods section, we describe our deep-learning method architecture and specific configurations such as dataset generation, data labeling, training, validation, and detector testing. In the Numerical Examples section, we confirm the performance of the proposed method by comparing with the computational efficiency of the amplitude comparison method through numerical examples. The experiment compared the computational efficiency of the seismic numerical modeling with various sizes of velocity models, each with different grid sizes. Consequently, it was confirmed that the efficiency of YOLOv2 method was better than the amplitude comparison method under various grid modeling conditions.

THEORY

Expanding Domain Method (EDM)

The 2D acoustic wave equation in the time domain performed to simulate the propagation of acoustic waves is as follows:

$$\frac{1}{c^2(x,z)}\frac{\partial^2 P(x,z,t)}{\partial t^2} = \nabla^2 P(x,z,t) + f(t)\delta(x-x_0)\delta(z-z_0) \quad , \qquad (1)$$

where c is P-wave velocity, P is the scalar pressure wavefield, f is source wavefield, and x_0 and z_0 are source positions of x- and z-direction, respectively. In the initial time step of seismic modeling, most of the wavefield in the time domain has a value of zero except near the location of the source. And the EDM reduces the computation time by skipping the computation of the domain where the wavefield is zero thus numerical computation is not needed. Fig. 1 shows the wavefield snapshots at t = 0.8 s that were obtained by time domain modeling with homogeneous medium; these images briefly explain the concept of the EDM. In general, when solving the acoustic wave equation to simulation wave propagation as in Fig. 1, discretization and numerical computation of the partial differential equation (PDE) solution are performed typically on the whole domain (the yellow dotted box in Fig. 1).

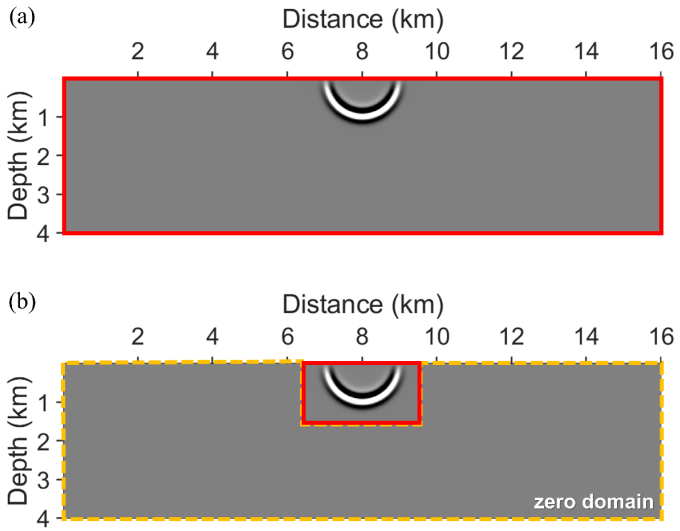

Fig. 1. The wavefield images of seismic numerical modeling in the time domain using the constant velocity (1500 m/s) model at an early time step (0.8 s). The red line indicates the computational domain, and the yellow dotted line indicates zero area. (a) Typical numerical modeling; (b) Numerical modeling using the expanding domain method.

The PDEs over the whole domain should be computed at every time step. Note that adaptive mesh refinement strategies (Tang and Shao, 2013) where the mesh can be refined around the wavefront may be applied to lessen the computational burden by coarsening the mesh around the locations far away from the sources, however, the implementation of the adaptive mesh refinement to existing forward wave simulation solvers can be time-consuming and also require overhead costs of remeshing in each timestep. On the other hand, EDM can eliminate the need for whole domain modeling and adaptive mesh refinement by considering the smaller domain in which the wave propagation is computed (the red box shown in Fig. 1).

The most important factor for successful EDM implementation is the selection of the appropriate computational domain. There are three types of EDMs using constant velocity layer, eikonal equation solver, and amplitude comparison (Suh and Wang, 2011). In the method of the constant velocity layer, the traveltime of the wave is defined under the assumption that the velocity model is simplified by having a constant velocity. This method is simple and fast to identify the wave propagation domain; however, its accuracy can be diminished when the velocity model is complex. Solving the eikonal equation is the traditional way of the traveltime calculation (Vidale, 1990) and the determination of the wave propagation domain with the eikonal equation can provide a comparatively compact domain size, while solving the eikonal equation can be computationally expensive.

Unlike the two methods previously mentioned, the amplitude comparison method defines the nonzero wave domain without calculating the traveltime. Instead, the computation boundary is updated and follows the wavefront by comparison with the maximum amplitude of the wavefield in each time step. Suh and Wang (2011) have demonstrated that the amplitude comparison method using 1/1024 of the maximum amplitude of the wavefield snapshot as the threshold produces a nearly similar computational domain to that produced from the eikonal equation solver-based method with less computational cost. However, in this study, we use 1E-6 for the maximum amplitude of the wavefield snapshot as the baseline for the amplitude comparison method for more accurate computational domain selection.

Expanding Domain Method (EDM)

The computational efficiency of the amplitude comparison expanding domain method (AC-EDM) depends on various modeling parameters such as the acquisition geometry, recording time, velocity model, and model size (Ryu et al., 2015). Among them, the model size is a critical parameter since the AC-EDM becomes inefficient as the model size becomes large. To confirm this, we compared the computation time of

seismic numerical modeling with and without the AC-EDM with respect to the model size, and analyzed the computational efficiency of the AC-EDM. The second-order finite difference method was used with the non-reflecting boundary condition (Cerjan et al., 1985) for the time domain modeling. The Marmousi-2 benchmark model (Martin et al., 2006) was used as the velocity model for this experiment shown in Fig. 2. The Marmousi-2 benchmark is the 17 km long and 3.5 km deep velocity model with 13601 (length) × 2081 (depth) grids with a uniform grid size of 1.25 m. In addition, we created coarse grid velocity models from the Marmousi-2 benchmark with a grid of 2.5 m, 5 m, and 10 m (Table 1) to investigate the performance when the resolution of the snapshots was improved to check whether the AC-EDM operates normally at the other resolution. The sampling time intervals in each of the four numerical modeling environments were determined by the grid size of each case, as shown in Table 1.

Table 1. The seismic numerical modeling parameters of four cases using the Marmousi-2 p-wave velocity model.

Modeling parameters	**Case 1**	**Case 2**	**Case 3**	**Case 4**
Grid size	10 m	5 m	2.5 m	1.25 m
Number of grids	1701 × 351	3401 × 701	6801 × 1401	13601 × 2801
Sampling time interval	1.5 ms	0.75 ms	0.375 ms	0.1875 ms

In terms of the acquisition geometry, a source was located at a distance of 8.5 km and a depth of 20 m below the sea level, and receivers were located at all points of the grids at 20 m below the sea level. The recording time was 5.7 s. The computational efficiency was calculated as follows:

$$Computational\ efficiency\ (\%) = \left(\frac{T_{ref} - T_{EDM}}{T_{ref}}\right) \times 100, \qquad (2)$$

where T_{ref} is the computation time of modeling without the EDM (reference modeling), and T_{EDM} is the computation time of modeling using the EDM. In this section, T_{EDM} corresponds to the computation time of the AC-EDM.

