A high order positivity-preserving discontinuous Galerkin remapping method based on a moving mesh solver for ALE simulation of the compressible fluid flow

Xiaolu Gu[1], Juan Cheng[2], and Chi-Wang Shu[3]

Abstract

The arbitrary Lagrangian-Eulerian (ALE) method is widely used in the field of compressible multi-material and multi-phase flow problems. In order to implement the indirect ALE approach for the simulation of compressible flow in the context of high order discontinuous Galerkin (DG) discretizations, we present a high order positivity-preserving DG remapping method based on a moving mesh solver in this paper. This remapping method is based on the ALE-DG method developed by Klingenberg et al. [17, 18] to solve the trivial equation $\frac{\partial u}{\partial t} = 0$ on a moving mesh, which is the old mesh before remapping at $t = 0$ and is the new mesh after remapping at $t = T$. An appropriate selection of the final pseudo-time T can always satisfy the relatively mild smoothness requirement (Lipschitz continuity) on the mesh movement velocity, which guarantees the high order accuracy of the remapping procedure. We use a multi-resolution weighted essentially non-oscillatory (WENO) limiter which can keep the essentially non-oscillatory property near strong discontinuities while maintaining high order accuracy in smooth regions. We further employ an effective linear scaling limiter to preserve the positivity of the relevant physical variables without sacrificing conservation and the original high order accuracy. Numerical experiments are provided to illustrate the high order accuracy, essentially non-oscillatory performance and positivity-preserving of our remapping algorithm. In addition, the performance of the ALE simulation based on the DG framework with our remapping algorithm is examined in one- and two-dimensional Euler equations.

Keywords: Remapping, Discontinuous Galerkin method, Arbitrary Lagrangian-Eulerian, High order accuracy, Multi-resolution WENO limiter, Positivity-preserving.

[1]Graduate School, China Academy of Engineering Physics, Beijing 100088, China. E-mail: guxiaolu20@gscaep.ac.cn.

[2]Corresponding author. Laboratory of Computational Physics, Institute of Applied Physics and Computational Mathematics, Beijing 100088, China and HEDPS, Center for Applied Physics and Technology, and College of Engineering, Peking University, Beijing 100871, China. E-mail: cheng_juan@iapcm.ac.cn. Research is supported in part by NSFC grant 12031001, and National Key R&D Program of China No. 2022YFA1004500.

[3]Division of Applied Mathematics, Brown University, Providence, RI 02912. E-mail: chi-wang_shu@brown.edu. Research is supported in part by NSF grant DMS-2010107.

1 Introduction

For the numerical simulation of computational fluid dynamics (CFD), the Eulerian framework and the Lagrangian framework are two classical approaches. In the Eulerian framework, the fluid flows through a fixed mesh. It has strong robustness and can be used in the flow field with large deformation, but getting the precise physical interface is challenging. The Lagrangian framework, in which the mesh moves with the fluid velocity, can naturally and precisely track the interface between different materials and can capture the contact discontinuities sharply. Nevertheless, the computing process in the flow field with significant deformation may be terminated due to mesh distortion. The arbitrary Lagrangian-Eulerian (ALE) approach, which allows the grid points to move with an arbitrary velocity, could combine the best properties of the Lagrangian method and the Eulerian method. The ALE method has been favored in computing compressible flow with large deformation and is flexible in dealing with multi-material problems and the problems with moving domains. The simulations of the compressible Euler equations using the ALE technique have attracted a lot of scientific attention [2, 13, 14, 17, 21, 38, 39, 43].

Generally, ALE methods can be implemented in two manners, i.e., the direct ALE method and the indirect ALE method. The indirect ALE method consists of three individual steps: a Lagrangian step, a rezoning step and a remapping step. In the Lagrangian step, the solution and the computational mesh are updated simultaneously. The nodes of the computational mesh are adjusted to more optimal positions during the rezoning step to improve the quality of the mesh and to relieve the error caused by mesh deformation. The remapping step is then performed, where the Lagrangian solutions are conservatively transferred from the old distorted Lagrangian mesh to the new rezoned mesh. The last two steps are as critical to the accuracy of the overall simulation as the first step since they must preserve the characteristic mesh features as well as the essential mathematical and physical properties of the Lagrangian solution.

In the application of computational fluid dynamics, we can classify the indirect ALE

framework as based on the finite volume (FV) method [10, 21, 38] or the Runge-Kutta discontinuous Galerkin (RK-DG) method [13,17,39]. The numerical solution of the DG method is approximated by polynomials within each element. Hence it is easy to handle problems including discontinuities. It is also flexible for complex mesh geometries and unstructured meshes. Due to the excellent compactness and high order accuracy of the DG method, it has been widely applied to deal with fluid dynamic problems. The type of solution projected in the remapping phase is determined by the discretization methods applied in the Lagrange phase. The remapping stage of the indirect ALE-FV technique transfers the cell averages from the old mesh to the new mesh. There has been much research on this strategy given in [5, 9, 15, 20–22, 30]. Under the indirect ALE framework in conjunction with the DG approach, it is necessary to transfer the high order polynomials to a different set of high order polynomials defined on the new rezoned mesh while maintaining good performance. In this paper, we concentrate on the remapping step in the indirect ALE framework together with the DG method.

There are two fundamentally different strategies for the remapping method. The intersection-based remapping method, expressed as an integral statement of conservation, can be used for the two meshes that are entirely independent of one another. The intersections between the old and the new mesh are precisely computed in this method to determine the contributions of the old cells to the new cells [3, 5, 21–23, 39]. It is suitable for dealing with large flow distortions and large deformation problems. Although this strategy is intuitive, it has a considerable computational cost due to the construction of intersections, especially for curved meshes or higher dimensional situations.

Another common way of remapping is the transport equation based (or flux-based) remapping method. The fundamental premise of this approach is that old and new meshes have the same number of cells and the same mesh connectivity topology, which can be viewed as a deformation of each other. This strategy expresses the remapping procedure as a dynamic process controlled by a linear transport equation. If the mesh optimiza-

tion results in little alterations to the original mesh, the node trajectories are restricted to the nearest neighbor cells, the remapping can be expressed as a flux-form transport algorithm [9, 20, 30, 38]. By solving a linear transport equation over a pseudo-time interval, it is possible to reduce the limits on the movement of grid nodes, for which there is a lot of research [1,2,8,24–26,28,29,33,36,40]. Among them, the authors in [8,25,26] design such an interpolation scheme for the finite element and finite volume schemes on the moving mesh, respectively. This format works with small mesh deformation and requires only one pseudo-time step to complete the interpolation. Ortega and Scovazzi [33] propose a conservative synchronized remap algorithm with nodal finite elements. They point out that the geometric conservation law (GCL) is the fundamental link between advection and remap algorithms. Lipnikov and Morgan [28, 29] provide a high order conservative remap for DG schemes on curved polygonal meshes, extending it to three dimensions. The authors in [40] use the DG method to solve the linear time-dependent equation on deforming meshes and combine it with the positivity-preserving limiter to develop a high order, conservative, and positivity-preserving remapping method applied to radiative transfer for moving grid DG simulations. The above discussed approach avoids accurate calculation of the intersection area between the old and new grid cells, thus reducing complexity and computational cost.

