Computers and Fluids 205 (2020) 104582

Contents lists available at ScienceDirect

Computers and Fluids

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

On moving mesh WENO schemes with characteristic boundary conditions for Hamilton-Jacobi equations

Yue Li[a], Juan Cheng[b,c,1], Yinhua Xia[d,2], Chi-Wang Shu[e,3,*]

[a] *Graduate School, China Academy of Engineering Physics, Beijing 100088, China*
[b] *Laboratory of Computational Physics, Institute of Applied Physics and Computational Mathematics, Beijing 100088, China*
[c] *Center for Applied Physics and Technology, Peking University, Beijing 100871, China*
[d] *School of Mathematics Sciences, University of Science and Technology of China, Hefei, Anhui 230026, PR China*
[e] *Division of Applied Mathematics, Brown University, Providence, RI 02912, USA*

ARTICLE INFO

Article history:
Received 1 October 2019
Revised 20 March 2020
Accepted 11 May 2020
Available online 14 May 2020

Keywords:
Hamilton-Jacobi equation
Moving mesh ALE method
High order WENO finite difference method
Boundary conditions

ABSTRACT

In this paper, we are concerned with the study of efficient and high order accurate numerical methods for solving Hamilton-Jacobi (HJ) equations with initial conditions defined in the whole domain. One of the commonly used strategy is to solve the problem only in a finite domain, but the determination of boundary conditions at the artificial boundary of the finite computational domain is a problem. If the initial condition decays fast in space, one could use zero boundary condition at the artificial boundary if the domain is large enough, but this may not be very efficient since the computational domain may need to be very large to justify this choice. In this paper we use the high order moving mesh arbitrary Lagrangian Eulerian (ALE) weighted essentially non-oscillatory (WENO) finite difference scheme, recently developed in [13], in a finite and moving computational domain, with numerical boundary conditions obtained by solving the characteristic ordinary differential equations (ODEs) along the artificial boundary of the moving computational domain. The usage of this moving characteristic boundary conditions allows us to solve the HJ equations in any initial finite domain that we are interested in, regardless of the magnitude of the initial condition at the artificial domain boundary. This method works well when singularities do not appear at the artificial boundary. Extensive numerical tests in one and two dimensions are given to demonstrate the flexibility and efficiency of our method in solving both smooth problems and problems with corner singularities.

1. Introduction

The Hamilton-Jacobi equations have wide applications in optimal control, image processing, and mechanics, among many others. In nonlinear solid mechanics, the Hamilton-Jacobi equations have their spatial variables defined over non-periodic unbounded or semi-unbounded domains [12]. Such problems with unbounded domains also appear in other applications.

In this paper, we are interested in solving the Hamilton-Jacobi (HJ) equations

$$\begin{cases}\phi_t + H(x, \phi_x, \phi, t) = 0 \\ \phi(x, 0) = \phi_0(x)\end{cases} \tag{1}$$

in one space dimension, and

$$\begin{cases}\phi_t + H(x, y, \phi_x, \phi_y, \phi, t) = 0 \\ \phi(x, y, 0) = \phi_0(x, y)\end{cases} \tag{2}$$

in two space dimensions, for which the initial condition ϕ_0 is defined over the whole domain R or R^2. One major difficulty in solving such problems is the infinite computational domain. For general partial differential equations (PDEs) defined over infinite domain, there are different approaches in solving them, for example the infinite element method and the boundary element method [2,7], spectral methods in using basis functions defined in the infinite domain [3], and the region decomposition algorithm based

---

* Corresponding author.
E-mail addresses: liyue9443@outlook.com (Y. Li), cheng_juan@iapcm.ac.cn (J. Cheng), yhxia@ustc.edu.cn (Y. Xia), chi-wang_shu@brown.edu (C.-W. Shu).

1 Research is supported in part by Science Challenge Project. No. TZ2016002, NSFC grants 11871111 and U1630247.
2 Research is supported in part by NSFC grant 11871449.
3 Research is supported in part by AFOSR grant FA9550-20-1-0055 and NSF grant DMS-1719410.

on natural boundary naturalization [5]. Boundary element method mainly starts with Green function and Green formula, the boundary value problem of partial differential equation is transformed into a strongly singular integral equation on the boundary. Another commonly used strategy is to solve the problem only in a finite domain. This has the advantage that most numerical methods, such as finite difference and finite element methods, can be directly applied. However, the determination of boundary conditions at the artificial boundary of the finite computational domain is a problem. If the initial condition ϕ_0 decays fast in space towards a constant (e.g. zero), one could use zero boundary condition at the artificial boundary if the domain is large enough. This, however, may not lead to the most efficient numerical method, since the computational domain may need to be very large in order to justify the choice of zero boundary condition. Moreover, if the initial condition does not go to zero (or constant) at infinity, then it is difficult to cut the initial computational domain to a finite domain, no matter how large, and figure out suitable boundary conditions at the artificial boundary, if the finite computational domain remains fixed in time. Engquist and Majda developed a systematic method for obtaining a hierarchy of local boundary conditions at these artificial boundaries for wave equations [6], see also [8,9]. It guarantees stable difference approximations and also minimizes the (unphysical) artificial reflections which occur at the boundaries. Notice that the determination of boundary conditions at the artificial boundary of the finite computational domain is very difficult, if not impossible, based solely on the initial condition inside the computational domain and the PDE, especially for the inflow boundary, because the boundary values of the exact solution at an inflow boundary depends on the initial condition outside the initial finite computational domain. It is however difficult, in many cases, to have the artificial boundary of the finite computational domain to be only outflow boundary, especially if the computational domain is fixed in time.

For the HJ Eqs. (1) and (2) that we are interested in this paper, one possible remedy of this difficulty is to use a moving computational domain, whose boundary corresponds to the characteristic curves of the HJ equations. This would ensure that the boundary of the computational domain is neither an inflow nor an outflow, but is a characteristic boundary. That is, the information on this moving characteristic boundary can be completely determined by the values of the initial condition at the boundary of the initial computational boundary. The numerical boundary conditions can be obtained by solving the characteristic ordinary differential equations (ODEs) along the artificial boundary of this moving computational domain. This method works well when singularities do not appear at the artificial boundary. In order to apply this framework, we need a stable and accurate numerical method which can be applied to such moving computational domains. The high order moving mesh arbitrary Lagrangian Eulerian (ALE) weighted essentially non-oscillatory (WENO) finite difference scheme, recently developed in [13], is a good choice and will be used in this paper. In two dimensions, this method is based on moving quadrilateral meshes, which are often used in Lagrangian type methods. We will only give a very brief description of this method in Section 2, in order to describe our algorithm, and will refer the readers to [13] and the references therein for more details and for the history of numerical methods for solving HJ equations.

This paper is organized as follows. In Section 2, we give a detailed discussion on our algorithm for solving the HJ equations, paying special attention to the procedure of obtaining the characteristic boundary conditions numerically in one and two dimensional cases. In Section 3, the numerical results solving several typical HJ equations are presented, to demonstrate the good performance of our algorithm. Finally, we will give concluding remarks in Section 4.

2. Algorithm

2.1. Characteristic boundary conditions

The basic idea behind the method of characteristics is to recast (appropriate types of) the PDEs as a system of ordinary differential equations (ODEs). For the type of HJ equation under investigation here, the method amounts to finding curves, which are called the characteristics, along which the solution ϕ can be computed from the giving initial condition. This is achieved by integrating a system of ODEs. Demidov has a historical perspective on this method in [4]. We compute our unknown boundary conditions along a moving computational domain by solving the characteristic ODEs. Once the numerical boundary values are obtained in this fashion, we will use the high order finite difference ALE-WENO scheme developed in [13] to solve the PDE inside the computational domain. We now describe the procedure to obtain and discretize the characteristic ODEs.

For the one dimensional HJ Eq. (1), we begin by writing it into the form

$$\begin{cases} F(p,r,\phi,x,t) = r + H(x,p,\phi,t) = 0 \\ \phi_0 = \phi(x_0,0) \end{cases}$$

with the parameterization

$$\begin{cases} p(t) = \frac{\partial \phi}{\partial x}(x(t),t), \\ r(t) = \frac{\partial \phi}{\partial t}(x(t),t), \\ \varphi(t) = \phi(x(t),t) \end{cases} \tag{3}$$

along the characteristic curve $(x(t), t)$, with an initial condition $x_0 = x(0)$. Using the HJ Eq. (1), we can obtain the characteristic curve initiating from x_0 by the system of ODEs

$$\begin{cases} \frac{dx}{dt} = \frac{\partial H}{\partial p}, \\ \frac{dp}{dt} = -\frac{\partial H}{\partial x} - \frac{\partial H}{\partial \phi} p, \\ \frac{dr}{dt} = -r\frac{\partial H}{\partial \phi}, \\ \frac{d\varphi}{dt} = p\frac{\partial H}{\partial p} + r, \end{cases} \tag{4}$$

together with the compatibility equations

$$\begin{cases} F(p_0,r_0,\phi_0,x_0,0) = r_0 + H(x_0,p_0,\phi_0,0) = 0 \\ p_0 = \frac{\partial \phi}{\partial x}(x_0,0) \end{cases} \tag{5}$$

that define $p_0 = p(0),\ r_0 = r(0)$.

Notice that the characteristic Eq. (4) form a coupled ODE system. We can solve this system along the two boundary points of the initial computational domain, thus obtaining the moving computational domain and its boundary conditions for the later time.

Similarly, for the two dimensional HJ Eq. (2), we also begin by writing the HJ equation into the form

$$\begin{cases} F(p_1,p_2,r,\phi,x,y,t) = r + H(x,y,p_1,p_2,\phi,t) = 0, \\ \phi_0 = \phi(x_0,y_0,0), \end{cases}$$

with the parameterization,

$$\begin{cases} p_1(t) = \frac{\partial \phi}{\partial x}(x(t),y(t),t), \\ p_2(t) = \frac{\partial \phi}{\partial y}(x(t),y(t),t), \\ r(t) = \frac{\partial \phi}{\partial t}(x(t),y(t),t), \\ \varphi(t) = \phi(x(t),y(t),t), \end{cases} \tag{6}$$

along the characteristic curve $(x(t), y(t), t)$, with the initial condition $x_0 = x(0),\ y_0 = y(0)$. Now the characteristic curve initiating from (x_0, y_0) is defined by the system of ODEs

$$\begin{cases} \frac{dx}{dt} = \frac{\partial H}{\partial p_1}, \\ \frac{dy}{dt} = \frac{\partial H}{\partial p_2}, \\ \frac{dp_1}{dt} = -\frac{\partial H}{\partial x} - \frac{\partial H}{\partial \phi} p_1, \\ \frac{dp_2}{dt} = -\frac{\partial H}{\partial y} - \frac{\partial H}{\partial \phi} p_2, \\ \frac{dr}{dt} = -r\frac{\partial H}{\partial \phi}, \\ \frac{d\varphi}{dt} = p_1 \frac{\partial H}{\partial p_1} + p_2 \frac{\partial H}{\partial p_2} + r, \end{cases} \tag{7}$$

together with the compatibility equations

$$\begin{cases} F(p_0, r_0, \phi_0, x_0, y_0, 0) = r_0 + H(x_0, y_0, p_0, \phi_0, 0) = 0, \\ p_{10} = \frac{\partial \phi}{\partial x}(x_0, y_0, 0), \\ p_{20} = \frac{\partial \phi}{\partial y}(x_0, y_0, 0) \end{cases} \tag{8}$$

that define $p_{10} = p_1(0),\ p_{20} = p_2(0),\ r_0 = r(0)$. We can then solve these characteristic ODEs along the boundaries of the initial computational domain to update the x, y, φ, thus obtaining the moving computational domain and its boundary conditions for the later time. We can find that the grid point x, y and the value φ (i.e. ϕ) in this point depend on p_1, p_2, r. We expect to get third order accuracy, thus we use the third order Runge-Kutta method to solve these characteristic ODEs.

In order to match the calculation inside the computational domain which will be introduced later, we take the same ODE solver as that for the time discretization of the PDE solver inside the computational domain, namely, the following third order SSP Runge-Kutta method [14],

$$\begin{aligned} \phi^{(1)} &= \phi^n + \Delta t L(\phi^n, t^n) \\ \phi^{(2)} &= \frac{3}{4}\phi^n + \frac{1}{4}(\phi^{(1)} + \Delta t L(\phi^{(1)}, t^n + \Delta t)) \\ \phi^{n+1} &= \frac{1}{3}\phi^n + \frac{2}{3}\left(\phi^{(2)} + \Delta t L\left(\phi^{(2)}, t^n + \frac{1}{2}\Delta t\right)\right) \end{aligned} \tag{9}$$

for solving the ODE $\phi_t = L(\phi, t)$, where L is the related derivative operator.

This procedure can be used if the singularities of the solution do not reach the boundary of the computational domain. That means the forward going characteristics along the boundary of the computational domain do not intersect with each other, nor do they intersect with any characteristics from inside the computational domain.