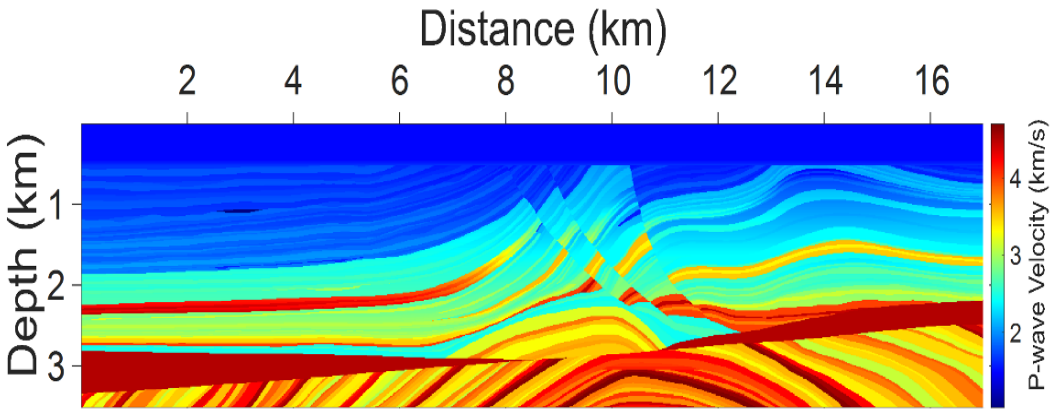

Fig. 2. P-wave velocity with the Marmousi-2 model.

All of these numerical modeling runs were implemented in MATLAB performed on a workstation equipped with an Intel Core i9-10980XE CPU (18 cores) at 3.70 GHz and 128 GB RAM. The results of the computational experiments through this simple numerical example are shown in Table 2. The results are the mean of three measurements. Note that the computational efficiency of the AC-EDM decreased as the model size increased. This result is due to the algorithmic aspect of the AC-EDM, which examines the amplitude values from the wavefronts at each time step and when the discretization grids become finer, the overhead cost of determining the correct boundary size surpasses the computational gain in the modeling.

Table 2. The computation time and efficiency of four seismic numerical modeling cases using the amplitude comparison expanding domain method (AC-EDM).

Modeling method	**Case 1**	**Case 2**	**Case 3**	**Case 4**
Computation time for Reference modeling[1)]	3 min 9 s	25 min 5 s	3 h 20 min 30 s	1 d 2 h 37 min 7 s
Computation time for AC-EDM modeling[2)]	2 min 36 s	23 min 7 s	3 h 25 min 38 s	1 d 4 h 11 min 4 s
Computational efficiency	**17.5 %**	**7.8 %**	**- 2.6 %**	**- 5.9 %**

1) seismic numerical modeling without the expanding domain method, and 2) amplitude comparison expanding domain method.

YOLOv2 object detector

We confirmed the computational efficiency of the AC-EDM. Using the EDM, it seems that the efficiency should be improved as the modeling domain increases, however, the opposite result was confirmed by using the AC-EDM. This is because the calculation to ascertain the computational domain requires a lot of time. We expected that the modeling efficiency could be improved by using a pre-trained deep learning tool to determine the computational domain. Among the various deep learning methods, we applied the deep learning-based YOLOv2 object detector, which derives the computational domain fast, even with large model size.

YOLO is an end-to-end object detection method that was proposed for real-time image processing in 2016 (Redmon et al., 2016). YOLOv2 is an improved version of the YOLO in terms of the detection accuracy and processing speed of YOLO (Redmon and Farhadi, 2017). YOLOv2 is efficient object detection by resizing the anchor box closest to the object rather than estimating the size of the object using a predefined object size (anchor box). Therefore, it is important to estimate the number and size of anchor boxes in order to generate a high-performance detector. In the next sections, we will show how the YOLOv2 object detector is trained and combined with the EDM to rapidly derive the computational domain for efficient seismic numerical modeling with large model size.

METHODS

Data collection and labeling

In this study, snapshot images were used as the input dataset. A snapshot image records the propagation pattern of a wave at an arbitrary time. The snapshots take various forms depending on the background velocity, the source position and type, the size of the exploration area, and the acquisition time. For training data creation, two-dimensional finite-difference time-domain seismic numerical modeling was performed in various model parameter configurations to extract numerous feature maps and train the detection model applicable to unseen data. Five different source location points, two different source functions, and 14 different velocity models were used. In particular, the 14 velocity models consisted of seven 1×1 ($2.24\ km \times 2.24\ km$) and seven 1×2 ($1.25\ km \times 2.50\ km$) velocity models, depending on the ratio of the depth and distance. This configuration is a part of the multiscale training intended to generalize the proposed deep learning model with various velocity model ratios. The velocity models used for the training dataset construction included homogeneous models, simple step models, and complex models. The complex models include the Marmousi2 model and the SEG/EAGE overthrust model (Aminzadeh et al., 1994), which are benchmark models.

All of the velocity models used to build the input dataset and source locations are shown in Fig. 3. The number of sources used in each model is 5. The source depth is $20\,m$ and the horizontal positions are $370\,m.740\,m.1{,}110\,m.1{,}480\,m.1{,}850\,m$ at "1 × 1 velocity model", $410\,m, 820\,m, 1{,}230\,m, 1{,}640\,m, 2{,}050\,m$ at "1 × 2 velocity model".

Fig. 3 (a) shows the 1 × 1 velocity models, (b) shows the 1 × 2 velocity models, and (c) shows the source locations used for input data generation, and the two types of source functions used to generate the data are shown in Fig. 4. In order to generate the wavefield changes over time, 95 images were created at 10-ms intervals, for a recording time of 1 s. The wavefield data after 50 ms were used because the seismic signal was too small at the early time points. In summary, 95 wavefield snapshot images for each model parameter configuration was generated with 140 different model parameter configurations resulting in a total of 13,300 wavefield training data. After the wavefield training data were generated, they were resized and normalized. The deep neural network input size used in this study was 224×224 pixels, the same as in the ResNet-50 (He et al., 2016) architecture. The non-zero wavefield area of the resized data was derived using the amplitude comparison method. The generated dataset and corresponding image labels are shown in Fig. 5. Finally, the labeled dataset was split up as follows: 70% of training data, 10% of validation data, 10%, and 20% of test data.

Anchor box estimation

Estimating the number of anchor boxes and their size is one of the most important stages for generating a high-performance detector (Loey et al., 2021). Increasing the number of anchors can improve the mean intersection-over-union (IoU) score; however, this can also increase the computational cost and lead to the overfitting of the detector (Itakura and Hosoi, 2020). Therefore, it is important to determine the appropriate anchor properties. To decide the number of anchor boxes, we used the 'estimateAnchorBoxes' function in MATLAB, which uses the mean IoU distance metric. The IoU is calculated as (Barhakur and Sarma, 2019).

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{3}$$

This function leads to boxes with similar aspect sizes and ratios being clustered together, thus providing anchor boxes that fit the dataset. The number of anchor boxes was estimated using the labeled dataset as explained in "Data collection and labeling" section. In the labeled dataset, the optimal number of anchor boxes was determined as 13 which maximizes the mean IoU (0.819) as shown in Fig. 6. The optimal sizes of these 13 anchors are shown in Table 3.