In this paper, we will discuss the design of the high order DG remapping method which will be used in the indirect ALE framework to simulate the compressible fluid flow. Besides the high order accuracy, we also need to take into account other properties such as conservation, positivity-preserving and essentially non-oscillation. For the simulation of compressible fluid flow, the solutions usually contain many discontinuities. The DG method is prone to produce significant non-physical oscillations and even nonlinear instability, resulting in numerical solution explosion. The application of a nonlinear limiter is a common strategy to control these spurious oscillations. One type of these limiters is based on the slope methods such as minmod-type limiters [6, 7, 16]. These limiters are effective in controlling oscillations. However, the accuracy of DG methods may decrease if they are used incorrectly in

smooth regions. Another type of limiters is based on the weighted essentially non-oscillatory (WENO) methodology [34], which can achieve both high order accuracy and non-oscillatory properties. Zhu, Qiu and Shu proposed a new multi-resolution WENO limiter for the discontinuous Galerkin method based on a set of local L^2 projection polynomials in the troubled cells [44, 45]. Compared with the previous WENO limiters, this new WENO limiter is very simple to construct and can be easily implemented to arbitrary high order accuracy and in higher dimensions. The linear weights used in the procedure of the new multi-resolution WENO limiters can be any positive numbers as long as they sum to one. Since the linear weights are not needed to be recalculated at each time step, this limiter is particularly advantageous for the moving mesh methods in which the shape of the mesh cells constantly changes.

For the ALE-DG method used in computational fluid dynamics, the positivity of certain physical quantities, such as density or internal energy, must be preserved during the remapping process to avoid the failure of the numerical solution. Generally speaking, the remapping algorithm alone may not automatically maintain the positivity of the function to be remapped, which should be addressed. Classical techniques use some strategies such as conservative repair procedure [20] and flux corrected method [2, 33, 38] to preserve local bounds. A very popular technique for high order method is a simple scaling positivity-preserving limiter [41], which can preserve high order accuracy without losing local conservation. Numerous numerical experiments have shown the superior performance of this positivity-preserving strategy for the simulation of compressible fluid flow using the high order ALE-DG method [14, 23].

The objective of this paper is to develop a high order positivity-preserving DG remapping method, based on the numerical solution of the trivial equation $\frac{\partial u}{\partial t} = 0$ on a moving mesh, which is the old mesh before remapping at $t = 0$ and is the new mesh after remapping at $t = T$. The simplex meshes in one- and two-dimensions are used as examples. To obtain a high order accurate remapping algorithm, we solve this equation $\frac{\partial u}{\partial t} = 0$ with

the moving mesh DG technique [17, 18] for the spatial discretization and the high order strong stability preserving (SSP) Runge-Kutta method for the time discretization. The novelty of this moving mesh DG technique is that high order accuracy and stability can be proved under very mild conditions on mesh movements. In our remapping method, no smoothness beyond Lipschitz continuity is required for the mesh movement velocity to ensure high order accuracy, which is always achieved by selecting the ending pseudo-time T appropriately. T can be very tiny when the new mesh is only a slight modification of the old mesh, which will reduce the number of pseudo time steps and computing costs. As a result, our remapping approach is beneficial for problems with small mesh modifications, such as ALE algorithm with frequent use of the remapping procedure. We will utilize the Zhang-Shu positivity-preserving framework for preserving the positivity of variables, including density and internal energy, by using a positivity-preserving limiter that is valid under a suitable time step constraint. Furthermore, we will introduce the multi-resolution WENO limiter into our remapping procedure to make the scheme more robust and have better resolutions without oscillation.

An outline of the rest of this paper is as follows. Section 2 describes the high order DG remapping method in 1D and 2D with the multi-resolution WENO limiter and the positivity-preserving limiter in detail. Numerical results are presented in Section 3 to demonstrate the excellent properties of our remapping algorithm, such as high order accuracy, essentially non-oscillatory performance, and positivity-preserving. Afterward, we apply our remapping algorithm in an indirect ALE method and show its performance on certain benchmark flow problems in Section 4. Finally, concluding remarks are given in Section 5.

2 Remapping algorithm

Suppose $\Omega \subset \mathbb{R}^d$ ($d = 1$ and 2) is a connected bounded computational domain. We assume that the old distorted mesh and the new rezoned mesh are given. Note that these two meshes have the same number of cells and vertices and the same connectivity. The old mesh

is considered as the initial mesh configuration at $t = 0$, denoted as $\mathcal{T}_h^0$. The new mesh is regarded as the final mesh configuration at the final time $t = T$, denoted as $\mathcal{T}_h^T$, where T is the pseudo time we have introduced. For the sake of consistency, we use the superscript 0 or T to be cells, nodes, and node coordinates of the old or new mesh, respectively. The moving mesh is obtained by connecting the corresponding nodes of the old and new meshes with straight lines in $[0, T]$. We assume that u is the function of the variable to be remapped, such as density, momentum or total energy. In order to obtain the high order polynomials on the new mesh, we can solve this trivial equation

$$\frac{\partial u}{\partial t} = 0, \tag{2.1}$$

on the moving mesh, since the function u is a time independent quantity in the remapping procedure. We use the scalar function u as an example to illustrate the remapping techniques.

2.1 Mesh movement velocity

We first need to consider the motion of the mesh node before describing the DG scheme for solving the equation (2.1) on the moving mesh. We introduce a variable ω to describe the mesh movement velocity in one dimension and a variable $\boldsymbol{\omega} = (\omega_x, \omega_y)$ to describe the moving speed in two dimensions.

In one dimension, assume that the old and new mesh generating points are given, denoted as $\left\{x^0_{j-\frac{1}{2}}\right\}_{j=1}^{N}$ and $\left\{x^T_{j-\frac{1}{2}}\right\}_{j=1}^{N}$. We give the definition of the mesh movement velocity

$$\omega_{j-\frac{1}{2}} = \frac{x^T_{j-\frac{1}{2}} - x^0_{j-\frac{1}{2}}}{T}. \tag{2.2}$$

We can get the spatial coordinates of the points $x_{j-\frac{1}{2}}(t)$ from the straight lines

$$x_{j-\frac{1}{2}}(t) = x^0_{j-\frac{1}{2}} + \omega_{j-\frac{1}{2}} t, \quad t \in [0, T].$$

Then the mesh configuration $\mathcal{T}_h(t)$ at any moment can be obtained. Besides, the mesh movement velocity is a piecewise linear mesh velocity function on the computational domain

[17, 18]. For the time-dependent cell $K_j(t) = [x_{j-\frac{1}{2}}(t), x_{j+\frac{1}{2}}(t)]$, it is given by

$$\omega(x,t) = \omega_{j+\frac{1}{2}} \frac{x - x_{j-\frac{1}{2}}(t)}{\triangle_j(t)} + \omega_{j-\frac{1}{2}} \frac{x_{j+\frac{1}{2}}(t) - x}{\triangle_j(t)}, \quad \triangle_j(t) = x_{j+\frac{1}{2}}(t) - x_{j-\frac{1}{2}}(t).$$

Similarly, in two dimensions, the old and new grids are composed of non-overlapping triangular cells $\left\{K_j^0\right\}_{j=1}^N$ and $\left\{K_j^T\right\}_{j=1}^N$, where N is the number of the cells. The three vertices of the triangle are represented by P_l $(l = 1, 2, 3)$. We have the definition of the mesh velocity $\boldsymbol{\omega}_l$ and the node $P_l(t) = (x_l(t), y_l(t))$,

$$\begin{aligned} \omega_{x_l} &= \frac{x_l^T - x_l^0}{T}, \quad x_l(t) = x_l^0 + \omega_{x_l} t,\ t \in [0,T], \\ \omega_{y_l} &= \frac{y_l^T - y_l^0}{T}, \quad y_l(t) = y_l^0 + \omega_{y_l} t,\ t \in [0,T], \end{aligned} \tag{2.3}$$

and these vertices form the time-dependent triangular cell $K_j(t) = \triangle P_1(t)P_2(t)P_3(t)$. The mesh movement velocity at any point on the edge connecting nodes $P_{l_1}(t)$ and $P_{l_2}(t)$ can also be obtained by linear interpolation

$$\begin{aligned} \omega_x(x,y,t) &= \omega_{x_{l_2}} \theta(x,y,t) + \omega_{x_{l_1}} (1 - \theta(x,y,t)), \\ \omega_y(x,y,t) &= \omega_{y_{l_2}} \theta(x,y,t) + \omega_{y_{l_1}} (1 - \theta(x,y,t)), \end{aligned}$$

where

$$\theta(x,y,t) = \frac{\sqrt{\left(x - x_{l_1}(t)\right)^2 + \left(y - y_{l_1}(t)\right)^2}}{\sqrt{\left(x_{l_2}(t) - x_{l_1}(t)\right)^2 + \left(y_{l_2}(t) - y_{l_1}(t)\right)^2}}.$$