2.2. The ALE-WENO scheme inside the computational domain

Once we have determined the boundary of the moving computational region, we need a high order numerical method to compute the solution inside the this moving computational domain. The method should work well for moving meshes, and should not require too stringent smoothness for the mesh movements. The high order finite difference ALE-WENO scheme developed in [13] to solve the HJ equation is a good choice for this purpose, as it can achieve high order accuracy with only boundedness and Lipschitz continuity requirements on the moving meshes. Firstly, we will give a brief discussion on the mesh movement here.

We will introduce a variable ω_i to describe the moving speed of the node x_i in one-dimension and a variable $\omega_{i,j} = (\omega_{x_{i,j}}, \omega_{y_{i,j}})$ to describe the moving speed of the node $(x_{i,j}, y_{i,j})$ in two-dimensions, from the time level n (denoted as t_n) to $n+1$ (denoted as t_{n+1}). We assume that the family of the mesh $\{T_n, n = 0\ldots, L\}$ at all the time levels gives the same mesh topology, i.e., the mesh T_{n+1} has the same number of nodes and the same connectivity as T_n. Under such restriction, we can set up a moving mesh connecting the node at the time level n and the corresponding node at the time level n + 1 linearly. This linear mapping between T_n and T_{n+1} is the crucial ingredient for the ALE-DG method in [11] to maintain stability and accuracy while allowing only very mild regularity of the mesh movement function (Lipschitz continuity is enough), and it is the framework adopted by the high order finite difference ALE-WENO scheme for solving HJ equations developed in [13] as well.

In one dimension, the mesh velocity ω_i^n and the node $x_i(t)$ are defined as

$$\omega_i^n := \frac{x_i^{n+1} - x_i^n}{\Delta t}, \qquad x_i(t) := x_i^n + \omega_i^n (t - t_n), \quad t \in [t_n, t_{n+1}]. \tag{10}$$

Similarly, in two dimensions, we have the definition as,

$$\begin{aligned} \omega_{x_{i,j}}^n &:= \frac{x_{i,j}^{n+1} - x_{i,j}^n}{\Delta t}, \qquad x_{i,j}(t) := x_{i,j}^n + \omega_{x_{i,j}}^n (t - t_n), \ \ t \in [t_n, t_{n+1}], \\ \omega_{y_{i,j}}^n &:= \frac{y_{i,j}^{n+1} - y_{i,j}^n}{\Delta t}, \qquad y_{i,j}(t) := y_{i,j}^n + \omega_{y_{i,j}}^n (t - t_n), \ \ t \in [t_n, t_{n+1}]. \end{aligned} \tag{11}$$

For the one dimensional HJ Eq. (1), its semi-discrete scheme is given by

$$\frac{\partial}{\partial t}\phi_i(t) + \hat{H}(x, \phi_{x_i}^-, \phi_{x_i}^+; \phi_i, t) = 0 \tag{12}$$

where $\phi_i(t)$ is the numerical approximation to the exact solution at the mesh point $\phi(x_i(t), t)$, $\phi_{x_i}^-$ and $\phi_{x_i}^+$ are the left-biased and right-biased WENO approximations to the derivative of ϕ at the node i, which are obtained from the interpolation polynomials with stencils biased to the left and to the right respectively, in a WENO fashion, please see [13] for more details. $\hat{H}$ is a monotone numerical Hamiltonian, which is monotonically non-decreasing in the second argument and monotonically non-increasing in the third argument. The method in [13] combines the solution evolution and the mesh movement into one step. We will use the simple Lax-Friedrichs numerical Hamiltonian as our first order monotone numerical Hamiltonian, suitably taking the mesh movement into account:

$$\begin{aligned} \hat{H}(x_i, u_i^-, u_i^+; \phi_i, t) = H\left(x_i, \frac{u_i^- + u_i^+}{2}, \phi_i, t\right) - \frac{1}{2}\omega_i(u_i^- + u_i^+) \\ -\frac{1}{2}\alpha_i(u_i^+ - u_i^-) \end{aligned} \tag{13}$$

where

$$\begin{aligned} \alpha_i &= \max_{a \le u \le b}\{|H_u(x, u, \phi, t) - \omega_i|\}, \\ a &= \min\{u_i^-, u_i^+\}, \ \ b = \max\{u_i^-, u_i^+\}, \\ \alpha &= \max_i\{\alpha_i\}. \end{aligned} \tag{14}$$

Here H_u denotes the partial derivative of $H(x, u, \phi, t)$ with respect to u, and ω_i is defined by (10).

For the two dimensional HJ Eq. (2), its semi-discrete scheme is given by

$$\begin{aligned} &\frac{\partial}{\partial t}\phi_{i,j}(t) \\ &+\hat{H}(x_{i,j}, y_{i,j}, (\nabla\phi_{i,j})_1, (\nabla\phi_{i,j})_2, (\nabla\phi_{i,j})_3, (\nabla\phi_{i,j})_4; \phi_{i,j}, t) = 0 \end{aligned} \tag{15}$$

where $\phi_{i,j}(t)$ is the numerical approximation to the exact solution at the mesh point $\phi(x_i(t), y_j(t), t)$, $(\nabla\phi_{i,j})_\ell = (\phi_{x_{i,j}}, \phi_{y_{i,j}})_\ell$, $\ell = 1, 2, 3, 4$, are the WENO approximations to the derivatives of ϕ at the node $(x_{i,j}, y_{i,j})$ obtained from the interpolation polynomials with stencils biased to the four quadrants, again please see [13] for more details. Just as in the one dimensional case, $\hat{H}$ is a monotone Hamiltonian, which we take to be the Lax-Friedrichs

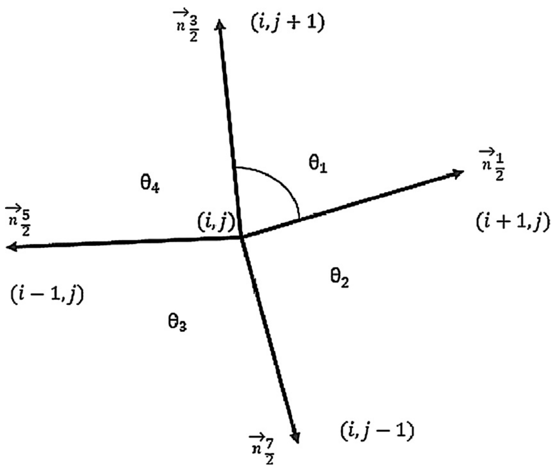

Fig. 1. The node $(x_{i,j}, y_{i,j})$ and its angular sectors.

type monotone Hamiltonian on unstructured meshes developed by Abgrall in [1]. For our moving mesh case, the Lax-Friedrichs monotone Hamiltonian is defined as

$$\hat{H}(x_{i,j}, y_{i,j}, (\nabla\phi_{i,j})_1, (\nabla\phi_{i,j})_2, (\nabla\phi_{i,j})_3, (\nabla\phi_{i,j})_4, \phi_{i,j}, t)$$
$$= H(x_{i,j}, y_{i,j}, u_{i,j}, v_{i,j}; \phi_{i,j}, t) - \omega_{x_{i,j}} u_{i,j} - \omega_{y_{i,j}} v_{i,j} - D(\phi_{x_{i,j}}, \phi_{y_{i,j}}) \tag{16}$$

where

$$u_{i,j} = \frac{\sum_{\ell=1}^{4}\theta_\ell(\phi_{x_{i,j}})_\ell}{2\pi}, \qquad v_{i,j} = \frac{\sum_{\ell=1}^{4}\theta_\ell(\phi_{y_{i,j}})_\ell}{2\pi},$$

$$D(\phi_{x_{i,j}}, \phi_{y_{i,j}}) = \frac{\alpha_{i,j}}{4}\sum_{\ell=1}^{4}\beta_{\ell+\frac{1}{2}}\left(\frac{(\nabla\phi_{i,j})_\ell + (\nabla\phi_{i,j})_{\ell+1}}{2}\right)\cdot\vec{n}_{\ell+\frac{1}{2}}, \tag{17}$$

$$\beta_{\ell+\frac{1}{2}} = \tan\left(\frac{\theta_\ell}{2}\right) + \tan\left(\frac{\theta_{\ell+1}}{2}\right), \tag{18}$$

$$\alpha_{i,j} = \max_{a\le u\le b, c\le v\le d}\{|H_u(x,y,u,v,\phi,t) - \omega_{x_{i,j}}|, |H_v(x,y,u,v,\phi,t) - \omega_{y_{i,j}}|\},$$
$$\alpha = \max_{i,j}\{\alpha_{i,j}\}. \tag{19}$$

θ_ℓ, for $\ell = 1,\ldots,4$, are the angles between two neighboring edges connecting the node $(i,\ j)$, and $\vec{n}_{\ell+\frac{1}{2}}$, for $\ell = 1,\ldots,4$, are the unit vectors along the edges passing through the node $(i,\ j)$ (See Fig. 1). $a = \min\{(\phi_{x_{i,j}})_1, (\phi_{x_{i,j}})_2, (\phi_{x_{i,j}})_3, (\phi_{x_{i,j}})_4\}$, $b = \max\{(\phi_{x_{i,j}})_1, (\phi_{x_{i,j}})_2, (\phi_{x_{i,j}})_3, (\phi_{x_{i,j}})_4\}$; $c = \min\{(\phi_{y_{i,j}})_1, (\phi_{y_{i,j}})_2, (\phi_{y_{i,j}})_3, (\phi_{y_{i,j}})_4\}$, $d = \max\{(\phi_{y_{i,j}})_1, (\phi_{y_{i,j}})_2, (\phi_{y_{i,j}})_3, (\phi_{y_{i,j}})_4\}$.

Following the above spacial discretization, we will use the third order SSP Runge-Kutta method (9) for the time discretization of the HJ Eqs. (1) and (2) respectively, where the operator L in the one dimensional scheme represents $-\hat{H}(\phi_{x_i^-}, \phi_{x_i^-}, x_i, t)$, and in the two dimensional scheme represents $-\hat{H}((\nabla\phi_{i,j})_1, (\nabla\phi_{i,j})_2, (\nabla\phi_{i,j})_3, (\nabla\phi_{i,j})_4, x_{i,j}, y_{i,j}, t)$.

2.3. The time step and evolution of the computational domain

In order to determine the computational domain at the next time step, we need to first choose a suitable time step Δt. While there is no stability issue for solving the characteristic ODEs at the boundary of the computational domain, there is a CFL condition for solving the PDE inside this domain, which is given by

$$\Delta t = \frac{c_1 h}{\alpha} \tag{20}$$

where c_1 is the CFL number, here we choose it as 0.6. The coefficient α is computed by (14) or (19) in the one dimensional or two dimensional cases respectively. In the one dimensional case, h is the minimum size of all the cells in the computational domain, and in the two dimensional case, h is the minimum diameter of the inscribed circles for all the quadrilateral cells in the domain. In the situation of the mesh moving along characteristics, α in (20) could be very small, leading to a very large Δt determined by the stability constraint (20). In such cases, it is prudent to reduce Δt in order to ensure temporal accuracy. In our numerical tests of characteristic type moving mesh, when Δt determined by (20) is larger than $5h$, we will set it to be $5h$.

In the temporal discretization, in order to ensure the accuracy of the algorithm on a moving mesh, as suggested in [11], the mesh movement speed should satisfy the following boundedness and Lipschitz continuity properties,

$$|\omega_i| \le c_2, \quad |\omega_i'| = \left|\frac{\omega_{i+1} - \omega_i}{x_{i+1} - x_i}\right| \le c_3. \tag{21}$$

In the following one- and two- dimensional numerical examples, we use $c_2 = 10$ and $c_3 = 10$ to enforce these conditions.

Generally we can easily choose the mesh motion which can guarantee the condition (21). For the case that the mesh moves along the characteristics, and the case that the mesh is perturbed randomly from the characteristic-type moving mesh, in order to make the mesh velocity satisfy the condition (21), we need to make some modification to the mesh moving speed if necessary. We refer to [13] for more details of these modifications and will not repeat them here.

Once the time step is determined, we will first solve the characteristic ODE (4) or (7) for the mesh points at the boundary and at the necessary ghost points (needed for the high order WENO scheme) near and outside the boundary of the computational domain, to determine the values of the numerical solution at the boundary and for these ghost points. We will use the third order SSP Runge-Kutta method (9) to determine the boundary condition of the computational domain. We will then determine the movement of the mesh points inside the computational domain, and then apply the above described third order ALE-WENO finite difference scheme with the third order SSP Runge-Kutta time discretization.

3. Numerical examples

In all the numerical examples of this section, we use the fully discrete third order ALE-WENO scheme with a characteristically moving computational domain, with the boundary conditions and the point values at the necessary ghost points obtained from solving the characteristic ODEs. Two different types of moving meshes inside the computational domain are used: the mesh with mesh points moving along the characteristic directions, subject to adjustments to satisfy the boundedness of mesh movements (21) (to be denoted by "characteristic-type moving mesh"), and the mesh with its points randomly perturbed (up to 12.5%) from the characteristic-type moving mesh (to be denoted by "randomly moving mesh"). Some of the examples in this section have been taken from Li et al. [13], however in [13] periodic boundary conditions were used, and here we use different computational domain not corresponding to a period to test our characteristic type boundary conditions.