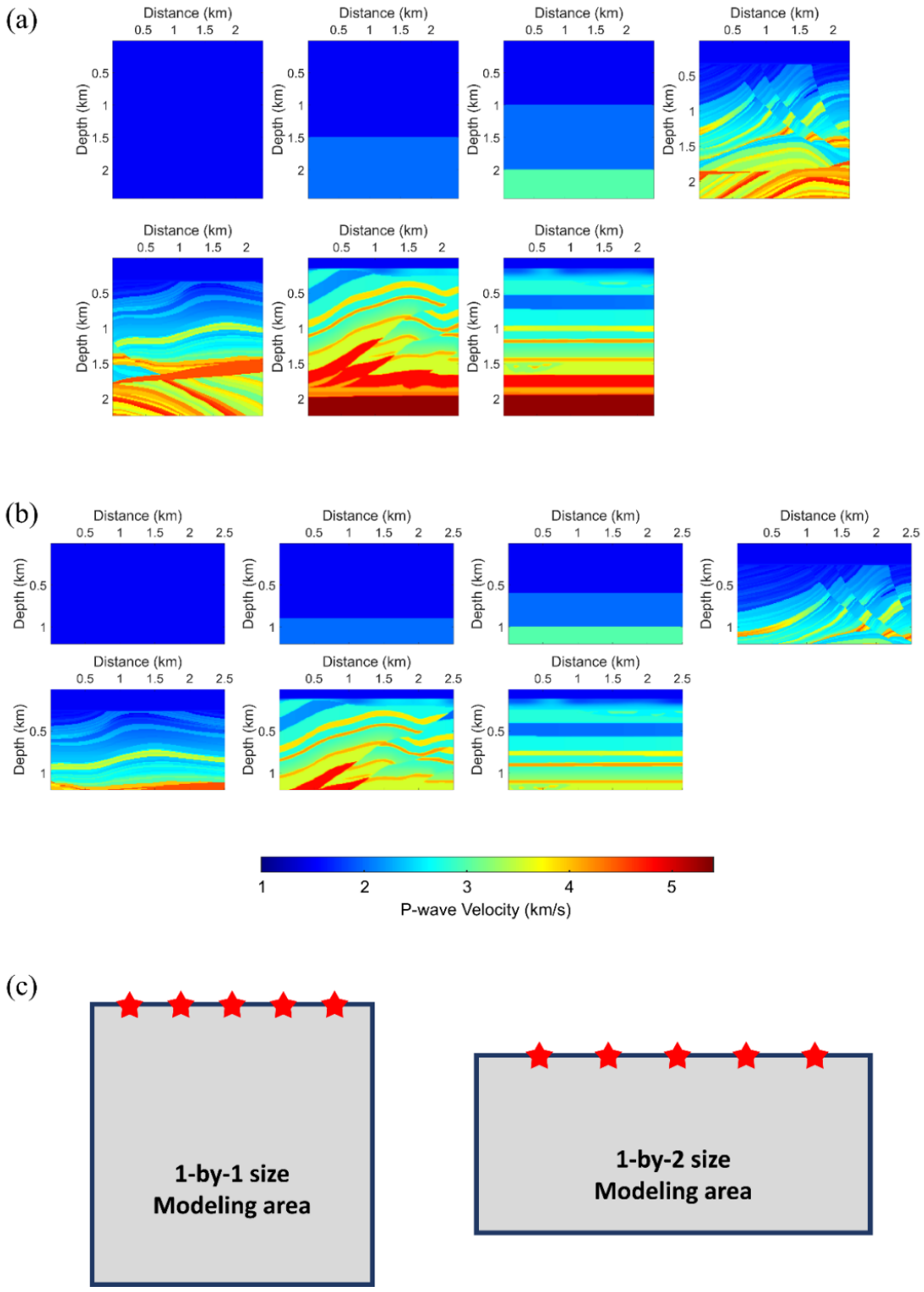

Fig. 3. Fourteen different velocity models with various numerical modeling environments that were used for dataset construction. The images shown in panel (a) are velocity models with a 1:1 ratio of depth and height, and the images shown in panel (b) are velocity models with a 1:2 ratio of depth and height, and (c) are five different points of source locations for each model ratio.

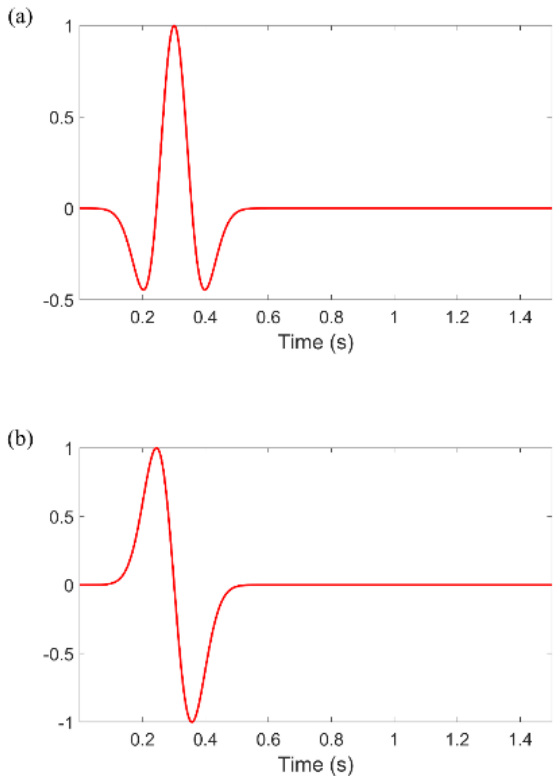

Fig. 4. To generate the training data, (a) the Ricker wavelet and (b) the first-derivative Gaussian function were used.

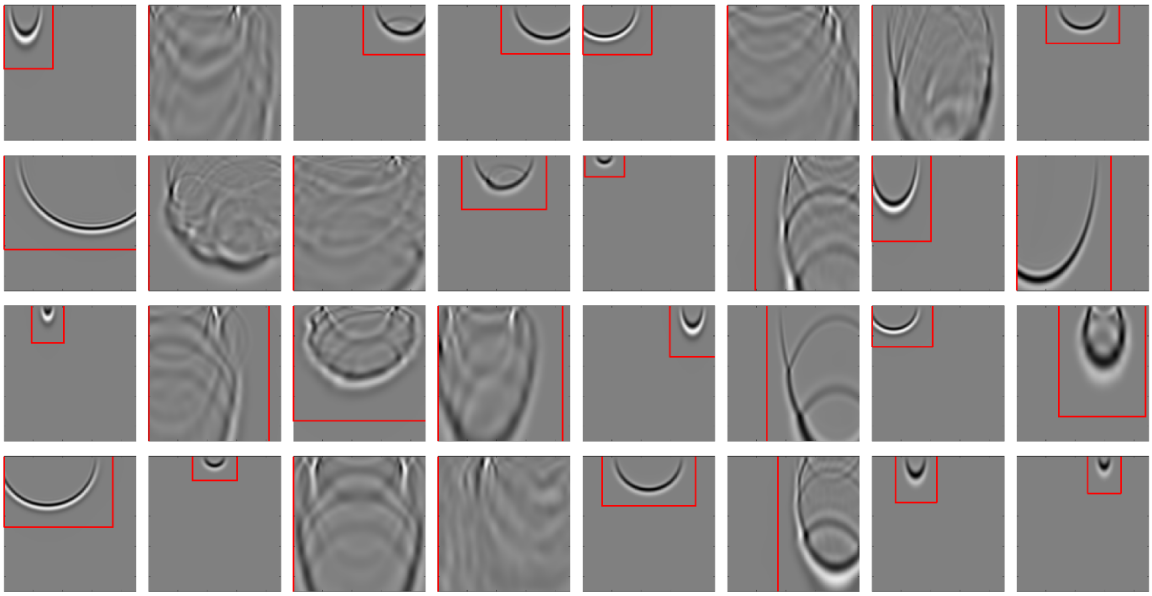

Fig. 5. Snapshot image of the generated input dataset and the label data (red boxes).

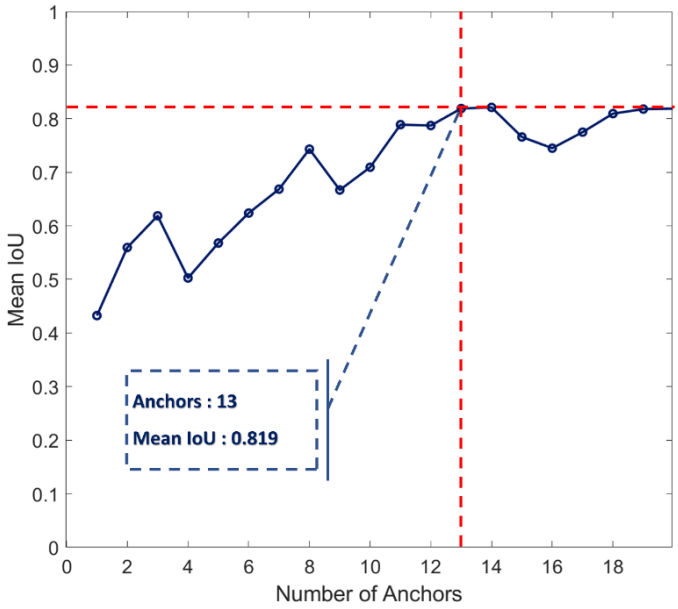

Fig. 6. Mean intersection-over-union (IoU) values for different numbers of anchors for the wavefield images.