The authors pointed out in [17, 18] that the mesh movement velocity should satisfy the following boundedness and Lipschitz continuity requirements to guarantee the accuracy of the scheme on a moving mesh. For instance, in one-dimensional space, we restrict the mesh movement velocity to that which satisfies

$$\max_{(x,t)\in\Omega\times[0,T]} |\omega(x,t)| \leqslant C_0, \qquad \max_{(x,t)\in\Omega\times[0,T]} |\partial_x \omega(x,t)| \leqslant C_{0,1}.$$

Similarly, the mesh movement velocity in two-dimensional space needs to satisfy

$$|\omega_x| \leqslant C_0,\ |\omega_y| \leqslant C_0,\ \left|\frac{\partial \omega_x}{\partial x}\right| \leqslant C_{0,1},\ \left|\frac{\partial \omega_y}{\partial x}\right| \leqslant C_{0,1},\ \left|\frac{\partial \omega_x}{\partial y}\right| \leqslant C_{0,1},\ \left|\frac{\partial \omega_y}{\partial y}\right| \leqslant C_{0,1},$$

for all points (x, y, t) in $\Omega \times [0, T]$. Here C_0 and $C_{0,1}$ are constants independent of the mesh size. We apply $C_0 = C_{0,1} = 10$ to impose these constraints in the following one- and two-dimensional numerical examples. Furthermore, these constraints can be transformed into the constraint of the final pseudo-time T. Specifically, in one dimensional space, T is taken as

$$T = \max_j \left\{ \frac{|x^T_{j-\frac{1}{2}} - x^0_{j-\frac{1}{2}}|}{C_0}, \frac{|(x^T_{j+\frac{1}{2}} - x^0_{j+\frac{1}{2}}) - (x^T_{j-\frac{1}{2}} - x^0_{j-\frac{1}{2}})|}{C_{0,1}\left(x^0_{j+\frac{1}{2}} - x^0_{j-\frac{1}{2}}\right)}, \frac{|(x^T_{j+\frac{1}{2}} - x^0_{j+\frac{1}{2}}) - (x^T_{j-\frac{1}{2}} - x^0_{j-\frac{1}{2}})|}{C_{0,1}\left(x^T_{j+\frac{1}{2}} - x^T_{j-\frac{1}{2}}\right)} \right\}.$$

We can also get T in the similar way in two dimensions. After selecting a suitable final pseudo-time T, we can obtain the mesh movement velocity according to (2.2) and (2.3).

2.2 High order spatial discretization

We define the approximation space as

$$\mathcal{V}_{h,d}(t) = \{v \in L^2(\Omega) : \, v \in P^k(K_j(t)), \, \forall K_j(t) \in \mathcal{T}_h(t)\},$$

where the index $d \in \{1, 2\}$ denotes the spatial dimension, $P^k(K_j(t))$ is the space of polynomials of degree at most k on $K_j(t)$. The space $\mathcal{V}_{h,d}(t)$ contains discontinuous functions. For $v \in \mathcal{V}_{h,1}(t)$, the values at the cell boundaries of $x_{j-\frac{1}{2}}(t)$ are defined as

$$v^-_{j-\frac{1}{2}} = \lim_{\epsilon \to 0^+} v(x_{j-\frac{1}{2}}(t) - \epsilon), \quad v^+_{j-\frac{1}{2}} = \lim_{\epsilon \to 0^+} v(x_{j-\frac{1}{2}}(t) + \epsilon).$$

In two dimensions, we define the values of the function $v \in \mathcal{V}_{h,2}(t)$ at any point on the cell boundary $e \in \partial K_j(t)$ with the outer normal vector $\boldsymbol{n}_e$ as

$$v^{int}|_e = \lim_{\epsilon \to 0^+} v(\boldsymbol{x} - \epsilon \boldsymbol{n}_e), \quad v^{ext}|_e = \lim_{\epsilon \to 0^+} v(\boldsymbol{x} + \epsilon \boldsymbol{n}_e).$$

In the discontinuous Galerkin method, the numerical solution $u_h(t) \in \mathcal{V}_{h,d}(t)$ is a piecewise polynomial which can be denoted as

$$u_h(\boldsymbol{x}, t)|_{K_j(t)} = \sum_{r=1}^{n_b} u_r^{K_j(t)}(t) \phi_r^{K_j(t)}(\boldsymbol{x}, t), \; \boldsymbol{x} \in K_j(t), \; \forall K_j(t) \in \mathcal{T}_h(t),$$

where $n_b = \frac{(k+d)!}{d!k!}$ is the number of basis functions and $\left\{\phi_1^{K_j(t)}(\boldsymbol{x},t),\cdots,\phi_{n_b}^{K_j(t)}(\boldsymbol{x},t)\right\}$ is a basis of the space $P^k(K_j(t))$. The coefficients $u_r^{K_j(t)}(t)$, $r=1,\cdots,n_b$ are the unknowns of the method.

Nest, we give the following lemma [18], which will lead to the semi-discrete scheme.

Lemma 2.1. *Let $d \in 1,2$ and $u:\Omega\times[0,T]\to\mathbb{R}$ be a sufficiently smooth function in any cell $K_j(t)\in\mathcal{T}_h(t)$. Then for all $v\in V_{h,d}(t)$ holds the transport equation*

$$\frac{d}{dt}\left(u,v\right)_{K_j(t)} = \left(\partial_t u, v\right)_{K_j(t)} + \left(\nabla\cdot\left(\boldsymbol{\omega}u\right),v\right)_{K_j(t)}. \tag{2.4}$$

We multiply the equation (2.1) by a test function $v\in\mathcal{V}_{h,d}(t)$ and apply the transport equation (2.4) as well as the integration by parts. The one-dimensional semi-discrete scheme can be written as follows: Find a function $u_h\in\mathcal{V}_{h,1}(t)$ such that for all $v\in\mathcal{V}_{h,1}(t)$ and $j=1,\cdots,N$

$$\frac{d}{dt}\left(u_h,v\right)_{K_j(t)} = \left(f(\omega,u_h),\partial_x v\right)_{K_j(t)} - \hat{f}(\omega_{j+\frac{1}{2}},u^-_{h,j+\frac{1}{2}},u^+_{h,j+\frac{1}{2}})v^-_{j+\frac{1}{2}} + \hat{f}(\omega_{j-\frac{1}{2}},u^-_{h,j-\frac{1}{2}},u^+_{h,j-\frac{1}{2}})v^+_{j-\frac{1}{2}}, \tag{2.5}$$

where $f(\omega,u)=-\omega u$ and $\hat{f}$ is the numerical flux. Here we use the upwind flux (Roe flux) numerical flux to reduce the numerical dissipation,

$$\hat{f}(\omega_{j+\frac{1}{2}},u^-_{h,j+\frac{1}{2}},u^+_{h,j+\frac{1}{2}}) = \begin{cases} f(\omega_{j+\frac{1}{2}},u^-_{h,j+\frac{1}{2}}), & \omega_{j+\frac{1}{2}}\leqslant 0, \\ f(\omega_{j+\frac{1}{2}},u^+_{h,j+\frac{1}{2}}), & \omega_{j+\frac{1}{2}}>0. \end{cases} \tag{2.6}$$