Example 3.1. We solve the one-dimensional linear equation with variable coefficient

$$\phi_t + \sin(x)\phi_x = 0, \tag{22}$$

by the characteristic boundary condition, that is, the initial computational domain is $x \in [-1, 2]$ and it is evolved in time by char-

Table 1
Numerical errors and orders of accuracy at $t = 1$ for Example 3.1 with the initial condition (23) on the randomly moving mesh with N mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	1.75E-04		1.98E-04		3.97E-04	
64	1.60E-05	3.45	2.42E-05	3.03	1.08E-04	1.88
128	2.71E-06	2.56	6.83E-06	1.83	3.56E-05	1.60
256	4.12E-07	2.72	7.23E-07	3.24	4.69E-06	2.92
512	6.63E-08	2.63	9.65E-08	2.91	5.87E-07	3.00

Table 2
Numerical errors and orders of accuracy at $t = 1$ for Example 3.1 with the initial condition (23) on the characteristic-type moving mesh with $\omega(x) = \sin x$ and with N mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.74E-04		1.63E-04		2.77E-04	
64	9.71E-06	4.82	6.05E-06	4.75	1.18E-05	4.55
128	1.07E-06	3.18	4.90E-07	3.63	4.64E-07	4.67
256	1.31E-07	3.03	5.96E-08	3.04	5.26E-08	3.14
512	1.63E-08	3.00	7.53E-09	2.99	6.74E-09	2.96

acteristics. This is a linear variable coefficient equation, we use it to verify the third-order accuracy of our algorithm for smooth solutions.

First, we choose the initial condition as

$$\phi(x,0) = \sin(x). \tag{23}$$

The exact solution for this problem is

$$\phi(x,t) = \sin\left(2\tan^{-1}\left(e^{-t}\tan\left(\frac{x}{2}\right)\right)\right).$$

We compute the equation up to the time $t = 1$. The numerical errors and orders of accuracy are shown in Tables 1–2 for both types of meshes. The L_2 errors versus the number of grid points are plotted in Fig. 2. The time history of the error at the middle point versus time, for $N = 256$ grid points and both types of mesh movements, is plotted in Fig. 3. It shows that the error does not grow fast in time after an initial transit, especially for the characteristic-type mesh movements.

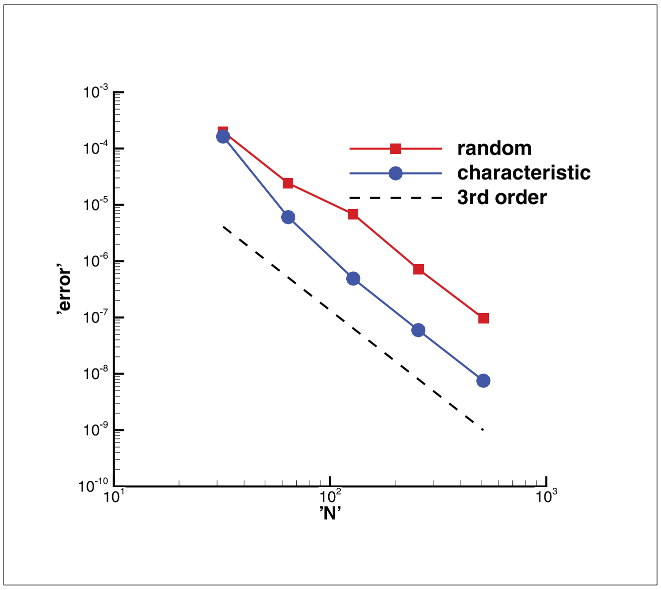

Fig. 2. L_2 errors versus the number of grid points. The characteristic-type moving mesh and the randomly moving mesh for Example 3.1 with the initial condition (23). In this and later error plots a dashed line with slope -3 is plotted to compare the results with the designed third order accuracy.

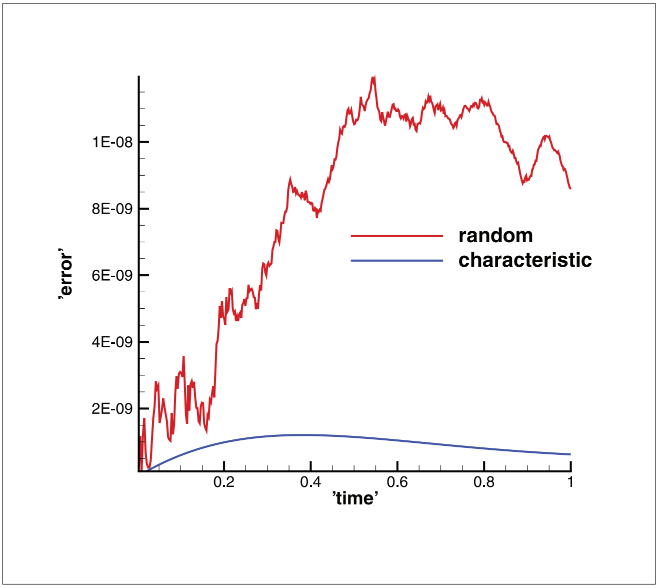

Fig. 3. The error of the middle point in the computational domain versus time, $N = 256$ grid points. The characteristic-type moving mesh and the randomly moving mesh for Example 3.1 with the initial condition (23).

From the error tables we can see the scheme with each type of mesh motions has achieved the expected third order accuracy. From the error tables and figures it is obvious that the magnitude of the error on the characteristic-type moving mesh is much smaller than that on the randomly moving mesh with the same number of mesh nodes.

Next, we use the initial condition

$$\phi(x,0) = e^{-\frac{x^2}{2}} + \frac{1}{2}e^{-\frac{(x-1)^2}{4}} \tag{24}$$

and compute the equation up to the time $t = 1$ using the characteristic-type moving mesh. The exact solution for this problem is

$$\phi(x,t) = e^{-2\tan^{-1}\left(e^{-t}\tan\left(\frac{x}{2}\right)\right)} + e^{-\frac{\left(2\tan^{-1}e^{-t}\tan\left(\frac{x}{2}\right)-1\right)^2}{2}}.$$

The numerical errors and orders of accuracy are shown in Table 3. We can find that even though we compute in a very small region, with the exact solution far from zero at the boundary of the computational domain, we can still get a very small error and the expected third order accuracy. If we would like to use the traditional approach prescribing zero boundary condition at the boundary of the computational domain, this domain must be very large in order to justify the choice of zero boundary condition and to get good accuracy.

Example 3.2. We solve the one-dimensional Burgers equation

$$\phi_t + \frac{1}{2}(\phi_x + 1)^2 = 0, \tag{25}$$

Table 3
Numerical errors and orders of accuracy at $t = 1$ for Example 3.1 with the initial condition (24) on the characteristic-type moving mesh with $\omega(x) = \sin x$ and with N mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	1.21E-04		8.49E-05		1.10E-04	
64	6.21E-06	4.29	3.37E-06	4.65	6.29E-06	4.13
128	6.24E-07	3.31	2.94E-07	3.52	2.14E-07	4.87
256	8.04E-08	2.96	3.84E-08	2.94	2.90E-08	2.89
512	1.03E-08	2.97	4.93E-09	2.96	3.76E-09	2.95

Table 4
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.2 with the initial condition (26) on the randomly moving mesh with N mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	1.83E-02		1.63E-02		2.59E-02	
64	3.70E-03	2.30	3.66E-03	2.16	6.37E-03	2.02
128	3.30E-04	3.49	3.89E-04	3.23	9.70E-04	2.72
256	1.14E-05	4.85	1.26E-05	4.94	3.87E-05	4.65
512	4.94E-07	4.53	3.84E-07	5.04	6.65E-07	5.86

Table 5
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.2 with the initial condition (26) on the characteristic-type moving mesh with $\omega(x) = \phi_x + 1$ and with N mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.30E-03		1.81E-03		1.91E-03	
64	3.64E-04	2.66	3.29E-04	2.46	4.14E-04	2.21
128	4.59E-05	2.99	4.79E-05	2.78	7.57E-05	2.45
256	1.79E-06	4.68	1.42E-06	5.08	2.34E-06	5.02
512	1.06E-07	4.08	7.16E-08	4.31	6.96E-08	5.07

Table 6
Numerical errors and orders of accuracy at $t = 0.5$ for Example 3.2 with the initial condition (24) on the characteristic-type moving mesh with $\omega(x) = \phi_x + 1$ and with N mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	3.30E-04		2.65E-04		4.36E-04	
64	1.32E-05	4.64	9.44E-06	4.81	1.14E-05	5.26
128	1.69E-06	2.97	1.14E-06	3.05	1.43E-06	2.99
256	2.18E-07	2.96	1.45E-07	2.98	1.79E-07	3.00
512	2.73E-08	2.99	1.81E-08	3.00	2.24E-08	3.00

Table 7
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.3 with the initial condition (28) on the randomly moving mesh with N mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.46E-03		2.35E-03		3.53E-03	
64	4.95E-04	2.31	5.30E-04	2.15	1.10E-03	1.68
128	4.83E-05	3.36	6.10E-05	3.12	1.62E-04	2.77
256	2.88E-06	4.07	3.94E-06	3.95	2.09E-05	2.95
512	1.87E-07	3.94	1.65E-07	4.58	4.06E-07	5.69
1024	2.32E-08	3.01	2.08E-08	2.99	5.32E-08	2.93
2048	3.00E-09	2.95	2.69E-09	2.95	6.82E-09	2.96
4096	3.79E-10	2.99	3.40E-10	2.99	8.63E-10	2.98

with the characteristic boundary condition, that is, the initial computational domain is $x \in [-1.6, 1.6]$ and it is evolved in time by characteristics. This is a nonlinear equation. We use it to show that our algorithm can obtain the designed third order accuracy when the solution is smooth, and can accurately capture the corner singularity without generating oscillations when the singularity appears at later time.

First, we use the initial condition

$$\phi(x, 0) = -\cos(\pi x), \tag{26}$$

and calculate the equation to $t = \frac{0.5}{\pi^2}$. At this time, the solution is still smooth. We test the scheme on the two types of meshes, and list the numerical errors and orders of accuracy in Tables 4–5. We also draw the L_2 error versus the number of grid points in Fig. 4 for the two mesh movement methods. We can observe that it is more efficient to use the characteristic-type moving mesh, as the error is smaller than the randomly moving mesh for the same number of mesh points.

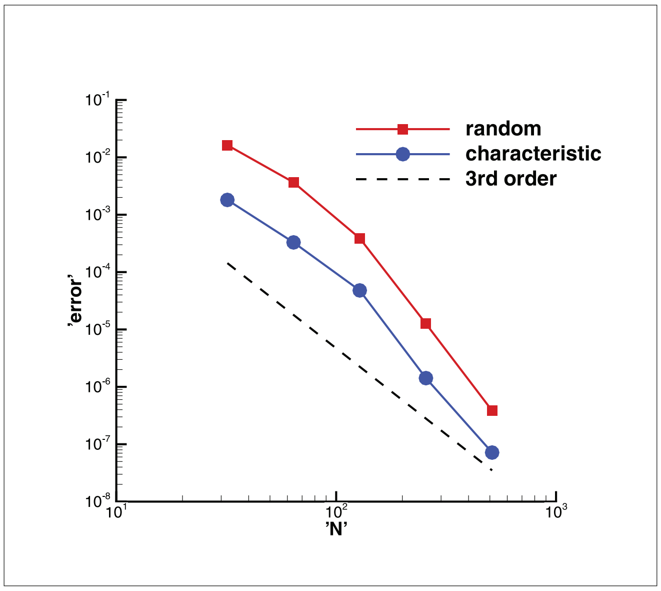

Fig. 4. L_2 errors versus the number of grid points. The characteristic-type moving mesh and the randomly moving mesh for Example 3.2 with the initial condition (23).

Next, we calculate the equation from the same initial condition (26) to the time $t = \frac{1.5}{\pi^2}$, when the solution is no longer smooth. We use the numerical solution on the characteristic-type mesh with 2048 nodes as the reference solution, and plot the results of our ALE-WENO scheme in Fig. 5. We can see that our scheme can achieve high resolution in this example, and the characteristic-type moving mesh produces better results than the randomly moving mesh.

Finally, we choose the Gaussian type function (24) as our initial condition, and compute the equation up to the time $t = 0.5$. The numerical errors and orders of accuracy are shown in Table 6, for the characteristic-type mesh movement. We can see that, even though our computational domain is relatively small and the solution is far from zero at its boundary, we have obtained very small errors with the designed order of accuracy for this example.

When we compute to the time $t = 0.85$ with the same initial condition (24), the solution is no longer smooth. We use the numerical solution on the characteristic-type moving mesh with 2048 points as the reference solution, and plot the results of our ALE-WENO scheme in Fig. 6. We can see that the numerical solution with 16 points is very close to the reference solution.