Neural network architecture

In general, YOLOv2 uses Darknet-19 as the default feature extraction layer. In this study, ResNet-50, which has been previously confirmed to perform with higher accuracy, was used as the feature extraction network to detect the computational domain in the wavefield data through seismic numerical modeling in real time (Sadak et al., 2020; Loey et al., 2021). ResNet-50 is one of the powerful deep neural networks which has shown excellent performance for representation learning and other recognition tasks; in fact, it won first place in various categories in the ImageNet Large Scale Visual Recognition Challenge and the 2015 COCO competitions (He et al., 2017). This network is built with 50 layers with 224 × 224 image input size; in addition, it is built with residual networks using shortcut connections that skip the convolutional layers, so that multi-scale features in the images can be captured and learned effectively. The total deep neural network architecture is shown in Fig. 7. Table 4 is the detailed network architecture for YOLOv2 layers in Fig. 7. The YOLOv2 layer uses the "Features" extracted from the "Feature extraction layers" to estimate the detected computational domain (x-, y-axis, width, height, class) of wavefield data.

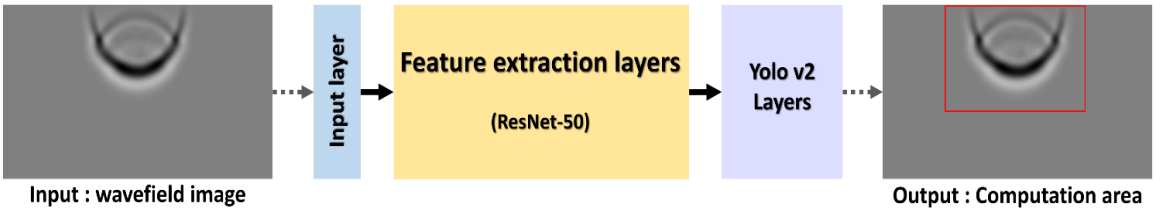

Fig. 7. Network architecture of YOLOv2 object detector expanding domain method.

Table 2. The detailed architectures of the YOLO v2 layers in Fig. 7.

#	YOLO v2 Layers
1	[3×3] Convolution 2D layer (1024) {1,1} + Batch Normalization layer + Relu
2	[3×3] Convolution 2D layer (1024) {1,1} + Batch Normalization layer + Relu
3	[3×3] Convolution 2D layer (78) {1,1}
4	YOLO v2 transform layer
5	YOLO v2 output layer

Neural network training and validation

The YOLOv2 object detector was trained with training and validation data sets, and test data were used for testing the generalization performance. With hyperparameter tuning, the Adam optimizer (Kingma and Ba, 2014) was chosen with a mini-batch size of 32, the initial learning rate of 0.0001. Since the loss and the root mean square error (RMSE) converged when the epoch exceeds 20, the number of epochs was set as 20. The data were shuffled at each epoch to prevent overfitting. The deep neural network was trained in a multi-GPU (two NVIDIA GeForce RTX 3090 GPUs) environment. The training hyper-parameters and GPU specifications are summarized in Table 5. Fig. 8 shows the loss and the RMSE values of training and validation over epochs. It was observed that the loss and RMSE decreased both for the training and validation data sets as the training epoch progressed, which indicates that the YOLOv2 object detector was trained well without overfitting issues. The trained YOLOv2 object detector processes 144 frames per second for an image of 224×224 size, which means it takes 0.023 sec to process one image. The test data set is explained in the next section.

Table 5. The training options of the detector model.

Batch size	Epoch	Data shuffle	Optimizer	Learning rate	Training environment
32	20	per every epoch	Adam	0.0001	2X NVIDIA GeForce RTX 3090 GPUs

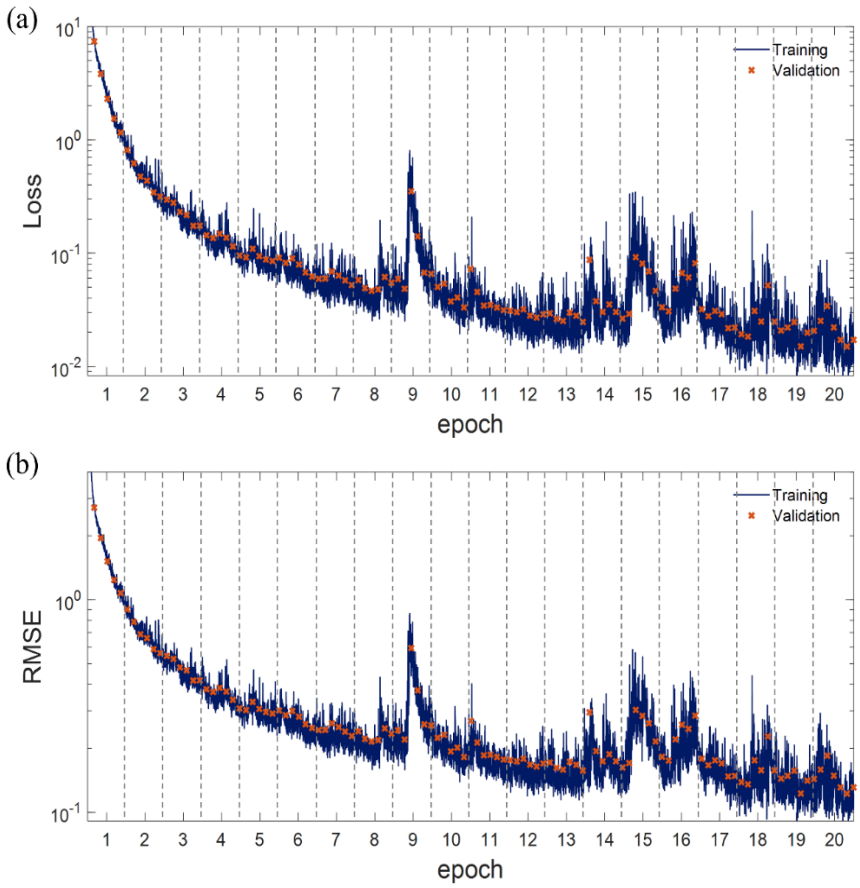

Fig. 8. The training process with (a) loss of training and validation, and (b) root mean square error (RMSE) of training and validation.

Detector performance test

The most common performance test for object detection is average precision (AP) (Deng et al., 2009; Geiger et al., 2013; Lin et al., 2014). The AP is defined as the ratio of the sum of precision to recall as follows:

$$Average\ Precisio\ \frac{\sum precision}{recall} \quad , \tag{4}$$

and evaluate the ability of the detector to find all relevant objects and to correctly detect objects (Padilla et al., 2020). Precision is the ability of the detector to handle only the objects expressed by the detector. It is the percentage of correct answers among the results derived by the detector. Recall is the ability of the detector to find all objects in the data. It is the percentage of correct answers among all given ground truths. They are values between 0 and 1. The ideal precision occurs at all recall levels. In many applications, a truly positive detection is based on an IoU threshold >50% in the AP test (Sadak et al., 2020). However, in our application, it was found that the accuracy was insufficient with a threshold of 50%. Therefore, we tested the performance at a higher threshold of 85%. The AP test result is shown in Fig. 9. Despite the high threshold, a high-performance detector was constructed with AP = 0.94. Lastly, we made a final check of the trained YOLOv2 object detector with a new wave propagation problem. Seismic numerical modeling with the YOLO v2 object detector-based EDM (YOLOv2-EDM) was carried out using the SEG/EAGE salt dome (Aminzadeh et al., 1994) with a crossline 4-km-sliced two-dimensional P-wave velocity model (Fig. 10), which is a benchmark model not used in training data generation. This velocity model had a length of 12 km, a depth of 4 km, and a grid size of 10 m. In this test, we used a Ricker wavelet with a dominant frequency of 4 Hz as the source, and the source was located 10 m under the water. The computational domain determined by the detector is shown in Fig. 11. Through this test, the performance of the YOLOv2 object detector was further confirmed.