The two-dimensional semi-discrete scheme can be written as follows: Find a function $u_h\in\mathcal{V}_{h,2}(t)$ such that

$$\frac{d}{dt}\left(u_h,v\right)_{K_j(t)} = \left(\boldsymbol{f}(\boldsymbol{\omega},u_h),\nabla v\right)_{K_j(t)} - \left\langle \hat{\boldsymbol{f}}\left(\boldsymbol{\omega},u_h^{int},u_h^{ext},\boldsymbol{n}\right),v^{int}\right\rangle_{\partial K_j(t)}, \tag{2.7}$$

for all $v\in\mathcal{V}_{h,2}(t)$ and cells, where $\boldsymbol{f}(\boldsymbol{\omega},u)=-(\omega_x u,\omega_y u)^T$, $\boldsymbol{n}_{K_j(t)}$ is the unit outward normal to the cell boundary $\partial K_j(t)$. The upwind flux is defined as

$$\hat{\boldsymbol{f}}(\boldsymbol{\omega},u_h^{int},u_h^{ext},\boldsymbol{n}) = \begin{cases} \boldsymbol{f}(\boldsymbol{\omega},u_h^{int})\cdot\boldsymbol{n}, & \boldsymbol{\omega}\cdot\boldsymbol{n}\leqslant 0, \\ \boldsymbol{f}(\boldsymbol{\omega},u_h^{ext})\cdot\boldsymbol{n}, & \boldsymbol{\omega}\cdot\boldsymbol{n}>0. \end{cases}$$

2.3 Fully discrete scheme

In this paper, we will adopt the high order strong stability preserving Runge-Kutta (SSP-RK) method for the time discretization, which is also known as total variation diminishing Runge-Kutta (TVD-RK) method. We use the two-dimensional case as an illustration. The one-dimensional case is similar to the two-dimensional case. Note that the motion of the vertices causes the basis function to depend on t. We first introduce the connection between test functions at different time levels and their counterparts on the reference element. The vertices of the two-dimensional triangle reference cell are $(0,0),(1,0)$ and $(0,1)$. The following time-dependent linear mapping can map the time-dependent cells $K_j(t)$ to the time-independent reference element $\mathcal{K}$

$$\mathcal{X}_{K_j(t)} : \mathcal{K} \to K_j(t), \quad \mathcal{X}_{K_j(t)}(\boldsymbol{\xi}, t) = \boldsymbol{A}_{K_j(t)}\boldsymbol{\xi} + P_{l_1}(t), \tag{2.8}$$

where the matrix $\boldsymbol{A}_{K_j(t)}$ and its corresponding determinant $J_{K_j(t)}$ are given by

$$\boldsymbol{A}_{K_j(t)} = (P_{l_2}(t) - P_{l_1}(t), P_{l_3}(t) - P_{l_1}(t)), \quad J_{K_j(t)} = \det\left(\boldsymbol{A}_{K_j(t)}\right) = 2|K_j(t)|,$$

where $P_{l_1}(t), P_{l_2}(t), P_{l_3}(t)$ are the three vertexes of cell $K_j(t)$ and $|K_j(t)|$ is the area of $K_j(t)$.

Now, the two-dimensional fully discrete scheme is written as follows

$$\begin{cases} \left(J_{K_j^{(1)}} u_h^{(1),*}, v^*\right)_{\mathcal{K}} = \left(J_{K_j^n} u_h^{n,*}, v^*\right)_{\mathcal{K}} + \triangle t^n \mathcal{G}\left(u_h^{n,*}, v^*, t_n\right), \\ \left(J_{K_j^{(2)}} u_h^{(2),*}, v^*\right)_{\mathcal{K}} = \dfrac{3}{4}\left(J_{K_j^n} u_h^{n,*}, v^*\right)_{\mathcal{K}} + \dfrac{1}{4}\left(J_{K_j^{(1)}} u_h^{(1),*}, v^*\right) \\ \qquad\qquad + \dfrac{1}{4}\triangle t^n \mathcal{G}\left(u_h^{(1),*}, v^*, t_{n+1}\right), \\ \left(J_{K_j^{n+1}} u_h^{n+1,*}, v^*\right)_{\mathcal{K}} = \dfrac{1}{3}\left(J_{K_j^n} u_h^{n,*}, v^*\right)_{\mathcal{K}} + \dfrac{2}{3}\left(J_{K_j^{(2)}} u_h^{(2),*}, v^*\right) \\ \qquad\qquad + \dfrac{2}{3}\triangle t^n \mathcal{G}\left(u_h^{(2),*}, v^*, t_{n+\frac{1}{2}}\right). \end{cases} \tag{2.9}$$

Here $u_h^* = u_h \circ \mathcal{X}_{K_j(t)}$ and $v_h^* = v_h \circ \mathcal{X}_{K_j(t)}$ are defined on the reference cell. $u_h^{(1)}, J_{K_j^{(1)}}, u_h^{n+1}, J_{K_j^{n+1}}$ are the values at $t = t_{n+1}$; and $u_h^{(2)}, J_{K_j^{(2)}}$ are the values at $t = t_{n+\frac{1}{2}}$. The operator $\mathcal{G}$ represents the terms at the right hand side of (2.7), namely

$$\begin{aligned} \mathcal{G}\left(u_h^*, v^*, t\right) = & \left(J_{K_j(t)} \boldsymbol{A}_{K_j(t)}^{-1} \boldsymbol{f}(\boldsymbol{\omega}, u_h^*), \nabla_{\boldsymbol{\xi}} v^*\right)_{\mathcal{K}} \\ & - \left\langle \hat{\boldsymbol{f}}\left(\boldsymbol{\omega}, u_h^{int,*}, u_h^{ext,*}, J_{K_j(t)} \boldsymbol{A}_{K_j(t)}^{-T} \boldsymbol{n}_{\mathcal{K}}\right), v^{int,*} \right\rangle_{\partial\mathcal{K}}. \end{aligned}$$

To ensure the stability of the scheme, the time step size $\triangle t^n$ is determined by the CFL condition

$$\triangle t^n \leqslant \frac{C_{cfl} h}{\alpha}, \quad \alpha = \max_{j,\, e \in \partial K_j^n} |\boldsymbol{\omega}_e \cdot \boldsymbol{n}_e| \tag{2.10}$$

where C_{cfl} is a constant typically chosen to be less than $1/(2k+1)$ and we choose it as 0.15 for $k = 2$ in our numerical tests. h is the minimum diameter of the inscribed circles for all the cells on the computational mesh. The stability restriction (2.10) will lead to a very large time step size $\triangle t^n$ if α is very small, which will destroy the temporal accuracy. In such cases, it is necessary to reduce $\triangle t^n$ in order to ensure temporal accuracy. Referring to [27], in our numerical experiments, we set $\triangle t^n$ to $1.5h$ if the value produced by (2.10) is more than that.

2.4 Multi-resolution WENO limiter

The DG method can simulate the weak discontinuity well without any modification while the nonlinear limiters must be applied to control the spurious oscillations near strong discontinuities such as the strong shocks or interfaces. First, we need to use an indicator to identify the troubled cells, which may contain strong discontinuities. In this paper, we apply the shock detection technique [12, 19, 37] to detect troubled cells and apply the limiter on those cells. The new type of multi-resolution WENO limiter [44, 45] builds a series of hierarchical L^2 projection polynomials from zeroth degree to the highest degree using the information of the DG solution essentially just within the troubled cell itself. The new polynomial modified by the multi-resolution WENO limiter is a convex combination of polynomials of different degrees, and the associated linear weights can be any positive numbers with the only requirement that their summation equals one. Such new limiters can maintain the original order of accuracy in smooth regions and simultaneously suppress spurious oscillations near strong discontinuities.