Example 3.3. We solve the one-dimensional nonlinear problem

$$\phi_t - \cos(\phi_x + 1) = 0, \tag{27}$$

with the characteristic boundary condition, that is, the initial computational domain is $x \in [-1.5, 1.5]$ and it is evolved in time by the characteristics. We first choose the initial condition

$$\phi(x, 0) = -\cos(\pi x). \tag{28}$$

This is a nonlinear, nonconvex Hamiltonian problem. We use it to verify the third-order accuracy of our algorithm when the solution is smooth, and the accurate and non-oscillatory capturing of the corner singularity when it appears at later time. First, we use our ALE-WENO scheme (9) to calculate the solution up to the time $t = \frac{0.5}{\pi^2}$, when the solution is still smooth, and list the numerical errors and orders of accuracy in Tables 7–8. We also plot the L_2 errors in Fig. 7. From the tables and figure, we can clearly see the the superiority of the characteristic-type moving mesh versus the randomly moving mesh.

Similar to Example 3.2, we compute the solution to the later time $t = \frac{1.5}{\pi^2}$, when a corner singularity has already appeared. We again choose the numerical solution on the characteristic-type

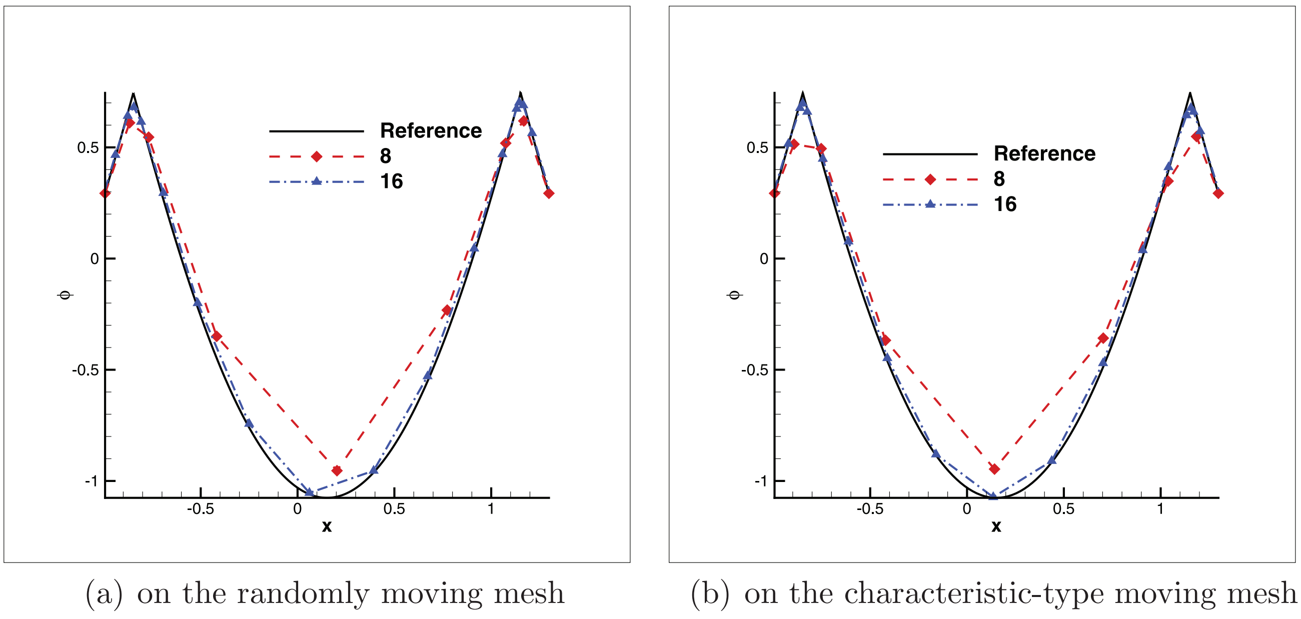

Fig. 5. ϕ at $t = \frac{1.5}{\pi^2}$ with $N = 8, 16$ mesh points for Example 3.2 with the initial condition (26).

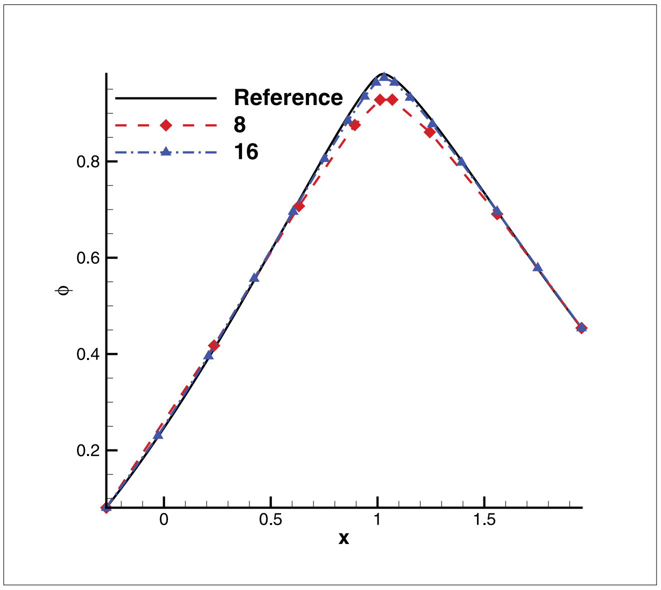

Fig. 6. ϕ at $t = 0.85$ with $N = 8, 16$ mesh points on the characteristic-type moving mesh for Example 3.2 with the initial condition (24).

mesh with 2048 points as the reference solution in the figures. The numerical results are shown in Fig. 8. We can observe that the characteristic-type mesh leads to better resolution with 32 points, than the randomly moving mesh, especially at the the corner singularities.

Next, we choose the Gaussian type function (24) as the initial condition, and compute the equation up to $t = 0.5$. The numerical errors and orders of accuracy are shown in Table 9. We notice that the error is already close to machine zero for $N = 256$, and higher than the expected third order accuracy has been achieved for this example.

From the same initial condition (24), we compute to the time $t = 2.0$. The solution is not smooth anymore. We use the the numerical solution on the characteristic-type mesh with 2048 points as the reference solution and then we plot the results of our ALE-WENO scheme in Fig. 9. The solution with 10 points is very close to the “exact” reference solution. Even the result with 5 points has good resolution.

Example 3.4. We solve the problem

$$\phi_t + \frac{1}{4}(\phi_x^2 - 1)(\phi_x^2 - 4) = 0, \tag{29}$$

which comes from Zhang and Shu [15], as a difficult test case for nonconvex Hamiltonians and entropy conditions (viscosity solutions). We use the initial condition

$$\phi(x, 0) = -2|x| \tag{30}$$

and compute the equation up to $t = 1$, and plot the results in Fig. 10, in which we compare the results of the characteristic-type moving meshes, and the randomly moving meshes, both with $N = 32$, 64 and 128 mesh points, against the “exact” reference solution. We choose the initial computational domain as $x \in [-4, 4]$

Table 8
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.3 with the initial condition (28) on the characteristic-type moving mesh with $\omega(x) = \sin(\phi_x + 1)$ and with N mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	1.12E-04		1.08E-04		2.16E-04	
64	6.56E-05	0.78	7.56E-05	0.51	1.62E-04	0.41
128	8.94E-06	2.88	1.21E-05	2.65	3.33E-05	2.28
256	5.09E-07	4.13	4.30E-07	4.81	9.15E-07	5.19
512	4.95E-08	3.36	3.84E-08	3.49	5.97E-08	3.94
1024	1.15E-08	2.10	9.12E-09	2.07	1.42E-08	2.07
2048	2.04E-09	2.49	1.62E-09	2.49	2.45E-09	2.54
4096	2.97E-10	2.78	2.37E-10	2.77	4.13E-10	2.57

Table 9
Numerical errors and orders of accuracy at $t = 0.5$ for Example 3.3 with the initial condition (24) on the characteristic-type moving mesh with $\omega(x) = \sin(\phi_x + 1)$ and with N mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	9.80E-08		1.25E-07		3.53E-07	
64	2.25E-10	8.77	2.09E-10	9.22	4.05E-10	9.77
128	3.27E-12	6.11	3.22E-12	6.02	6.30E-12	6.01
256	5.36E-14	5.93	5.20E-14	5.95	1.04E-13	5.92

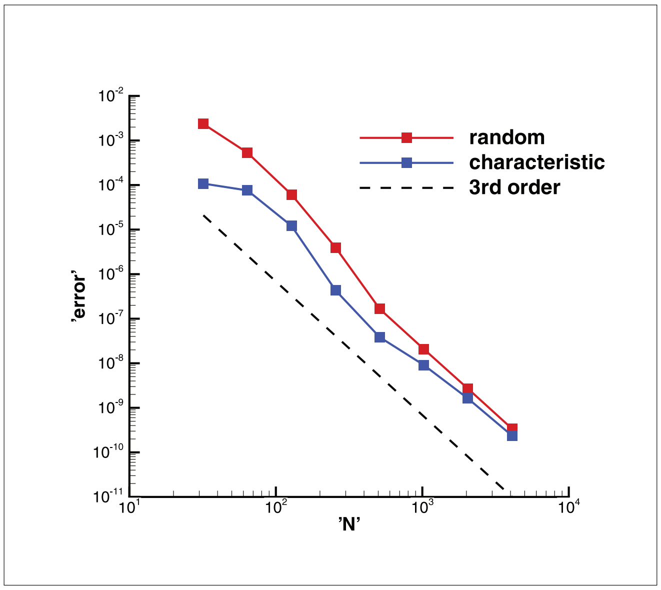

Fig. 7. L_2 errors versus the number of grid points. The characteristic-type moving mesh and the randomly moving mesh for Example 3.3 with the initial condition (23).

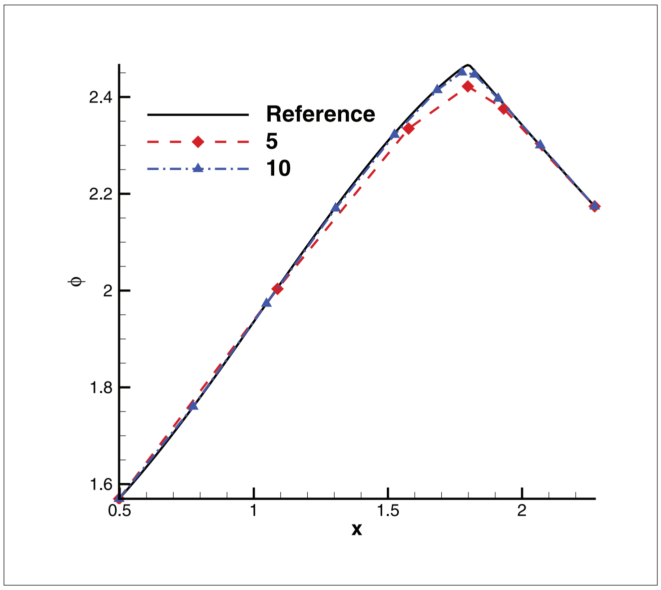

Fig. 9. ϕ at $t = 1.0$ with $N = 5$ and 10 mesh points on the characteristic-type moving mesh for Example 3.3 with the initial condition (24).

and plot the solution at the final time. From the figure, we can clearly observe that the characteristic-type moving mesh produces more accurate results than the randomly moving mesh, with the same number of mesh points. This example is quite demanding. Many numerical methods can not get convergence to the correct viscosity solution. We can see that our algorithm has good convergence for this difficult test case.

Example 3.5. We solve the two-dimensional linear equation with variable coefficients

$$\phi_t + \sin\left(\frac{x+y}{2}\right)(\phi_x + \phi_y) = 0 \tag{31}$$

with the characteristic boundary condition, that is, the initial computational domain is $(x, y) \in [-2, 3]^2$ and it is evolved in time by characteristics. We extend Example 3.1 from a one-dimensional situation to a two-dimensional situation. This is a linear variable coefficient equation in two dimensional case. We verify the third-order accuracy of our algorithm for calculating smooth solutions for such variable coefficient equations.