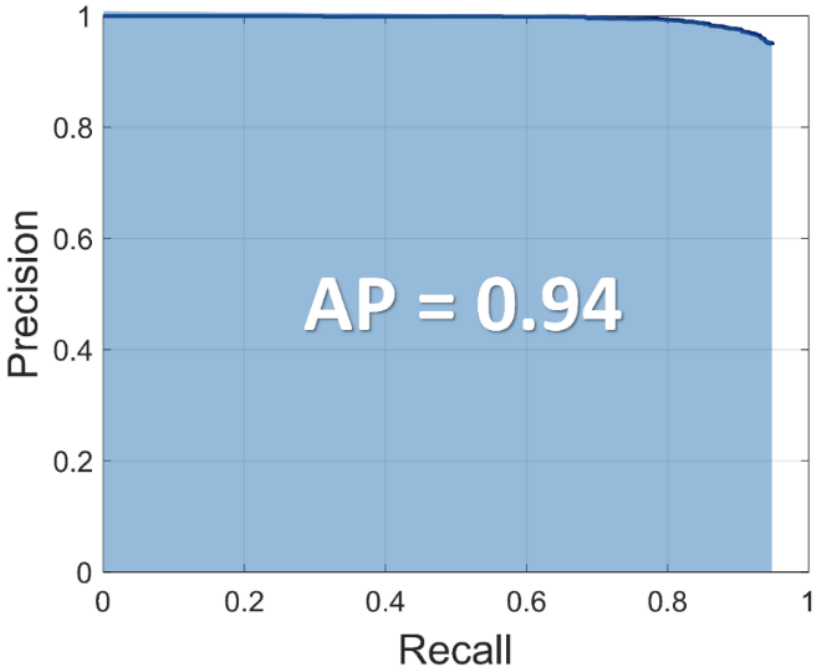

Fig. 9. Average precision (AP) graph at an intersection-over-union (IoU) threshold of 85%.

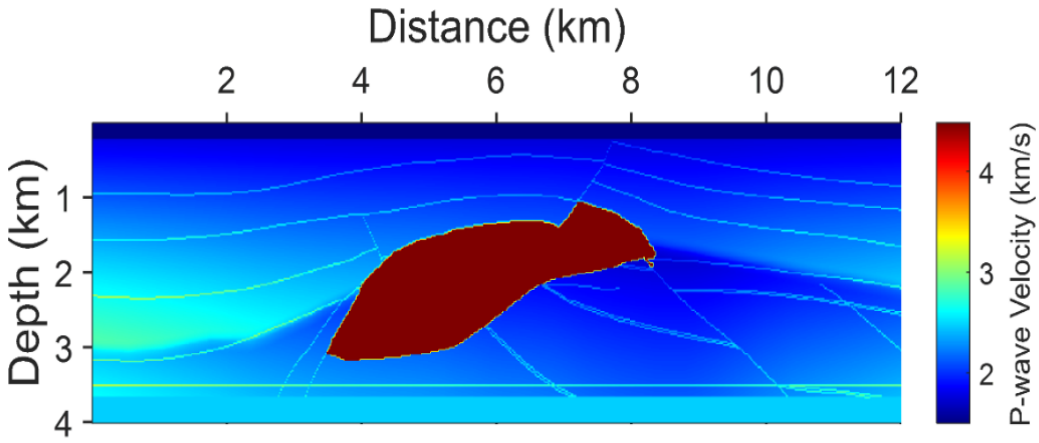

Fig. 10. SEG/EAGE salt dome crossline 4-km-sliced two-dimensional P-wave velocity model for testing.

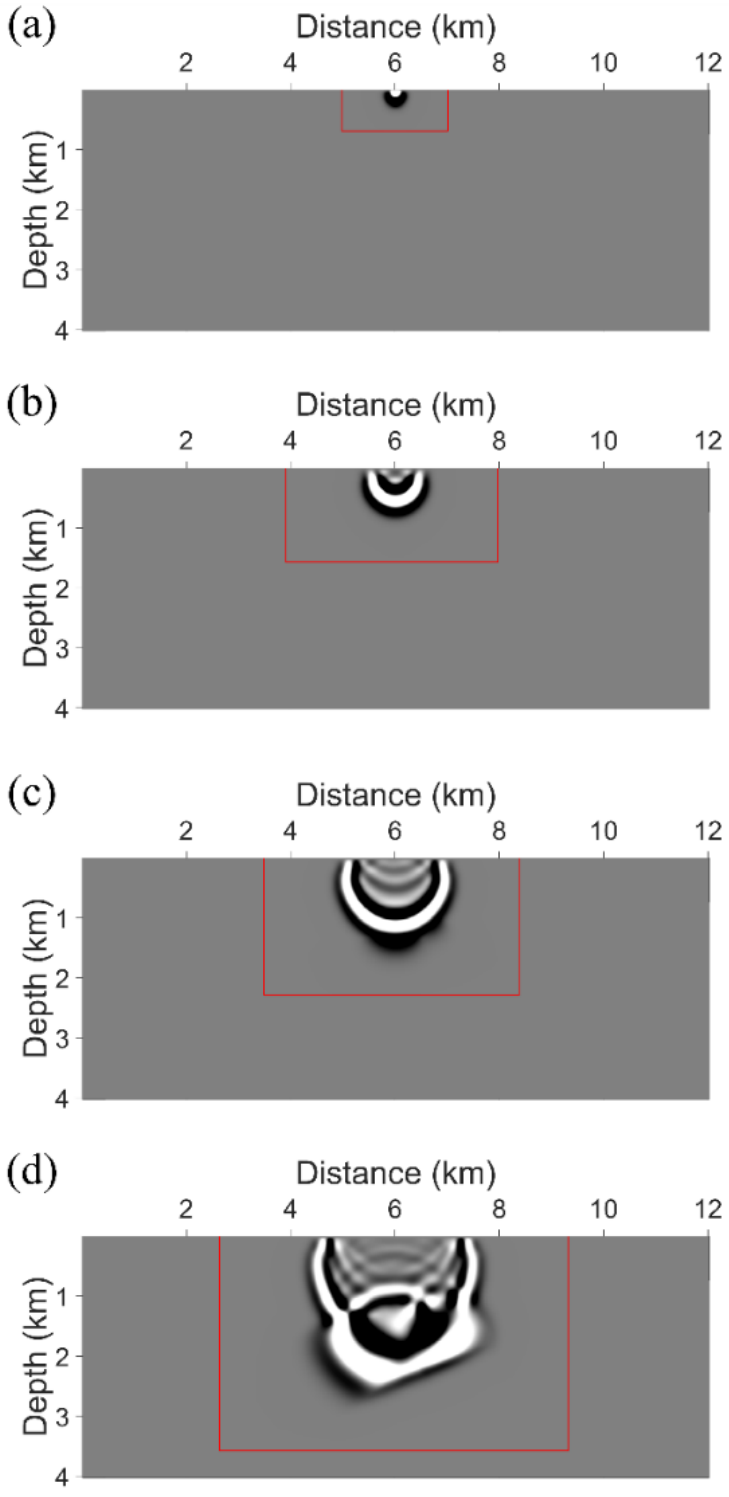

Fig. 11. The wavefield image and detected computational domain by a YOLOv2 object detector according to the following times: (a) 0.3 s, (b) 0.6 s, (c) 0.9 s, (d) 1.2 s.