In order to save space, we take the third-order multi-resolution WENO limiter on triangular meshes as an example to discuss the specific procedure. Now we assume Δ_0 is a

troubled cell and its neighboring triangular cells are marked as $\Delta_1, \Delta_2, \Delta_3$. The procedure to reconstruct a new polynomial on the troubled cell Δ_0 includes the following steps.

Step 1. First, we construct a series of polynomials $q_\ell(x,y), \ell = 0, 1, 2$, which satisfy

$$\int_{\Delta_0} q_\ell(x,y) v_r^{(0)}(x,y) dxdy = \int_{\Delta_0} u_h(x,y) v_r^{(0)}(x,y) dxdy, \quad r = 1, \cdots, n_b.$$

where $v_r^{(0)}(x,y)$ is the basis function defined on the triangle Δ_0.

Step 2. We take $p_{0,1}(x,y) = q_0(x,y)$ and define the equivalent expression of the high order polynomial as follows

$$p_{1,1}(x,y) = \frac{1}{\gamma_{1,1}} q_1(x,y) - \frac{\gamma_{0,1}}{\gamma_{1,1}} p_{0,1}(x,y),$$
$$p_{2,2}(x,y) = \frac{1}{\gamma_{2,2}} q_2(x,y) - \frac{\gamma_{1,2}}{\gamma_{2,2}} p_{1,2}(x,y),$$

where

$$p_{1,2}(x,y) = w_{1,1} p_{1,1}(x,y) + w_{0,1} p_{0,1}(x,y),$$

with the linear weights $\gamma_{\ell-1,\ell} + \gamma_{\ell,\ell} = 1, \ell = 1, 2$. $w_{0,1}$ and $w_{1,1}$ are the nonlinear weights which will be explicitly described later. To make a balance between the accuracy in smooth regions and the sharp and essentially non-oscillatory shock transitions in non-smooth regions, we set $\gamma_{\ell-1,\ell} = 0.01$ and $\gamma_{\ell,\ell} = 0.99$.

Step 3. Compute the smoothness indicators β_{ℓ,ℓ_2} which measure how smooth the function $p_{\ell,\ell_2}(x,y)$ are in the cell Δ_0.

$$\beta_{\ell,\ell_2} = \sum_{|\alpha|=1}^{\ell} \int_{\Delta_0} \Delta_0^{|\alpha|-1} \left(\frac{\partial^{|\alpha|}}{\partial x^{\alpha_1} \partial y^{\alpha_2}} p_{\ell,\ell_2}(x,y) \right)^2 dxdy, \quad \ell = \ell_2 - 1, \ell_2; \quad \ell_2 = 1, 2,$$

where $\alpha = (\alpha_1, \alpha_2)$ and $|\alpha| = \alpha_1 + \alpha_2$. We take another way to calculate $\beta_{0,1}$ because the value obtained by the above formula is 0. First of all, we use the L^2 projection methodology to define the linear polynomials on the adjacent cells respectively, satisfying the following conditions

$$\int_{\Delta_1} q_{0,1}(x,y) v_r^{(1)}(x,y) dxdy = \int_{\Delta_1} u_h(x,y) v_r^{(1)}(x,y) dxdy, \quad r = 1, 2, 3,$$
$$\int_{\Delta_2} q_{0,2}(x,y) v_r^{(2)}(x,y) dxdy = \int_{\Delta_2} u_h(x,y) v_r^{(2)}(x,y) dxdy, \quad r = 1, 2, 3,$$
$$\int_{\Delta_3} q_{0,3}(x,y) v_r^{(3)}(x,y) dxdy = \int_{\Delta_3} u_h(x,y) v_r^{(3)}(x,y) dxdy, \quad r = 1, 2, 3.$$

The associated smoothness indicators are

$$\zeta_{0,1} = \int_{\Delta_0} \left(\frac{\partial}{\partial x} q_{0,1}(x,y)\right)^2 + \left(\frac{\partial}{\partial y} q_{0,1}(x,y)\right)^2 dxdy,$$
$$\zeta_{0,2} = \int_{\Delta_0} \left(\frac{\partial}{\partial x} q_{0,2}(x,y)\right)^2 + \left(\frac{\partial}{\partial y} q_{0,2}(x,y)\right)^2 dxdy,$$
$$\zeta_{0,3} = \int_{\Delta_0} \left(\frac{\partial}{\partial x} q_{0,3}(x,y)\right)^2 + \left(\frac{\partial}{\partial y} q_{0,3}(x,y)\right)^2 dxdy.$$

Then we define $\beta_{0,1}$ as

$$\beta_{0,1} = \min\left(\zeta_{0,1}, \zeta_{0,2}, \zeta_{0,3}\right).$$

Step 4. Based on the linear weights and the smoothness indicators, we compute the nonlinear weights using the WENO-Z recipe [4]. The nonlinear weights are given as

$$w_{\ell_1,\ell_2} = \frac{\bar{w}_{\ell_1,\ell_2}}{\sum_{\ell=\ell_2-1}^{\ell_2} \bar{w}_{\ell,\ell_2}},\ \bar{w}_{\ell_1,\ell_2} = \gamma_{\ell_1,\ell_2}\left(1 + \frac{\tau_{\ell_2}}{\varepsilon + \beta_{\ell_1,\ell_2}}\right), \quad \ell_1 = \ell_2 - 1, \ell_2;\ \ell_2 = 1,2$$

where

$$\tau_{\ell_2} = \left(\beta_{\ell_2,\ell_2} - \beta_{\ell_2-1,\ell_2}\right)^2, \quad \ell_2 = 1,2,$$

and ε is taken as 10^{-10} in our code.

Step 5. The new polynomial on the trouble cell $\triangle_0$ is given by

$$u_h^{new}(x,y) = w_{1,2}p_{1,2}(x,y) + w_{2,2}p_{2,2}(x,y).$$

Therefore, we obtain the modified polynomial $u_h^{new}(x,y)$ which is virtually non-oscillatory and as accurate as the original polynomial $u_h(x,y)$.

2.5 The positivity-preserving property

To prevent the failure of numerical solutions, the remapping algorithm, especially those applied in the indirect ALE method to compute fluid flow problems, must maintain the positive property of certain physical parameters, such as density and internal energy. However, even with the WENO limiter, our above remapping algorithm does not automatically keep the remapping variables positive. Following the widely used positivity-preserving framework [41, 42], we first prove our proposed first order remapping algorithm is positive under

an appropriate time step and then prove the cell average of numerical solution from the high order remapping algorithm can ensure the positivity of density and internal energy under the suitable time step.

We define the set of admissible states by

$$G = \left\{ \boldsymbol{U} = (\rho,\, \boldsymbol{m},\, E)^T,\ \rho > 0,\ e > 0 \right\},$$

where ρ is the density, $\boldsymbol{m}$ is the momentum which in the one-dimensional case is a scalar and in the two-dimensional case is a vector, E is the total energy and $e = E - \frac{1}{2}\frac{|\boldsymbol{m}|^2}{\rho}$ is the internal energy. G can be proven to be a convex set. A scheme is called positivity-preserving if the new state updated by the scheme also belongs to G when starting from a set of physically admissible states.