We choose the initial condition

$$\phi(x, y, 0) = \sin\left(\frac{x+y}{2}\right) \tag{32}$$

Then we list the numerical errors and orders of accuracy simulated by the ALE-WENO scheme in Tables 10-11. We also plot the L_2 error of the characteristic-type moving mesh and the randomly moving mesh cases in Fig. 11. We can clearly see that it is more efficient to use the characteristic-type moving mesh because its error is smaller than that on the randomly moving mesh with the same

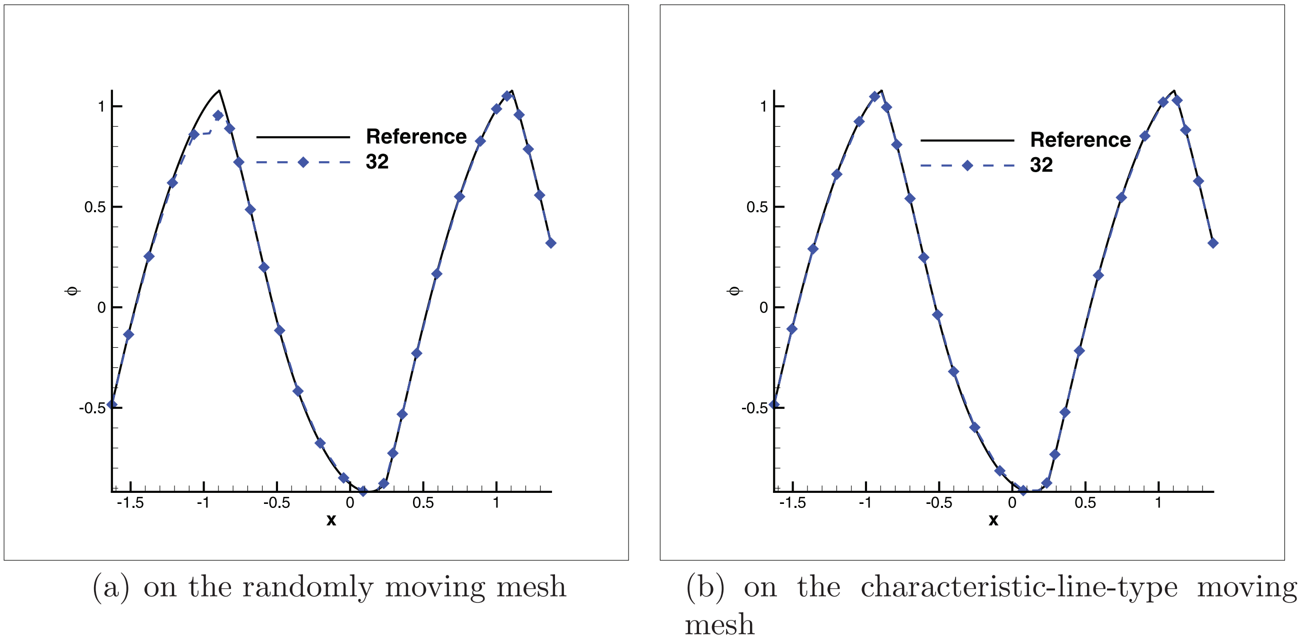

Fig. 8. ϕ at $t = \frac{1.5}{\pi^2}$ with $N = 32$ mesh points for Example 3.3.

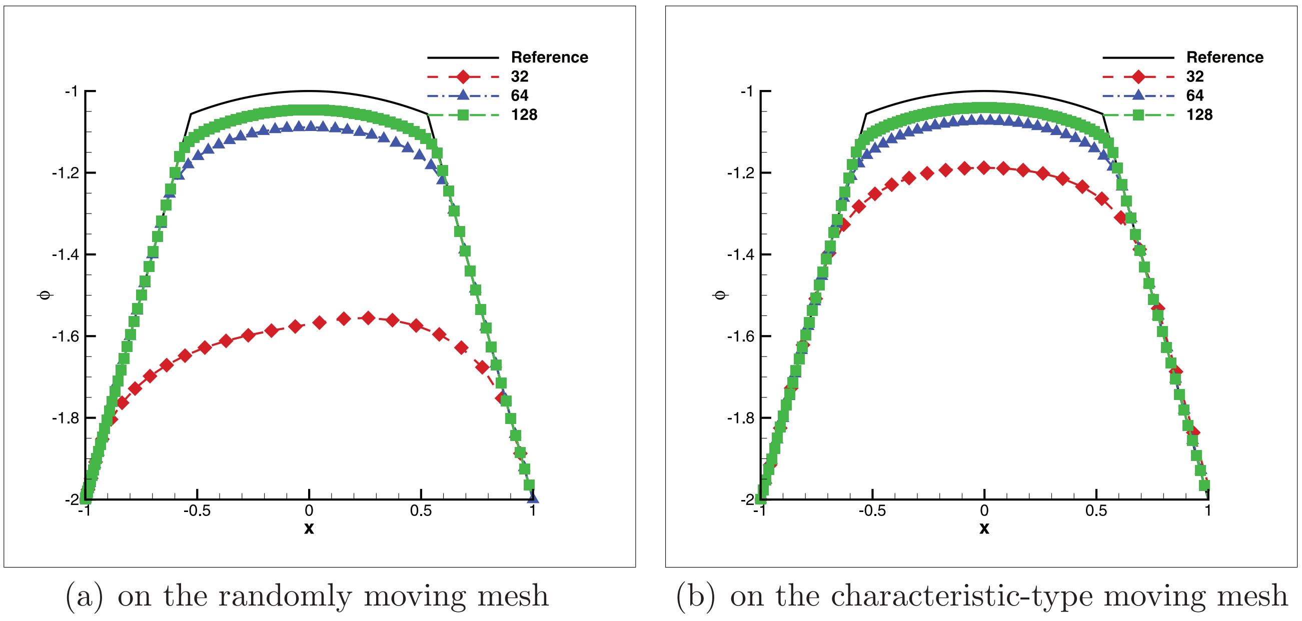

Fig. 10. ϕ at $t = 1$ with $N = 32, 64, 128$ mesh points for Example 3.4.

number of mesh points. In this case, since the computational domain is relatively large, hence h for the coarse mesh is quite large. In order to reduce the error in the time direction, we set Δt determined by (20) to be h when it is larger than h.

Next, we use the following two dimensional Gaussian type function as the initial condition

$$\phi(x, y, 0) = e^{-\frac{(x+y)^2}{8}} + \frac{1}{2}e^{-\frac{(x+y-1)^2}{16}} \tag{33}$$

and compute the equation up to the time $t = 1$. The numerical errors and orders of accuracy by the characteristic-type moving meshes are shown in Table 12. We can find that even though we compute in a relatively small initial domain, with the exact solution far from zero at its boundary, we can still get small errors and the expected third order accuracy.

Example 3.6. We solve the two dimensional Burgers equation

$$\phi_t + \frac{1}{2}(\phi_x + \phi_y + 1)^2 = 0, \tag{34}$$

Table 10
Numerical errors and orders of accuracy at $t = 1$ for Example 3.5 with the initial condition (32) on the randomly moving mesh with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	1.75E-06		2.57E-06		6.91E-06	
64	2.60E-07	2.76	3.55E-07	2.85	8.76E-07	2.98
128	3.96E-08	2.71	4.98E-08	2.83	1.12E-07	2.97
256	6.31E-09	2.65	7.43E-09	2.75	1.52E-08	2.88
512	1.02E-09	2.64	1.16E-09	2.68	2.51E-09	2.60

Table 11
Numerical errors and orders of accuracy at $t = 1$ for Example 3.5 with the initial condition (32) on the characteristic-type moving mesh with $\omega(x, y) = (\sin(\frac{x+y}{2}), \sin(\frac{x+y}{2}))$ and with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	1.35E-06		1.81E-06		6.54E-06	
64	1.42E-07	3.25	1.77E-07	3.35	7.78E-07	3.07
128	1.56E-08	3.18	1.86E-08	3.25	9.49E-08	3.03
256	1.83E-09	3.09	2.12E-09	3.13	1.17E-08	3.01
512	2.22E-10	3.04	2.54E-10	3.07	1.47E-09	3.00

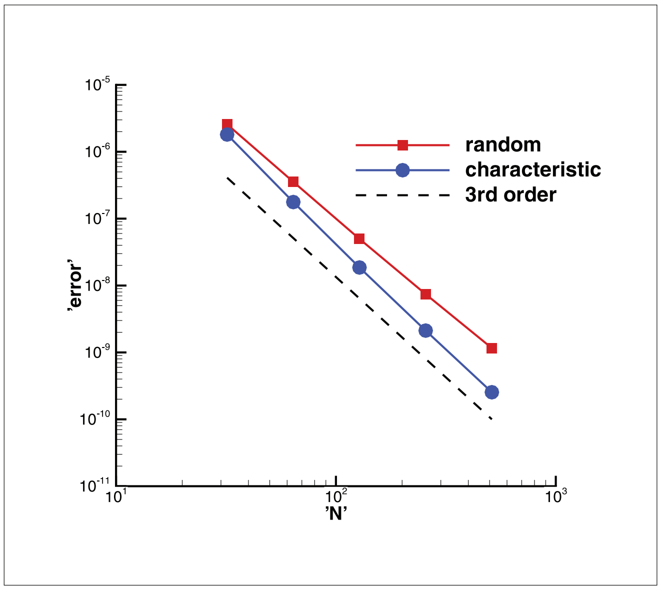

Fig. 11. L_2 errors versus the number of grid points in each direction. The characteristic-type moving mesh and the randomly moving mesh for Example 3.5 with the initial condition (32).

Table 12
Numerical errors and orders of accuracy at $t = 1$ for Example 3.5 with the initial condition (33) on the characteristic-type moving mesh with $\omega(x, y) = (\sin(\frac{x+y}{2}), \sin(\frac{x+y}{2}))$ and with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	1.30E-06		1.78E-06		8.75E-06	
64	1.28E-07	3.34	1.73E-07	3.36	1.04E-06	3.07
128	1.32E-08	3.28	1.75E-08	3.31	1.27E-07	3.03
256	1.45E-09	3.18	1.89E-09	3.21	1.57E-08	3.01
512	1.69E-10	3.10	2.16E-10	3.13	1.96E-09	3.00

with the characteristic boundary conditions, that is, the initial computational domain is $(x, y) \in [-4, 4]^2$ and it is evolved in time by characteristics. Compared with Example 3.2, this example extends the one-dimensional case to the two-dimensional case. Simi-

Table 13
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.6 with the initial condition (35) on the randomly moving mesh with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	7.57E-03		1.02E-02		2.32E-02	
64	1.33E-03	2.51	2.16E-03	2.23	5.67E-03	2.04
128	9.56E-05	3.80	1.71E-04	3.66	5.63E-04	3.33
256	3.75E-06	4.67	4.49E-06	5.25	1.35E-05	5.38
512	3.44E-07	3.45	3.72E-07	3.59	6.32E-07	4.42

Table 14
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.6 with the initial condition (35) on the characteristic-type moving mesh with $\omega(x, y) = (\phi_x + \phi_y + 1, \phi_x + \phi_y + 1)$ and with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	3.46E-03		4.97E-03		1.17E-02	
64	4.61E-04	2.91	7.79E-04	2.67	2.16E-03	2.44
128	2.23E-05	4.37	3.16E-05	4.62	9.08E-05	4.57
256	1.46E-06	3.93	1.66E-06	4.25	3.08E-06	4.88
512	1.84E-07	2.99	1.95E-07	3.09	2.78E-07	3.47

Table 15
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.6 with the initial condition (33) on the characteristic-type moving mesh with $\omega(x, y) = (\phi_x + \phi_y + 1, \phi_x + \phi_y + 1)$ and with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.17E-05		2.81E-05		7.01E-05	
64	2.67E-06	3.02	3.35E-06	3.07	8.04E-06	3.12
128	3.63E-07	2.88	4.47E-07	2.91	1.02E-06	2.97
256	4.66E-08	2.96	5.69E-08	2.97	1.29E-07	2.99
512	5.91E-09	2.98	7.19E-09	2.99	1.61E-08	3.00

Table 16
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.7 with the initial condition (37) on the randomly moving mesh with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.09E-03		2.89E-03		6.75E-03	
64	3.58E-04	2.55	5.91E-04	2.29	1.71E-03	1.98
128	2.85E-05	3.65	5.05E-05	3.55	1.81E-04	3.24
256	1.32E-06	4.43	1.58E-06	5.00	3.77E-06	5.58
512	1.31E-07	3.34	1.63E-07	3.28	4.80E-07	2.98

Table 17
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.7 with the initial condition (37) on the characteristic-type moving mesh with $\omega(x, y) = (\sin(\phi_x + \phi_y + 1), \sin(\phi_x + \phi_y + 1))$ and with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	1.97E-03		2.74E-03		6.00E-03	
64	3.38E-04	2.54	5.61E-04	2.29	1.47E-03	2.03
128	2.72E-05	3.64	4.83E-05	3.54	1.63E-04	3.17
256	1.27E-06	4.42	1.51E-06	5.00	3.23E-06	5.66
512	1.25E-07	3.34	1.55E-07	3.28	3.93E-07	3.04

lar to the one-dimensional case, we verify the designed third-order accuracy of our algorithm when the solution is smooth, and plot the solution when corner singularities appear to show that our algorithm can accurately capture them without generating oscillations.

We take

$$\phi(x, y, 0) = -\cos\left(\pi \frac{x+y}{2}\right) \tag{35}$$

as the initial condition.

At the time $t = \frac{0.5}{\pi^2}$, the solution is still smooth. We list the numerical errors and orders of accuracy simulated by the ALE-WENO scheme in Tables 13–14. It can again be observed that our ALE-WENO scheme on both types of mesh motions can achieve high order accuracy. Also from the tables, we can see that we have smaller errors by the characteristic-type moving mesh than by the randomly moving mesh with the same number of mesh points. It can also been seen clearly from Fig. 12 which shows the L_2 errors of the two moving meshes.