NUMERICAL EXAMPLES

In the previous section, we observed that in the AC-EDM the computational efficiency decreases as the model size increases. To overcome this limitation, we proposed the YOLOv2-EDM to determine the computational domain with a smaller overhead cost. In this section, accordingly, the YOLOv2-EDM object detector was trained with various wavefield snapshots and validated with new data sets. We named the EDM using the produced detector as YOLOv2-EDM. In this section, the computational efficiencies of AC-EDM and YOLOv2-EDM were compared using various model sizes.

Modeling environment

We performed a numerical experiment using the Marmousi2 p-wave velocity model that was the same as the experiment described in THEORY section (Table 1). The source was located at a distance of 8.5 km and a depth of 20 m below the sea level, and receivers were located on all grids at 20 m below the sea level as acquisition geometry. The source function was a Ricker wavelet with a dominant frequency of 4 Hz, and the recording time was 5.7 s. All numerical examples were run on a workstation with Intel Core i9-10980XE CPU cores at 3.70 GHz and 2 NVIDIA GeForce RTX 3090GPUs. All numerical examples are performed in MATLAB R2021a.

Computational efficiency and accuracy analysis

The modeling environments are the same as those used in the computational efficiency analysis of AC-EDM in THEORY 2. We compared the computational time and the efficiency of the proposed method and AC-EDM in each model. The computational efficiency results are shown in Table 6, and that results for computational efficiency are visually shown in Fig. 12. The results of computational time are the mean of three measurements.

Table 6. The computation time, efficiency, and MAE of four seismic numerical modeling cases.

		Case 1	Case 2	Case 3	Case 4
Reference modeling[1)]		3 min 9 s	25 min 5 s	3 h 20 min 30 s	1 d 2 h 37 min 7 s
AC-EDM[2)]	Computation time	2 min 36 s	23 min 7 s	3 h 25 min 38 s	1 d 4 h 11 min 4 s
	Efficiency	**17.5 %**	7.8 %	- 2.6 %	- 5.9 %
	MAE[4)]	2.53E-5	5.03E-5	1.07E-4	2.34E-4
YOLOv2-EDM[3)]	Computation time	2 min 57 s	19 min 55 s	2 h 32 min 12 s	19 h 55 in 51 s
	Efficiency	6.3 %	**20.6 %**	**24.1 %**	**25.1 %**
	MAE	5.98E-10	5.60E-13	6.88E-14	1.033E-13

1) seismic numerical modeling without the expanding domain method, 2) amplitude comparison expanding domain method, and 3) YOLOv2 object detector expanding domain method (proposed) 4) mean absolute error.

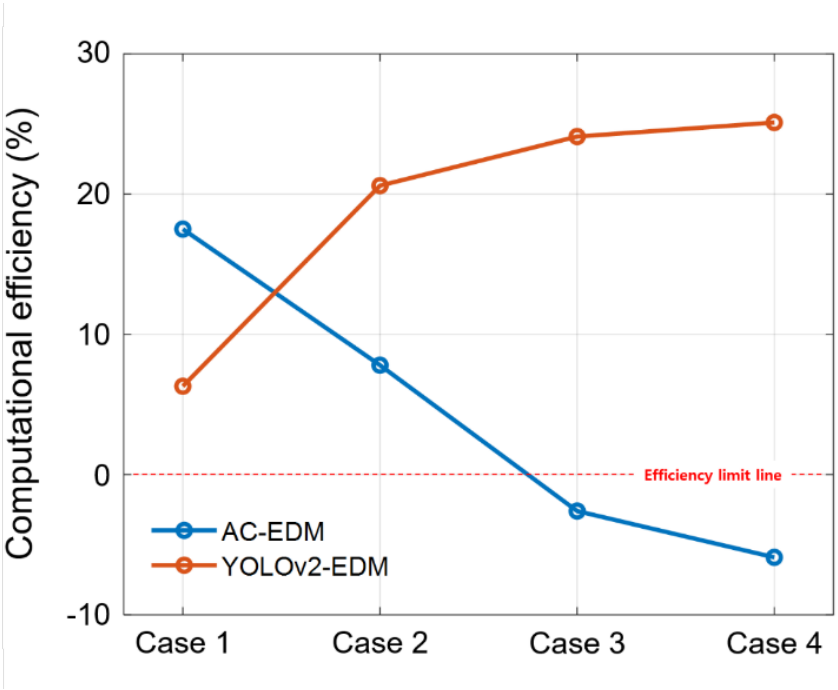

Fig. 12. The computational efficiency graph of AC-EDM and YOLOv2-EDM.

In Case 1, with small model size, the efficiency of the proposed YOLOv2-EDM method was 6.3%, lower than that of AC-EDM. However, In Cases 2-4, with increased model sizes, it was found that the proposed YOLOv2-EDM method showed much better efficiency with computational scaling. Even in Cases 3 and 4, where the efficiency of AC-EDM exceeds the efficiency limit, the YOLOv2-EDM method showed high efficiency. Finally, the modeling accuracy was determined by comparing the seismograms obtained by these two methods (AC-EDM and YOLOv2-EDM) with reference modeling (without EDM). Figs. 13-16 show the seismograms obtained for Cases 1-4 respectively: the reference seismograms are shown in Figs. 13-16 (a); the AC-EDM seismograms are shown in Figs. 13-16 (b); the YOLOv2-EDM seismograms are shown in Figs. 13-16 (c); and the differences between the two EDMs are shown in Figs. 13-16 (d) and Figs. 13-16 (e). As shown in the figures, the errors obtained from both EDMs are too small to be considered (RMSE < 0.001). To check the two EDMs accuracy, we calculate the mean absolute error (MAE) of each EDM and reference seismometers (Table 6).

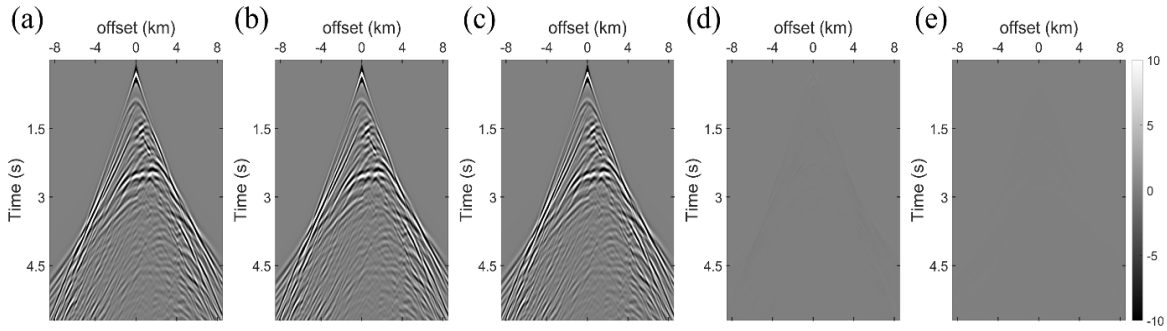

Fig. 13. The seismograms and difference obtained for case 1. (a) Reference seismogram; (b) AC-EDM seismogram; (c) YOLOv2-EDM seismogram (proposed); (d) Difference between AC-EDM and the reference; and (e) Difference between YOLOv2-EDM and the reference.