2.5.1 The first order positivity-preserving remapping algorithm

In this part, we will demonstrate that the first order remapping algorithm is positivity-preserving under the appropriate time step when the upwind flux and Euler forward method are used in one- and two-dimensional space, respectively. Taking the Euler forward time discretization for the one-dimensional semi-discrete scheme (2.5) based on P^0 approximation and taking the test function $v = 1$, the scheme can be written in the finite volume form as

$$\bar{\boldsymbol{U}}_{h,j}^{n+1} = \frac{\triangle_j^n}{\triangle_j^{n+1}} \bar{\boldsymbol{U}}_{h,j}^n - \frac{\triangle t}{\triangle_j^{n+1}} \left[\hat{f}(\omega_{j+\frac{1}{2}}, \boldsymbol{U}_{h,j}^n, \boldsymbol{U}_{h,j+1}^n) - \hat{f}(\omega_{j-\frac{1}{2}}, \boldsymbol{U}_{h,j-1}^n, \boldsymbol{U}_{h,j}^n) \right]. \tag{2.11}$$

Here, $\bar{\boldsymbol{U}}_{h,j}^n = \frac{1}{\triangle_j^n} \int_{K_j^n} \boldsymbol{U}_{h,j}^n dx$ denotes the cell average value of the polynomial $\boldsymbol{U}_{h,j}^n$ on the cell K_j^n. For P^0 approximation, the solution $\boldsymbol{U}_{h,j}^n$ equals to the value of the cell average $\bar{\boldsymbol{U}}_{h,j}^n$. $\triangle_j^n$ is the cell length of K_j^n. To prove the positivity-preserving property for the fully discrete scheme (2.11), we will show that $\bar{\boldsymbol{U}}_{h,j}^{n+1}$ is in the set of G when $\bar{\boldsymbol{U}}_{h,j-1}^n, \bar{\boldsymbol{U}}_{h,j}^n$ and $\bar{\boldsymbol{U}}_{h,j+1}^n$ are all in the set of G. There is another expression of the upwind flux (2.6)

$$\hat{f}(\omega_{j+\frac{1}{2}}, \boldsymbol{U}_{h,j}^n, \boldsymbol{U}_{h,j+1}^n) = \frac{1}{2} \left(-\omega_{j+\frac{1}{2}} (\boldsymbol{U}_{h,j}^n + \boldsymbol{U}_{h,j+1}^n) - |\omega_{j+\frac{1}{2}}| (\boldsymbol{U}_{h,j+1}^n - \boldsymbol{U}_{h,j}^n) \right). \tag{2.12}$$

Substitute the (2.12) to the equation (2.11) and we have

$$\bar{\boldsymbol{U}}_{h,j}^{n+1} = \left[\frac{\triangle_j^n}{\triangle_j^{n+1}} + \frac{\triangle t}{\triangle_j^{n+1}}\left(\frac{\omega_{j+\frac{1}{2}} - \omega_{j-\frac{1}{2}}}{2} - \frac{|\omega_{j+\frac{1}{2}}| + |\omega_{j-\frac{1}{2}}|}{2}\right)\right]\bar{\boldsymbol{U}}_{h,j}^n + \frac{\triangle t}{\triangle_j^{n+1}}\left(\frac{\omega_{j+\frac{1}{2}}}{2} + \frac{|\omega_{j+\frac{1}{2}}|}{2}\right)\bar{\boldsymbol{U}}_{h,j+1}^n + \frac{\triangle t}{\triangle_j^{n+1}}\left(\frac{|\omega_{j-\frac{1}{2}}|}{2} - \frac{\omega_{j-\frac{1}{2}}}{2}\right)\bar{\boldsymbol{U}}_{h,j-1}^n.$$

Under the time step restriction

$$\triangle t^n \leqslant \frac{1}{2}\frac{\min\limits_j \triangle_j^n}{\max\limits_j |\omega_{j+\frac{1}{2}}|}, \tag{2.13}$$

we notice that $\bar{\boldsymbol{U}}_{h,j}^{n+1}$ is a convex combination of $\bar{\boldsymbol{U}}_{h,j-1}^n, \bar{\boldsymbol{U}}_{h,j}^n$ and $\bar{\boldsymbol{U}}_{h,j+1}^n$.

Theorem 2.1. *Assume* $\left\{\bar{\boldsymbol{U}}_{h,j}^n \in G, \forall j = 1, \cdots, N\right\}$*, the scheme* (2.11) *is positivity-preserving, namely,* $\left\{\bar{\boldsymbol{U}}_{h,j}^{n+1} \in G, \forall j = 1, \cdots, N\right\}$ *under the time step restriction* (2.13)*.*

Next, we take into account the first order remapping algorithm in two space dimensions. Based on P^0 approximation and Euler forward time discretization, we take the test function $v = 1$ and get the following form

$$\bar{\boldsymbol{U}}_h^{n+1} = \frac{|K^n|}{|K^{n+1}|}\bar{\boldsymbol{U}}_h^n - \frac{\triangle t}{|K^{n+1}|}\sum_{i=1}^{3}\hat{\boldsymbol{f}}(\boldsymbol{\omega}_i, \boldsymbol{U}_h^n, \boldsymbol{U}_{h,i}^n, \boldsymbol{n}_i) l_i. \tag{2.14}$$

Here we ignore the index j of the triangular element for simplicity; i.e., we will write K_j^n as K^n. $\bar{\boldsymbol{U}}_h^n$ stands for the cell average value of $\boldsymbol{U}_h^n$ in the cell K^n and we use the cell average instead of the numerical solution in this part. $\boldsymbol{U}_{h,i}^n, i = 1, 2, 3$ are the solutions denoted on K_i^n which are the neighbor cells of the triangular cell K^n along the i-th edge e_i. l_i is the length of the edge e_i with the outward normal vector $\boldsymbol{n}_i$. $\boldsymbol{\omega}_i$ is the value of the mesh movement velocity at the middle point of the edge. We also rewrite the upwind flux similar to the form of the Lax-Friedrichs flux and substitute it into the equation (2.14),

$$\bar{\boldsymbol{U}}_h^{n+1} = \frac{|K^n|}{|K^{n+1}|}\left(1 - \frac{\triangle t}{|K^n|}\sum_{i=1}^{3}|\boldsymbol{\omega}_i \cdot \boldsymbol{n}_i| l_i\right)\bar{\boldsymbol{U}}_h^n + \frac{\triangle t}{|K^{n+1}|}\sum_{i=1}^{3}\frac{1}{2}\left(|\boldsymbol{\omega}_i \cdot \boldsymbol{n}_i| + (\boldsymbol{\omega}_i \cdot \boldsymbol{n}_i)\right)(\bar{\boldsymbol{U}}_h^n + \bar{\boldsymbol{U}}_{h,i}^n) l_i.$$

We constrain the time step to permit the cell average value $\bar{\boldsymbol{U}}_h^{n+1}$ to be stated as a convex combination of elements in G. As a result, we can derive the following theorem.

Theorem 2.2. *Assume $\bar{\boldsymbol{U}}_h^n \in G$ for all elements K^n, the scheme (2.14) is positivity-preserving under the time step restriction*

$$\triangle t^n \max_{K^n} \left\{ \frac{\sum_{i=1}^{3} |\boldsymbol{\omega_i} \cdot \boldsymbol{n}_i| l_i}{|K^n|} \right\} \leqslant 1.$$

2.5.2 The high order positivity-preserving remapping algorithm

Similar to the derivation of the positivity-preserving property of the high order remapping algorithm on the quadrilateral meshes [15], we first prove that under the condition of appropriate time step, the cell average values obtained by the high order scheme will be in the set G. Afterwards, the numerical solutions are modified in conjunction with the positivity-preserving limiter developed by Zhang and Shu [41] so that the values of the numerical solutions on the quadrature points preserve nonnegativity. We limit our discussion to the Euler forward time discretization to conserve space. As a convex combination of the Euler forward scheme, the following results will still hold for the third-order Runge-Kutta method.