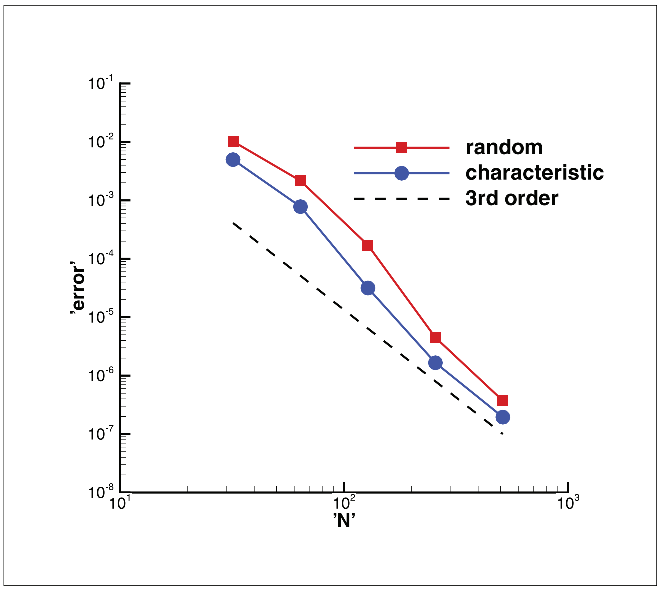

Fig. 12. L_2 errors versus the number of grid points in each direction. The characteristic-type moving mesh and the randomly randomly moving mesh for Example 3.6 with the initial condition (32).

The mesh and the contours of ϕ obtained by the scheme (9) at $t = \frac{1.5}{\pi^2}$, when the solution is no longer smooth, are shown in Figs. 13–15. We can observe that the characteristic-type moving mesh produces more accurate results than the randomly moving mesh with the same number of mesh points.

Also, we use the two dimensional Gaussian type function (33) as the initial condition. The initial computational domain is $(x, y) \in [-8, 8]^2$ and it is evolved in time by the characteristics. The numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$, when the solution is smooth, are shown in Table 15. Again, we observe that we can get small errors and the expected third order accuracy.

We then compute to the time $t = 0.85$, when the solution is no longer smooth. We plot the results of our ALE-WENO scheme in Figs. 16–17. We can again observe that the characteristic-type moving mesh produces very good results.

Example 3.7. We solve the two dimensional nonlinear equation

$$\phi_t - \cos(\phi_x + \phi_y + 1) = 0 \tag{36}$$

with the characteristic boundary condition, that is, the initial computational domain is $(x, y) \in [-3, 3]^2$ and it is evolved in time by characteristics. This example is the two-dimensional counterpart of Example 3.3. Similar to the previous example, we are interested in verifying the third-order accuracy for smooth solutions, and verifying that the scheme can capture the corner singularities sharply without oscillations.

The initial condition is

$$\phi(x, y, 0) = -\cos\left(\pi \frac{x+y}{2}\right) \tag{37}$$

We compute the equation up to the time $t = \frac{0.5}{\pi^2}$. The numerical errors and orders of accuracy are shown in Tables 16–17. We can

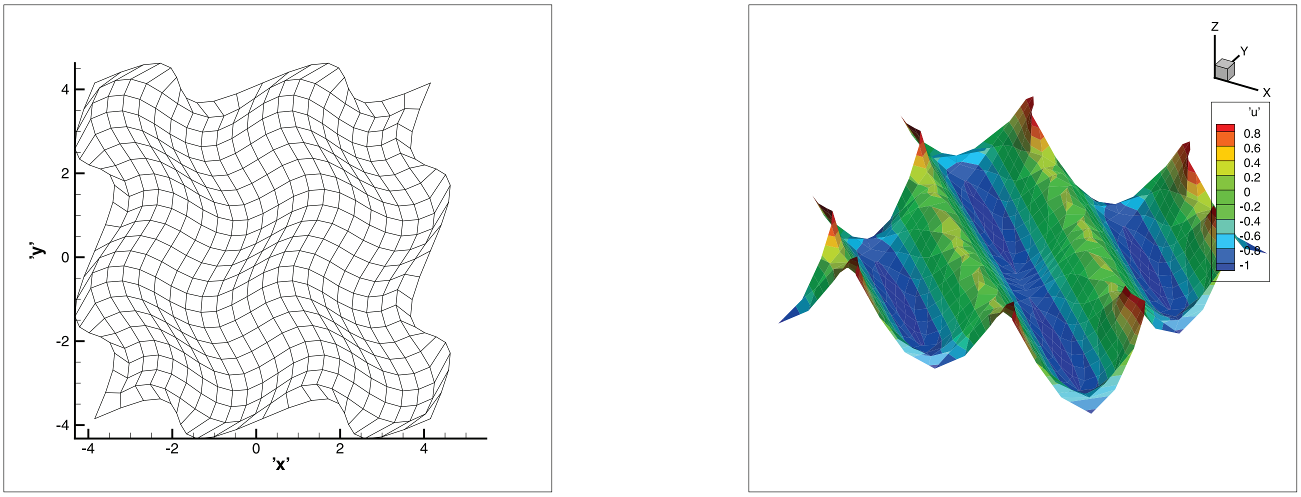

Fig. 13. The mesh and contours of ϕ at $t = \frac{1.5}{\pi^2}$ for Example 3.6 with the initial condition (35), on the randomly moving mesh, 22 × 22 mesh points.

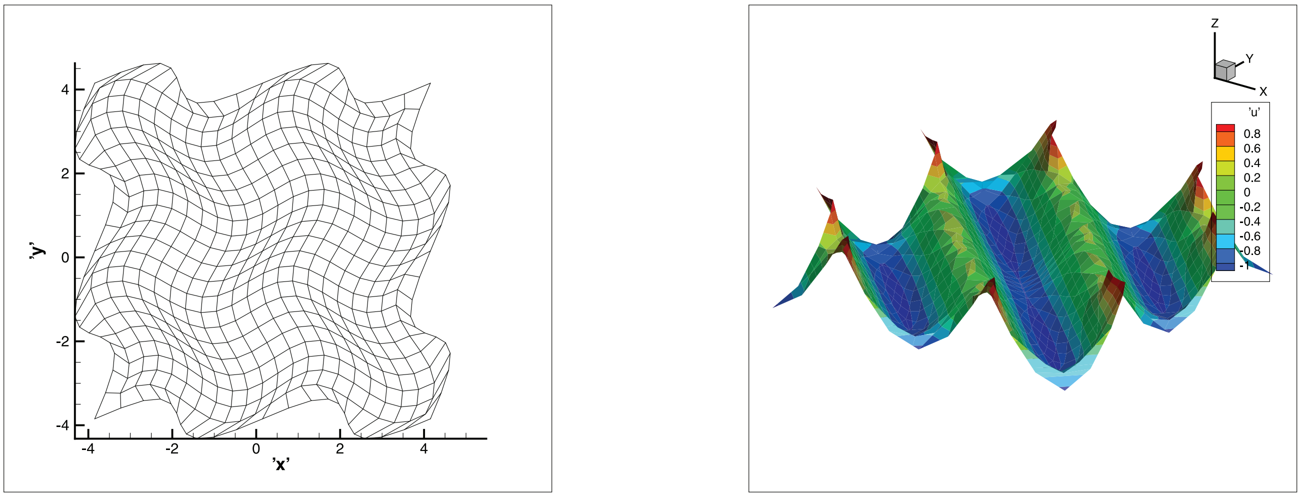

Fig. 14. The mesh and contours of ϕ at $t = \frac{1.5}{\pi^2}$ for Example 3.6 on the characteristic-line-type moving mesh $\omega(x, y) = (\phi_x + \phi_y + 1, \phi_x + \phi_y + 1)$, with the initial condition (35), 22 × 22 mesh points.

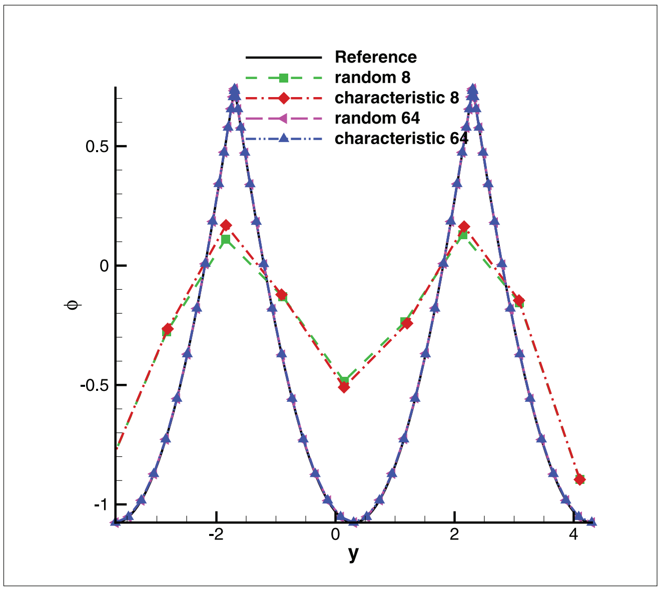

Fig. 15. ϕ at $t = \frac{1.5}{\pi^2}$ in Example 3.6, cut along $x = 0$ for the comparison of results between the characteristic-type moving mesh and the randomly moving mesh against the reference solution, $N = 8 \times 8, 64 \times 64$ mesh points.

clearly see that we get the expected order accuracy. From tables and the L_2 error plots in Fig. 18, we can see that the errors of the characteristic-type moving mesh method is slightly smaller than that of the randomly moving mesh method under the same number of mesh points.

The mesh and the contours of ϕ at $t = \frac{1.5}{\pi^2}$, when the solution is no longer smooth, are shown in Figs. 19–21. We can observe that the performance of this example is similar to Example 3.6 in capturing corners.

Like in the previous examples, next we use the two dimensional Gaussian type function (33) to be the initial condition. We adopt the characteristic boundary condition to obtain the moving com-

Table 18
Numerical errors and orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.7 with the initial condition (33) on the characteristic-type moving mesh with $\omega(x, y) = (\sin(\phi_x + \phi_y + 1), \sin(\phi_x + \phi_y + 1))$ and with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	4.71E-05		7.32E-05		2.78E-04	
64	2.63E-06	4.16	3.30E-06	4.47	7.55E-06	5.20
128	3.11E-07	3.08	3.96E-07	3.06	9.54E-07	2.98
256	3.91E-08	2.99	4.98E-08	2.99	1.20E-07	2.99
512	4.93E-09	2.99	6.25E-09	2.99	1.51E-08	2.99

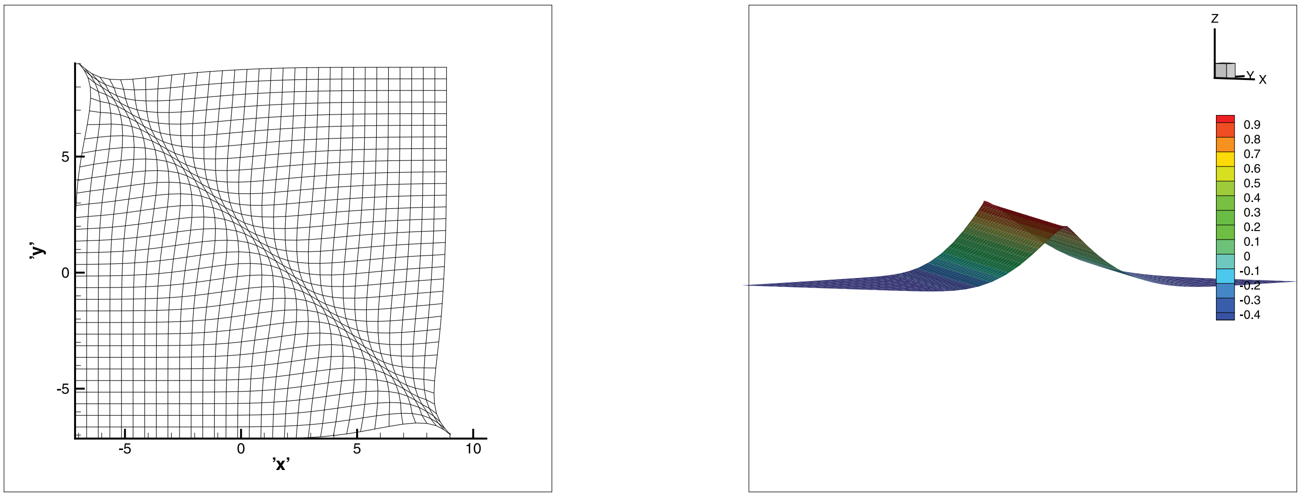

Fig. 16. The mesh and contours of ϕ at $t = 0.85$ for Example 3.6 with the initial condition (33) on the characteristic-type moving mesh with $\omega(x, y) = (\phi_x + \phi_y + 1, \phi_x + \phi_y + 1)$, 32×32 mesh points.