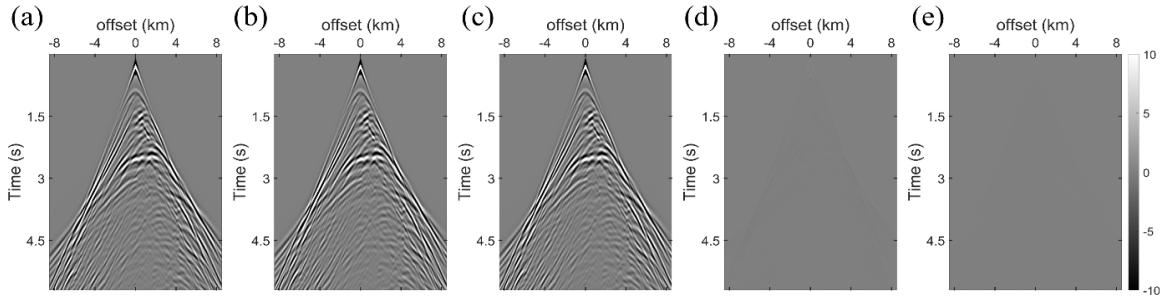

Fig. 14. The seismograms and difference obtained for case 2. (a) Reference seismogram; (b) AC-EDM seismogram; (c) YOLOv2-EDM seismogram (proposed); (d) Difference between AC-EDM and the reference; and (e) Difference between YOLOv2-EDM and the reference.

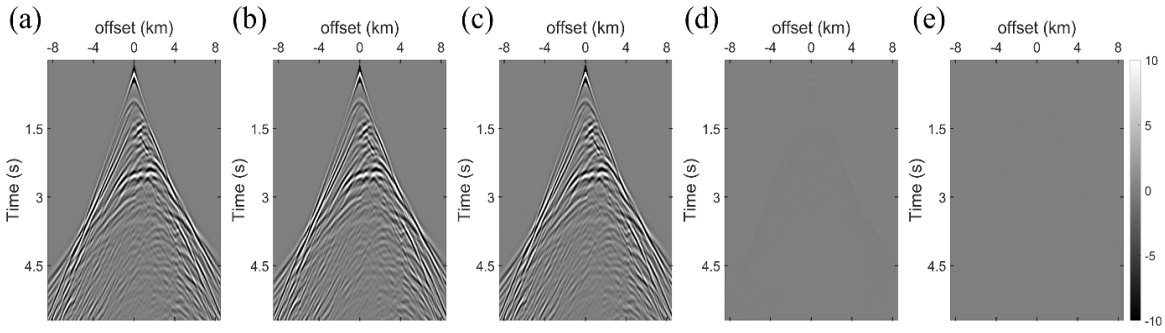

Fig. 15. The seismograms and difference obtained for case 3. (a) Reference seismogram; (b) AC-EDM seismogram; (c) YOLOv2-EDM seismogram (proposed); (d) Difference between AC-EDM and the reference; and (e) Difference between YOLOv2-EDM and the reference.

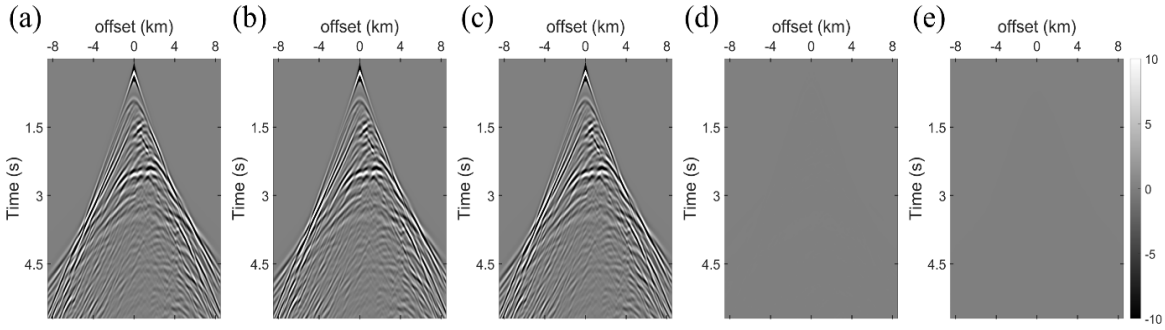

Fig. 16. The seismograms and difference obtained for case 4. (a) Reference seismogram; (b) AC-EDM seismogram; (c) YOLOv2-EDM seismogram (proposed); (d) Difference between AC-EDM and the reference; and (e) Difference between YOLOv2-EDM and the reference.

CONCLUSIONS

We proposed a new seismic modeling technique that combines the EDM and a deep learning-based object detector to improve computational efficiency. The EDM uses an algorithm to obtain computational efficiency by excluding the zero wavefield domain from the computation and avoiding the use of the total modeling domain at the early wave propagation times. The amplitude comparing method is typically utilized for determining the zero domain; however, this algorithm shows a lower efficiency as the computational area becomes larger. Once trained, the data-driven deep learning-based EDM can keep its scalability by minimizing the overhead of the domain identification.

The YOLO v2 object detector is one type of deep learning object detector that identifies the target object's location by separating it from the background and then make a classification. The YOLOv2 object detector is widely used for its real-time image processing capability and in this work the EDM was combined with a trained YOLOv2 object detector in order to determine the computational domain rapidly. In specific, ResNet-50 was integrated to YOLOv2 for feature extraction, and the anchor box properties were selected through the anchor box estimation. After training the deep neural network, the detector test was performed through the AP method. For further validation a numerical modeling test on the other benchmark model, which was not included in the training data was performed.

Finally, we evaluated the performance of YOLOv2-EDM by comparing AC-EDM and YOLOv2-EDM using various model sizes. The experiment was performed by comparing the computational efficiency of the seismic numerical modeling with Marmousi-2 velocity models, each with a different grid size. In the Marmousi2 velocity model with a grid spacing of 10 m, AC-EDM and YOLOv2-EDM show similar computational efficiencies due to the negligible computation gain from the EDM. However, it was observed that YOLOv2-EDM was more efficient than AC-EDM when the finer grids were used in the modeling. Contrary to AC-EDM, YOLOv2-EDM showed a tendency to increase computational efficiency as the number of grids increases. Moreover, according to the experimental results of Cases 3 and 4, the efficiency converged to ~25.1 % computational gains under these modeling parameters. We expect that this technique could be applied to efficient large-scale high-frequency modeling that requires dense grid spacing.

ACKNOWLEDGMENTS

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2021R1I1A3045763) and supported by the Korea Maritime & Ocean University Research Fund in 2021.

REFERENCES

Abdelkhalek, R., Calandra, H., Coulaud, O., Roman, J. and Latu, G., 2009. Fast seismic modeling and reverse time migration on a GPU cluster. 2009 Internat. Conf. High Performance Comput. Simulat.: 36-43.

Alkhalifah, T., 2000. An acoustic wave equation for anisotropic media, Geophysics, 65: 1239-1250.

Aminzadeh, F., Burkhard, N., Nicoletis, L., Rocca, F. and Wyatt, K., 1994. SEG/EAEG 3-D modeling project, 2nd update. The Leading Edge, 13: 949-952.