We first consider the general scheme provided by the Euler forward time discretization of the one-dimensional semi-discrete formulation (2.5) with the test function $v = 1$

$$\bar{\boldsymbol{U}}_{h,j}^{n+1} = \frac{\triangle_j^n}{\triangle_j^{n+1}} \bar{\boldsymbol{U}}_{h,j}^n - \frac{\triangle t}{\triangle_j^{n+1}} \left[\hat{f}(\omega_{j+\frac{1}{2}}, \boldsymbol{U}_{h,j+\frac{1}{2}}^{n,-}, \boldsymbol{U}_{h,j+\frac{1}{2}}^{n,+}) - \hat{f}(\omega_{j-\frac{1}{2}}, \boldsymbol{U}_{h,j-\frac{1}{2}}^{n,-}, \boldsymbol{U}_{h,j-\frac{1}{2}}^{n,+}) \right]. \quad (2.15)$$

We apply the Gauss-Lobatto quadrature rule with q_N points on the interval K_j to decompose the cell average, where q_N is the smallest integer satisfying $2q_N - 3 \geqslant k$. The integration points are denoted by $S_j = \left\{ x_{j-\frac{1}{2}} = \hat{x}_j^1 < \hat{x}_j^2 \cdots < \hat{x}_j^{q_N} = x_{j+\frac{1}{2}} \right\}$ and the associated quadrature weights are denoted by $\{\hat{c}_q\}_{q=1}^{q_N}$ which satisfy $\frac{1}{2} \sum_{q=1}^{q_N} \hat{c}_q = 1$. The q_N-point Gauss-Lobatto rule is exact for the polynomial $\boldsymbol{U}_{h,j}(x)$, which implies

$$\bar{\boldsymbol{U}}_{h,j}^n = \sum_{q=1}^{q_N} \frac{\hat{c}_q}{2} \boldsymbol{U}_{h,j}^n(\hat{x}_j^q) = \sum_{q=2}^{q_N-1} \frac{\hat{c}_q}{2} \boldsymbol{U}_{h,j}^{n,q} + \frac{\hat{c}_1}{2} \boldsymbol{U}_{h,j-\frac{1}{2}}^{n,+} + \frac{\hat{c}_{q_N}}{2} \boldsymbol{U}_{h,j+\frac{1}{2}}^{n,-} \quad (2.16)$$

Combining the equations (2.12),(2.16), we can derive that

$$\begin{aligned}\bar{\boldsymbol{U}}_{h,j}^{n+1} =&\frac{\triangle_j^n}{\triangle_j^{n+1}}\sum_{q=2}^{q_N-1}\frac{\hat{c}_q}{2}\boldsymbol{U}_{h,j}^{n,q}+\frac{\triangle t}{\triangle_j^{n+1}}\left[\frac{1}{2}\left(\omega_{j+\frac{1}{2}}+|\omega_{j+\frac{1}{2}}|\right)\boldsymbol{U}_{h,j+\frac{1}{2}}^{n,+}+\frac{1}{2}\left(|\omega_{j-\frac{1}{2}}|-\omega_{j-\frac{1}{2}}\right)\boldsymbol{U}_{h,j-\frac{1}{2}}^{n,-}\right]\\&+\frac{\triangle t}{\triangle_j^{n+1}}\left[\frac{1}{2}\left(\omega_{j+\frac{1}{2}}+|\omega_{j+\frac{1}{2}}|\right)\boldsymbol{U}_{h,j+\frac{1}{2}}^{n,-}+\frac{1}{2}\left(|\omega_{j-\frac{1}{2}}|-\omega_{j-\frac{1}{2}}\right)\boldsymbol{U}_{h,j-\frac{1}{2}}^{n,+}\right]\\&+\frac{\triangle_j^n}{\triangle_j^{n+1}}\frac{\hat{c}_{q_N}}{2}\left(1-\frac{\triangle t}{\triangle_j^n}\frac{|\omega_{j+\frac{1}{2}}|}{\hat{c}_{q_N}/2}\right)\boldsymbol{U}_{h,j+\frac{1}{2}}^{n,-}+\frac{\triangle_j^n}{\triangle_j^{n+1}}\frac{\hat{c}_1}{2}\left(1-\frac{\triangle t}{\triangle_j^n}\frac{|\omega_{j-\frac{1}{2}}|}{\hat{c}_1/2}\right)\boldsymbol{U}_{h,j-\frac{1}{2}}^{n,+}.\end{aligned}$$

If we use the positivity-preserving limiter introduced later to implement $\boldsymbol{U}_{h,j}^{n,q}\in G$ for all $q=1,\cdots,q_N$ and $j=1,\cdots,N$ and restrict the time step to satisfy a suitable condition, it is easy to conclude that $\bar{\boldsymbol{U}}_{h,j}^{n+1}\in G$. Note that $\hat{c}_1=\hat{c}_{q_N}$ and we can obtain the following theorem.

Theorem 2.3. *Assume $\boldsymbol{U}_{h,j}^{n,q}\in G$ for all $q=1,\cdots,q_N$ and $j=1,\cdots,N$, the scheme* (2.15) *is positivity-preserving under the time step restriction*

$$\triangle t\leqslant\frac{\hat{c}_1}{2}\frac{\min\limits_j\triangle_j^n}{\max\limits_j|\omega_{j+\frac{1}{2}}|}.$$

Next, we consider the sufficient condition to enable our high order remapping algorithm in two space dimensions to have the positivity-preserving property. We begin with a special quadrature rule which includes the Gauss quadrature points for the edges. The cell average $\bar{\boldsymbol{U}}_h^n$ can be represented by

$$\bar{\boldsymbol{U}}_h^n=\sum_{i=1}^{3}\sum_{\beta=1}^{k+1}\frac{2}{3}c_\beta\hat{c}_1\boldsymbol{U}_{i,\beta}^{int}+\sum_{\gamma=1}^{L}\tilde{c}_\gamma\boldsymbol{U}_\gamma^{\widetilde{int}},\tag{2.17}$$

where $\boldsymbol{U}_\gamma^{\widetilde{int}}$ and $\boldsymbol{U}_{i,\beta}^{int}$ indicate the values of the numerical solution $\boldsymbol{U}_h$ at the quadrature points inside and on the boundary of cell K, respectively. We define the set of all the quadrature points on the cell K as S_K. For relevant weights $c_\beta,\hat{c}_1,\tilde{c}_\gamma$ and more detailed quadrature rules, we refer to [42]. The scheme satisfied by the cell averages of the two-dimensional remapping method using the first order Euler forward time discretization can be written as

$$\bar{\boldsymbol{U}}_h^{n+1}=\frac{|K^n|}{|K^{n+1}|}\bar{\boldsymbol{U}}_h^n-\frac{\triangle t}{|K^{n+1}|}\sum_{i=1}^{3}\int_{e_i}\hat{\boldsymbol{f}}(\boldsymbol{\omega},\boldsymbol{U}_h^{int},\boldsymbol{U}_h^{ext},\boldsymbol{n}_i)dl.\tag{2.18}$$

The edge integrals can be approximated by a $k+1-$point Gauss quadrature formula and we have

$$\bar{\boldsymbol{U}}_h^{n+1} = \frac{|K^n|}{|K^{n+1}|}\bar{\boldsymbol{U}}_h^n - \frac{\triangle t}{|K^{n+1}|}\sum_{i=1}^{3}\sum_{\beta=1}^{k+1}\hat{\boldsymbol{f}}(\boldsymbol{\omega}_{i,\beta}, \boldsymbol{U}_{i,\beta}^{int}, \boldsymbol{U}_{i,\beta}^{ext}, \boldsymbol{n}_i)c_\beta l_i,$$

where $\boldsymbol{U}_{i,\beta}^{int}$ and $\boldsymbol{U}_{i,\beta}^{ext}$ are the values of the DG solutions $\boldsymbol{U}_h^n$ evaluated in the $\beta-$th Gauss quadrature point on the edge e_i. Similarly, $\boldsymbol{\omega}_{i,\beta}$ are the values of the mesh movement velocity $\boldsymbol{\omega}$ calculated in the $\beta-$th Gauss quadrature point on the edge e_i. The corresponding quadrature weights are denoted as c_β.