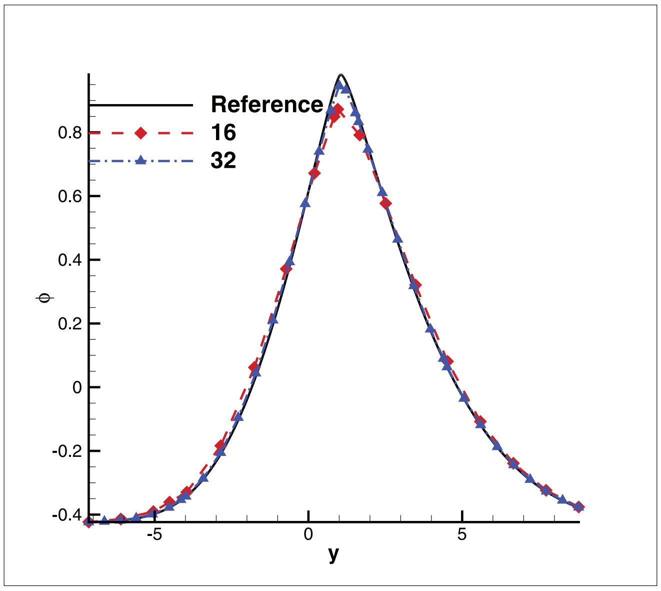

Fig. 17. ϕ at $t = 0.85$ in Example 3.6 with the initial condition (33), cut along $x = 1$ for the comparison of results on the characteristic-type moving mesh between the $N = 16 \times 16$ and $N = 32 \times 32$ mesh points against the reference solution.

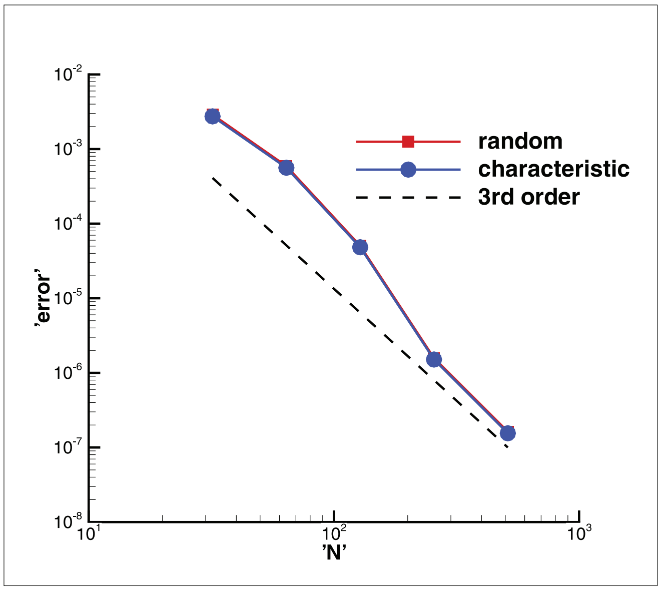

Fig. 18. L_2 errors versus the number of grid points in each direction. The characteristic-type moving mesh and the randomly moving mesh for Example 3.7 with the initial condition (32).

putational domain. The numerical errors and orders of accuracy are shown in Table 18. Obviously we can find that it has similarly good performance as before.

Finally, we compute to the time $t = 1.4$, when the solution is not smooth anymore. We use the ALE-WENO scheme on the characteristic-type moving mesh and then we plot the mesh and the contours of ϕ in Fig. 22 and the comparison of results in cuts from the 8×8 and 16×16 mesh points in Fig. 23.

Example 3.8. We solve the problem

$$\phi_t + H(x, y, \phi, \phi_x, \phi_y) = 0 \tag{38}$$

with the Hamiltonian

$$H = \phi - \frac{1}{2}[e^y coshx + e^{-y} - 2] - e^{-y}\tanh\left(\frac{x}{2}\right)\frac{\partial\phi}{\partial x}$$
$$-[1 - e^{-y}]\frac{\partial\phi}{\partial y} + \frac{e^{-y}}{4\cosh^2(\frac{x}{2})}\left(\frac{\partial\phi}{\partial x}\right)^2$$

This example comes from Lefevre et al. [12], as a model in nonlinear solid mechanics. We use the realistic model Hamilton-Jacobi equation to verify that our scheme can effectively calculate complex nonlinear problems and guarantee third-order accuracy. We have used this example in [13] to test our WENO-ALE algorithm with a zero initial condition. Here, to be consistent with the previous examples, we take the initial condition as

$$\phi(x, y, 0) = e^{-\frac{(x+y)^2}{8}} + \frac{1}{2}e^{-\frac{(x+y-1)^2}{16}} \tag{39}$$

In this example, the initial domain is $(x, y) \in [-4, 4]^2$ and it is evolved in time by characteristics.

The solution is computed up to the time $t = 0.1$, and we list the numerical errors and orders of accuracy simulating by the ALE-WENO scheme (9) in Tables 19–20. We can see that we have obtained the designed third order accuracy. We also plot the L_2 errors in Fig. 24, from which we can see that the L_2 errors for the characteristic-type moving meshes are smaller in magnitude than those with the randomly moving meshes for the same number of mesh points.

We also plot the meshes and the solutions corresponding to both types of mesh movements in Figs. 25–26.

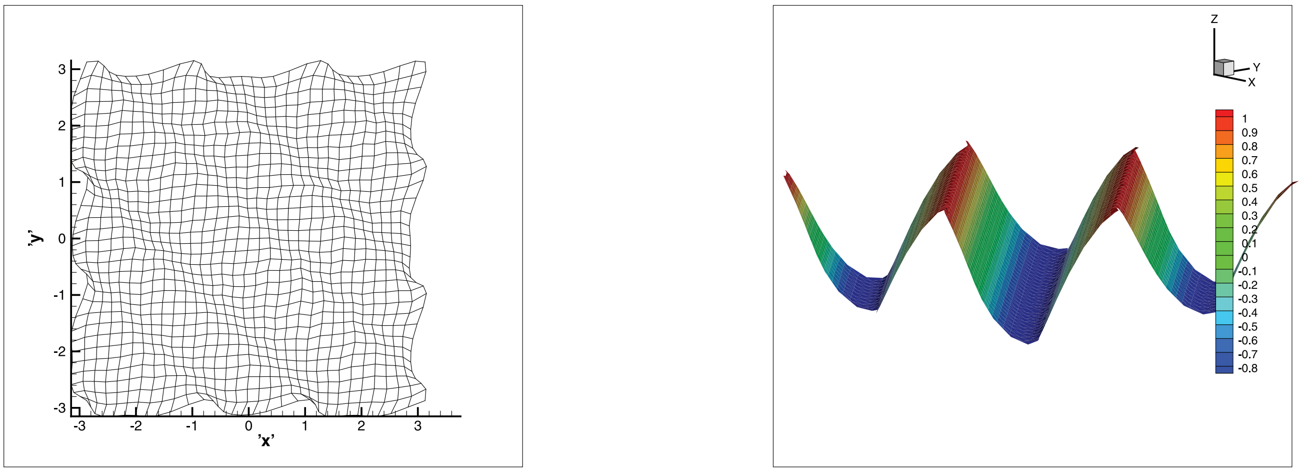

Fig. 19. The mesh and contours of ϕ at $t = \frac{1.5}{\pi^2}$ for Example 3.9 on the randomly moving mesh, 32×32 mesh points.

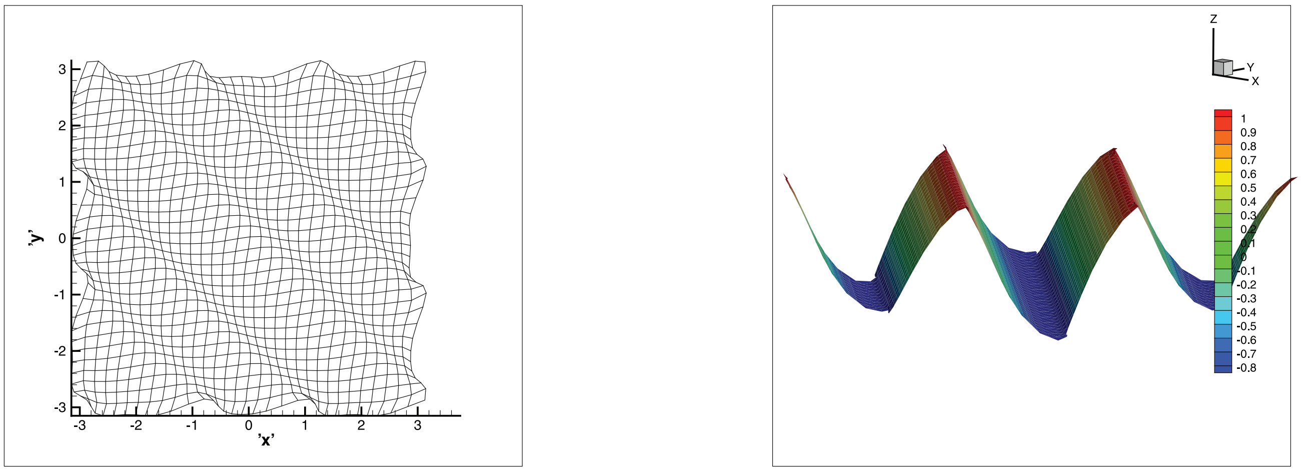

Fig. 20. The mesh and contours of ϕ at $t = \frac{1.5}{\pi^2}$ for Example 3.9 on the characteristic-type moving mesh with $\omega(x, y) = (\phi_x + \phi_y + 1, \phi_x + \phi_y + 1)$, 32×32 mesh points.

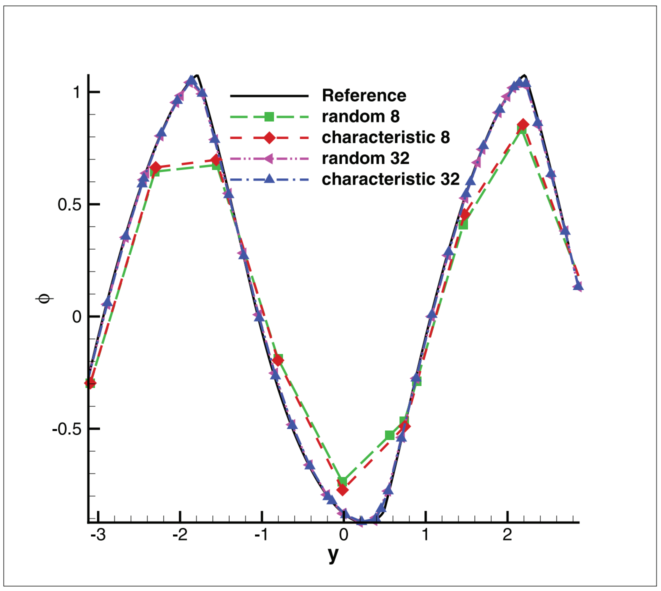

Fig. 21. ϕ at $t = \frac{1.5}{\pi^2}$ in Example 3.7, cut along $x = 0$ for the comparison of results from the characteristic-type moving mesh and the randomly moving mesh against the reference solution, $N = 8 \times 8, 32 \times 32$ mesh points.

Table 19
Numerical errors and orders of accuracy at $t = 0.1$ for Example 3.8 on the randomly moving mesh with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	3.44E-03		1.25E-02		1.09E-01	
64	1.14E-03	1.60	4.92E-03	1.34	5.73E-02	0.92
128	2.82E-04	2.01	1.78E-03	1.46	2.18E-02	1.39
256	5.16E-05	2.45	3.21E-04	2.47	5.44E-03	2.00
512	6.79E-06	2.92	4.62E-05	2.80	7.94E-04	2.78

Table 20
Numerical errors and orders of accuracy at $t = 0.1$ for Example 3.8 on the characteristic-line-type moving mesh with $\omega(x, y) = (-e^{-y}\tanh\left(\frac{x}{2}\right) + \frac{e^{-y}}{2G\cosh^2(\frac{x}{2})}\phi_x, -(1 - e^{-y}))$ and with $N \times N$ mesh points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.47E-03		1.02E-02		1.01E-01	
64	3.91E-04	2.66	2.54E-03	2.01	4.31E-02	1.23
128	2.60E-05	3.91	1.06E-04	4.58	1.23E-03	5.13
256	4.16E-06	2.64	1.81E-05	2.55	2.44E-04	2.33
512	5.79E-07	2.84	2.62E-06	2.79	3.94E-05	2.63

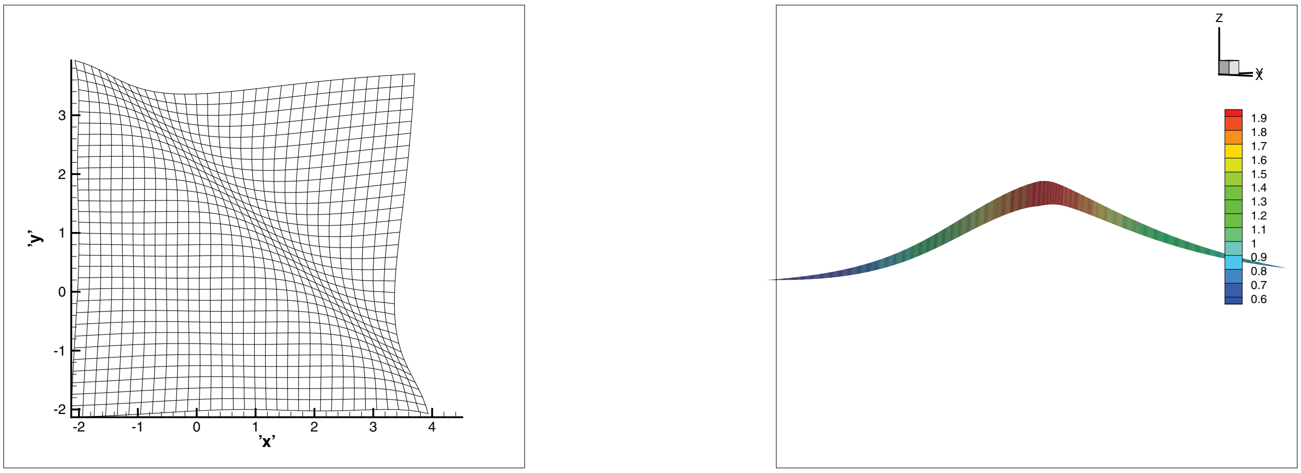

Fig. 22. The mesh and contours of ϕ at $t = 1.0$ for Example 3.7 on the characteristic-type moving mesh with $\omega(x, y) = (\phi_x + \phi_y + 1, \phi_x + \phi_y + 1)$, 32×32 mesh points.