Barthakur, M. and Sarma, K.K., 2019. Semantic segmentation using K-means clustering and deep learning in satellite image. 22nd Internat. Conf. Innovat. Electronics, Signal Process. Communicat. (IESC): 192-196.
Carcione, J.M., Herman, G.C. and Ten Kroode, A.P.E., 2002. Seismic modelling, Geophysics, 67: 1304-1325.
Carcione, J.M., Kosloff, D. and Kosloff, R., 1988. Viscoacoustic wave propagation simulation in the earth, Geophysics, 53: 769-777.
Cerjan, C., Kosloff, D., Kosloff, R. and Reshef, M., 1985. A nonreflecting boundary condition for discrete acoustic and elastic wave equations, Geophysics, 50: 705-708.
Chassagnon, G., Vakalopolou, M., Paragios, N. and Revel, M.P., 2020. Deep learning: definition and perspectives for thoracic imaging. Eur. Radiol., 30: 2021-2030.
Deng, J., Dong, W., Socher, R., Li, L.J., Li, K. and Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database. 2009 IEEE Conf. Comput. Vision Patt. Recognit., 248-255.
Drossaert, F.H. and Giannopoulos, A., 2007. A nonsplit complex frequency-shifted PML based on recursive integration for FDTD modeling of elastic waves. Geophysics, 72(2): T9-T17.
Etgen, J.T. and O'Brien, M.J., 2007. Computational methods for large-scale 3D acoustic finite-difference modeling: A tutorial. Geophysics, 72(5): SM223-SM230.
Fowler, P.J. and King, R., 2011. Modeling and reverse time migration of orthorhombic pseudo-acoustic P-waves. Expanded Abstr., 81st. Ann. Internat. SEG Mtg., San Antonio: 190-195.
Geiger, A., Lenz, P., Stiller, C. and Urtasun, R., 2013. Vision meets robotics: The kitti dataset. Internat. J. Rob. Res., 32: 1231-1237.
Girshick, R., 2015. Fast R-CNN. Proc. IEEE Internat. Conf. Comput. Vis.: 1440-1448.
Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. Proc. IEEE Conf. Comput. Vis. Patt. Recognit.. 580-587.
Graves, R.W., 1996. Simulating seismic wave propagation in 3D elastic media using staggered-grid finite differences. Bull. Seismol. Soc. Am., 86: 1091-1106.
Ha, W. and Shin, C., 2012. Efficient Laplace-domain modeling and inversion using an axis transformation technique. Geophysics, 77(4): R141-R148.
He, K., Gkioxari, G., Dollár, P. and Girshick, R., 2017. Mask R-CNN. Proc. IEEE Internat. Conf. Comput. Vis. (ICCV): 2961-2969.
He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. Proc. IEEE Conf. Comput. Vis. Patt. Recognit.: 770-778.
Itakura, K. and Hosoi, F., 2020. Automatic tree detection from three-dimensional images reconstructed from 360 spherical camera using YOLOv2. Remote Sens., 12: 988.
Jun, H., Jou, H.T., Kim, C.H., Lee, S.H. and Kim, H.J., 2020. Random noise attenuation of sparker seismic oceanography data with machine learning, Ocean Sci., 16: 1367-1383.
Kim, S., Seol, S.J., Byun, J., Park, J. and Oh, S., 2020. Extraction of diffraction events from seismic data using deep learning-based approach. Expanded Abstr., 90th Ann. Internat. SEG Mtg., Houston: 2840-2844.
Kingma, D.P. and Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
Komatitsch, D. and Tromp, J., 1999. Introduction to the spectral element method for three-dimensional seismic wave propagation, Geophys. J. Internat., 139: 806-822.
Levander, A.R., 1988. Fourth-order finite-difference P-SV seismograms, Geophysics, 53: 1425-1436.
Li, S., Yang, C., Sun, H. and Zhang, H., 2019. Seismic fault detection using an encoder–decoder convolutional neural network with a small training set. J. Geophys. Engineer., 16: 175-189.

Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L., 2014. Microsoft COCO: Common objects in context. In: Fleet, D., Pajdla, T., Schiele, B. and Tuytelaars ,T. (Eds.), European Conf. Comput. Vis., Lecture Notes in Computer Science, Vol. 8693. Springer Verlag, Cham.: 740-755.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y. and Berg, A.C., 2016. SSD: Single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.), European Conf. Comput, Vis.. Springer Verlag, Cham.: 21-37.

Loey, M., Manogaran, G., Taha, M.H.N. and Khalifa, N.E.M., 2021. Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection. Sustain. Cities Soc., 65: 102600.

Martin, G.S., Wiley, R. and Marfurt, K.J., 2006. Marmousi-2: An elastic upgrade for Marmousi. The Leading. Edge, 25: 156-166.

Özdenvar, T. and McMechan, G.A., 1997. Algorithms for staggered-grid computations for poroelastic, elastic, acoustic, and scalar wave equations, Geophys. Prosp., 45: 403-420.

Padilla, R., Netto, S.L. and da Silva, E.A., 2020. A survey on performance metrics for object-detection algorithms, In: Internat. Conf. Systems, Signals Image Process. (IWSSIP): 237-242.

Petrov, I.B. and Khokhlov, N.I., 2014. Modeling 3D seismic problems using high-performance computing systems, Mathemat. Models Comput. Simulat., 6: 342-350.

Pochet, A., Diniz, P.H., Lopes, H. and Gattass, M., 2018. Seismic fault detection using convolutional neural networks trained on synthetic poststacked amplitude maps. IEEE Geosci. Remote Sens. Lett., 16: 352-356.

Ramsden, C., Bennett, G. and Long, A., 2005. High-resolution 3D seismic imaging in practice. The Leading Edge, 24: 423-428.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. IEEE Conf. Comput. Vis. Patt. Recognit. (CVPR): 779-788.

Redmon, J. and Farhadi, A., 2017. YOLO9000: better, faster, stronger. IEEE Conf. Comput. Vis. Patt. Recognit. (CVPR): 7263-7271.

Redmon, J. and Farhadi, A., 2018. YOLOv3: An incremental improvement, arXiv preprint arXiv:1804.02767.

Robertsson, J.O., Blanch, J.O. and Symes, W.W., 1994. Viscoelastic finite-difference modelling, Geophysics, 59: 1444-1456.

Ryu, D., Kim, A. and Ha, W., 2015. Expanding domain method for 3D time-Laplace-domain hybrid modelling, Geosystem Eng., 18 (5), 259–265.

Sadak, F., Saadat, M. and Hajiyavand, A.M., 2020. Real-time deep learning-based image recognition for applications in automated positioning and injection of biological cells, Comput. Biol. Med., 125: 103976.

Santos, J.E., Douglas Jr., J., Morley, M.E. and Lovera, O.M., 1988. Finite element methods for a model for full waveform acoustic logging, IMA J. Numer. Anal., 8: 415-433.

Siahkoohi, A., Louboutin, M. and Herrmann, F.J., 2019. Neural network augmented wave-equation simulation. arXiv preprint arXiv:1910.00925.

Suh, S. and Wang, B., 2011. Expanding domain methods in GPU based TTI reverse time migration. Expanded Abstr., 81st. Ann. Internat. SEG Mtg., San Antonio: 3460-3464.

Szegedy, C., Toshev, A. and Erhan, D., 2013. Deep neural networks for object detection. Advan. Neural Informat. Process. Syst., 1-9.

Tang, X. and Shao, Q., 2013. Numerical simulation on seismic liquefaction by adaptive mesh refinement due to two recovered fields in error estimation. Soil Dynam. Earthq. Engineer., 49: 109-121.

Vidale, J.E., 1990. Finite-difference calculation of traveltimes in three dimensions. Geophysics, 55: 521-526.

Virieux, J., 1986. P-SV wave propagation in heterogeneous media: Velocity-stress finite-difference method. Geophysics, 51: 889-901.
Vrolijk, J.W. and Blacquière, G., 2021. Source deghosting of coarsely-sampled common-receiver data using a convolutional neural network. Geophysics, 86(3): V185-V196.
Wang, B., Zhang, N., Lu, W., and Wang, J., 2019. Deep-learning-based seismic data interpolation: A preliminary result. Geophysics, 84(1), V11-V20
Womg, A., Shafiee, M.J., Li, F. and Chwyl, B., 2018. Tiny SSD: A tiny single-shot detection deep convolutional neural network for real-time embedded object detection, 2018 15th Conference on Computer and Robot Vision (CRV), 95-101.
Yang, F. and Ma, J., 2019. Deep-learning inversion: A next-generation seismic velocity model building method. Geophysics, 84(4), R583-R599.