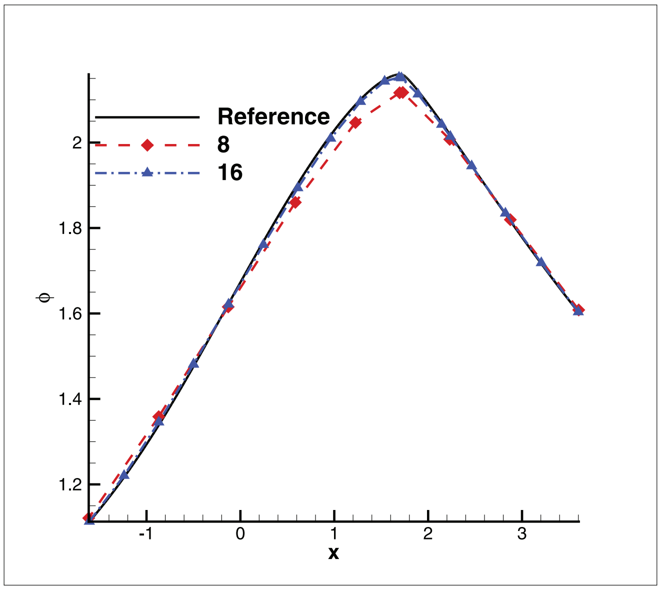

Fig. 23. ϕ at $t = 1.4$ in Example 3.7, cut along $x = 1$, for the comparison of results from the 8×8 and 16×16 mesh points against the reference solution.

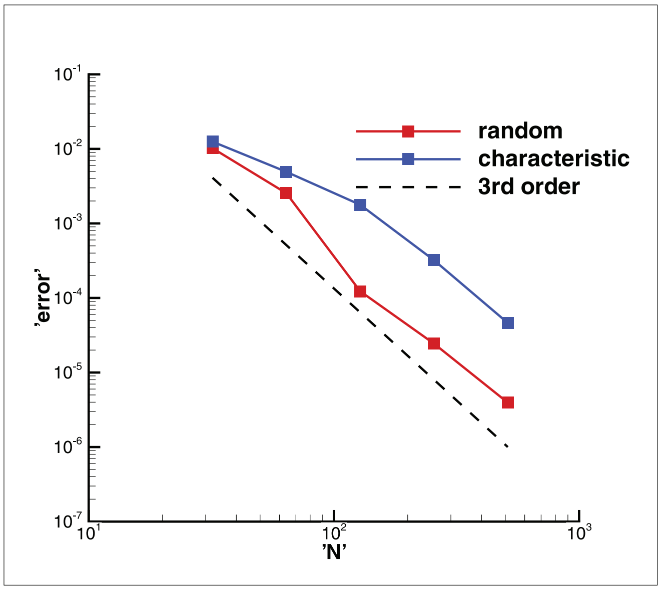

Fig. 24. L_2 errors versus the number of grid points in each direction. The characteristic-type moving mesh and the randomly moving mesh for Example 3.8 with the initial condition (32).

4. Concluding remarks

In this paper, we have developed a framework to use a finite and moving domain and characteristic boundary conditions by evolving the characteristic ODEs along the boundary of the computational domain, to solve Hamilton-Jacobi equations defined on infinite domains. The high order multi-resolution finite difference WENO scheme in the ALE framework on moving meshes, developed recently in [13], is used inside the moving computational domain. Our algorithm can achieve high order accuracy in smooth regions and can avoid spurious oscillations near the corner singularities, and can save the computational cost significantly by solving only in the domain which is of interest to us. Ample numerical examples including the Gaussian type initial conditions without compact support are used to verify the robustness and accuracy of our algorithm. One of the limitations of the proposed boundary treatment is that we require that singularities do not appear at the artificial boundary, nor do singularities from inside the computational domain reach the artificial boundary during the time of computation. This is because when such singularities appear at the artificial boundary, the characteristic ODEs from different initial points would have intersecting solutions, thus requiring special techniques such as the Hopf formula [10] to single out the viscosity solution. This generalization will be studied in the future. Also in the future, we plan to extend this framework to solve hyperbolic conservation laws including compressible Euler equations, and eventually we hope to develop a combined ALE-WENO solver to simulate multi-material flows.

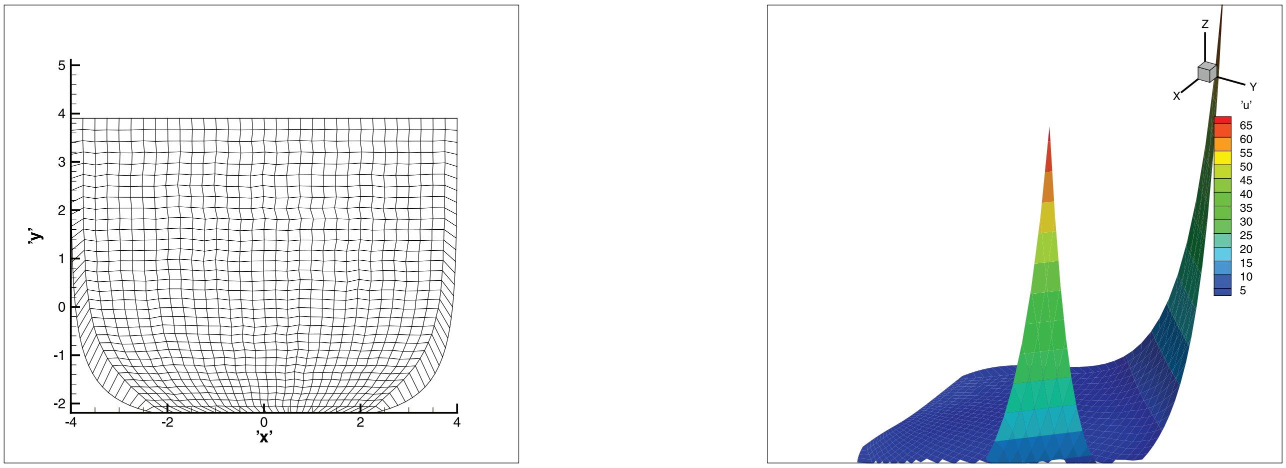

Fig. 25. The mesh and contours of ϕ at $t = 0.1$ for Example 3.8 on the randomly moving mesh, 32×32 mesh points.

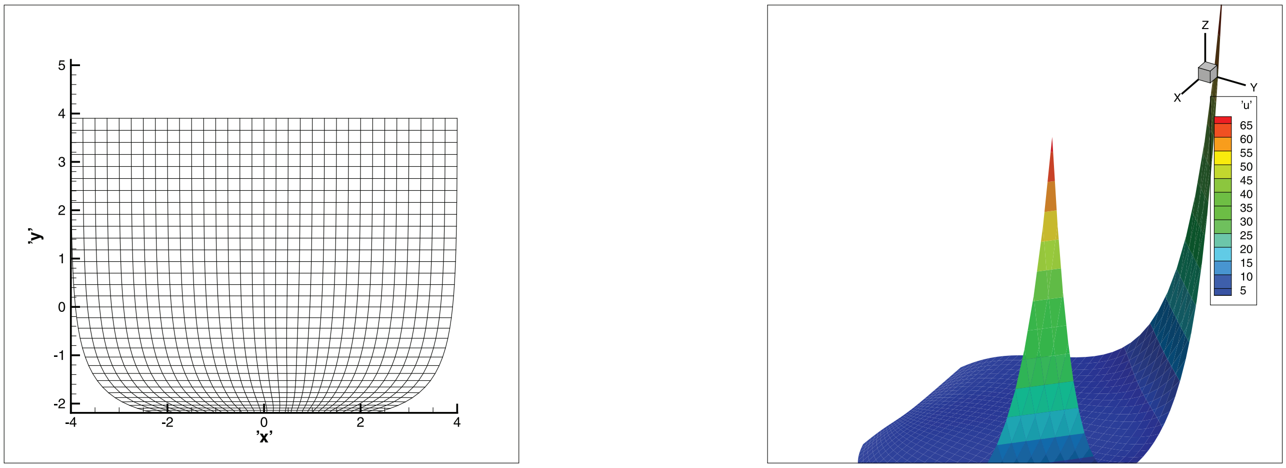

Fig. 26. The mesh and contours of ϕ at $t = 0.1$ for Example 3.8 on the characteristic-line-type moving meshes with $\omega(x, y) = \left(e^{-y}\tanh\left(\frac{x}{2}\right) + \frac{e^{-y}}{2G\cosh^2\left(\frac{x}{2}\right)}\phi_x, -(1 - e^{-y})\right)$, 32×32 mesh points.

CRediT authorship contribution statement

Yue Li: Conceptualization, Methodology, Writing - original draft, Writing - review & editing, Software. **Juan Cheng:** Conceptualization, Methodology, Writing - original draft, Writing - review & editing. **Yinhua Xia:** Conceptualization, Methodology, Writing - original draft, Writing - review & editing. **Chi-Wang Shu:** Conceptualization, Methodology, Writing - original draft, Writing - review & editing.

References

[1] Abgrall R. Numerical discretization of the first-order Hamilton-Jacobi equation on triangular meshes. Commun Pure Appl Math 1996;49:1339–73.
[2] Brebbia CA. The boundary element method for engineers. London: Pentech Press; 1978.
[3] Corral R, Jiménez J. Fourier/Chebyshev methods for the incompressible Navier-Stokes equations in infinite domains. J Comput Phys 1995;121:261–70.
[4] Demidov SS. The study of partial differential equations of the first order in the 18th and 19th centuries. Arch Hist Exact Sci 1982;26:325–50.
[5] Du Q, Zhang M. A non-overlapping domain decomposition algorithm based on the natural boundary reduction for wave equations in an unbounded domain. Numer Math J Chinese Univ 2004;13:121–32.
[6] Engquist B, Majda A. Absorbing boundary conditions for numerical simulation of waves. Math Comp 1977;31:629–51.
[7] Feng K. Finite element method and natural boundary reduction. Proc Int CongrMath 1983;1:1439–53.
[8] Givoli D. Non-reflecting boundary conditions. J Comput Phys 1991;94:1–29.
[9] Givoli D, Keller JB. A finite element method for large domains. Comput Methods Appl Mech Eng 1989;76:41–66.
[10] Hopf E. Generalized solutions of nonlinear equations of the first order. J Math Mech 1965;14:951–73.
[11] Klingenberg C, Schnucke G, Xia Y. An arbitrary Lagrangian-Eulerian local discontinuous Galerkin method for Hamilton-Jacobi equations. J Sci Comput 2017;73:906–42.
[12] Lefevre V, Garnica A, Lopez-Pamies O. A WENO finite-difference scheme for a new class of Hamilton-Jacobi equations in nonlinear solid mechanics. Comput Methods Appl Mech Eng 2019;349:17–44.
[13] Li Y, Cheng J, Xia Y, Shu CW. High order arbitrary Lagrangian-Eulerian finite difference WENO scheme for Hamilton-Jacobi equations. Commun Comput Phys 2019;26:1530–74.
[14] Shu C-W, Osher S. Efficient implementation of essentially non-oscillatory shock capturing schemes. J Comput Phys 1988;77:439–71.
[15] Zhang Y-T, Shu CW. High order WENO schemes for Hamilton-Jacobi equations on triangular meshes. SIAM J Sci Comput 2003;24:1005–30.