By some algebraic manipulations, $\bar{\boldsymbol{U}}_h^{n+1}$ can be rewritten as

$$\bar{\boldsymbol{U}}_h^{n+1} = \frac{|K^n|}{|K^{n+1}|}\sum_{\gamma=1}^{L}\tilde{c}_\gamma \boldsymbol{U}_\gamma^{int} + \sum_{\beta=1}^{k+1}\frac{2}{3}c_\beta\hat{c}_1\left(\mathbf{H}_{1,\beta}+\mathbf{H}_{2,\beta}+\mathbf{H}_{3,\beta}\right)$$

where

$$\begin{aligned}
\mathbf{H}_{1,\beta} =& \frac{|K^n|}{|K^{n+1}|}\boldsymbol{U}_{1,\beta}^{int} - \frac{3\triangle t}{2\hat{c}_1|K^{n+1}|}\left[\hat{\boldsymbol{f}}(\boldsymbol{\omega}_{1,\beta}, \boldsymbol{U}_{1,\beta}^{int}, \boldsymbol{U}_{1,\beta}^{ext}, \boldsymbol{n}_1)l_1 - \hat{f}(\boldsymbol{\omega}_{1,\beta}, \boldsymbol{U}_{2,\beta}^{int}, \boldsymbol{U}_{1,\beta}^{int}, \boldsymbol{n}_1)l_1\right],\\
\mathbf{H}_{2,\beta} =& \frac{|K^n|}{|K^{n+1}|}\boldsymbol{U}_{2,\beta}^{int} - \frac{3\triangle t}{2\hat{c}_1|K^{n+1}|}\left[\hat{\boldsymbol{f}}(\boldsymbol{\omega}_{1,\beta}, \boldsymbol{U}_{2,\beta}^{int}, \boldsymbol{U}_{1,\beta}^{int}, \boldsymbol{n}_1)l_1 + \hat{\boldsymbol{f}}(\boldsymbol{\omega}_{2,\beta}, \boldsymbol{U}_{2,\beta}^{int}, \boldsymbol{U}_{2,\beta}^{ext}, \boldsymbol{n}_2)l_2\right.\\
&\left.+\hat{\boldsymbol{f}}(\boldsymbol{\omega}_{3,\beta}, \boldsymbol{U}_{2,\beta}^{int}, \boldsymbol{U}_{3,\beta}^{int}, \boldsymbol{n}_3)l_3\right],\\
\mathbf{H}_{3,\beta} =& \frac{|K^n|}{|K^{n+1}|}\boldsymbol{U}_{3,\beta}^{int} - \frac{3\triangle t}{2\hat{c}_1|K^{n+1}|}\left[\hat{\boldsymbol{f}}(\boldsymbol{\omega}_{3,\beta}, \boldsymbol{U}_{3,\beta}^{int}, \boldsymbol{U}_{3,\beta}^{ext}, \boldsymbol{n}_3)l_3 - \hat{f}(\boldsymbol{\omega}_{3,\beta}, \boldsymbol{U}_{2,\beta}^{int}, \boldsymbol{U}_{3,\beta}^{int}, \boldsymbol{n}_3)l_3\right].
\end{aligned}$$

$\mathbf{H}_{2,\beta}$ is a formal two-dimensional first order positivity-preserving scheme, namely, the same type as (2.14). $\mathbf{H}_{1,\beta}$ and $\mathbf{H}_{3,\beta}$ are formal one-dimensional first order positivity-preserving schemes, such as the scheme (2.11). Note that $\bar{\boldsymbol{U}}_h^{n+1}$ is a convex combination of $\boldsymbol{U}_\gamma^{int}$ and $\mathbf{H}_{i,\beta}$. We have therefore the following theorem.

Theorem 2.4. *Assume that the values of the solution $\boldsymbol{U}_h$ at all points in the set S_K belong to the set G for all cell K, the scheme* (2.18) *is positivity-preserving under the time step restriction*

$$\triangle t^n \max_{K^n}\left\{\frac{\sum_{i=1}^{3}\max_\beta\left\{|\boldsymbol{\omega}_{i,\beta}\cdot\boldsymbol{n}_i|\right\}l_i}{|K^n|}\right\} \leqslant \frac{2}{3}\hat{c}_1,$$

where $\hat{c}_1$ is the quadrature weight of Gauss-Lobatto rule for the first quadrature point.

Remark. Theorem 2.3 and Theorem 2.4 theoretically prove that one- and two-dimensional high order remapping algorithms are positivity-preserving under the suitable time step constraints, respectively. We adopt another more efficient approach in our numerical experiments by relaxing the time step $\triangle t^n$. We implement the standard CFL condition (2.10) and check whether all new cell averages belong to set G at each time step first. If yes, we continue the computation. Otherwise, we need to return to the previous time step and march time with $\triangle t^n/2$. This procedure is repeated if needed. Theorem 2.3 and Theorem 2.4 ensure that we return only a finite number of times to have the cell averages belonging to the set G.

2.5.3 The positivity-preserving limiter

For this part, we mainly refer to the content in [41]. The DG technique does not guarantee that the numerical solution values at the quadrature points belong to the set G. The linear scaling positivity-preserving limiter can enforce it. The purpose of this limiter is to compress the polynomial $\boldsymbol{U}_j(x)$ toward its non-negative cell average $\bar{\boldsymbol{U}}_j$, as shown in Figure 1. Using the one-dimensional situation as an illustration, the specific implementation can be described as follows.

The first step is to enforce the positivity of density. We modify the polynomial $\rho_j(x)$ by

$$\hat{\rho}_j(x) = \theta_1(\rho_j(x) - \bar{\rho}_j) + \bar{\rho}_j,$$
$$\theta_1 = \min\{1, |\frac{\bar{\rho}_j - \varepsilon_2}{\bar{\rho}_j - b}|\},\ b = \min_{x\in S_j} \rho_j(x),$$

where S_j is a set of Gauss-Lobatto quadrature points in the cell K_j, ε_2 is a very small positive constant which satisfies $\bar{\rho}_j \geqslant \varepsilon_2$ for all j. For example, we take $\varepsilon_2 = 10^{-14}$ in our code.

The second step is to enforce the positivity of the internal energy e. Define $\hat{\boldsymbol{U}}_j(x) = (\hat{\rho}_j(x), \boldsymbol{m}_j(x), E_j(x))^T$ after the first step. For each point in S_j, if $e(\hat{\boldsymbol{U}}_j(x)) \geqslant 0$ set $\theta_2 = 1$; otherwise, set

$$\theta_2 = \min_{(x)\in S_j} \frac{e(\bar{\boldsymbol{U}}_j)}{e(\bar{\boldsymbol{U}}_j) - e(\hat{\boldsymbol{U}}_j(x))}.$$

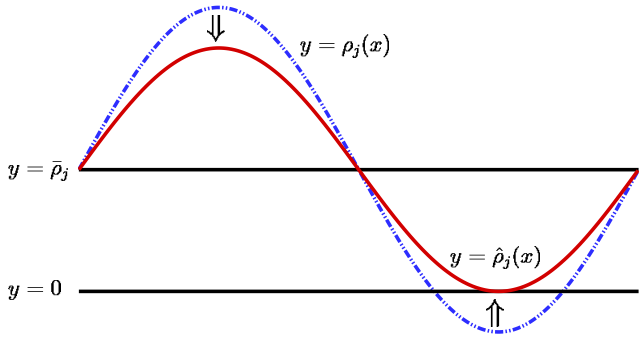


Figure 1: The schematic diagram of the positivity-preserving limiter.

Then we get the limited polynomial

$$\tilde{\boldsymbol{U}}_j(x) = \theta_2(\hat{\boldsymbol{U}}_j(x) - \bar{\boldsymbol{U}}_j) + \bar{\boldsymbol{U}}_j.$$

Such a new polynomial can satisfy $\tilde{\boldsymbol{U}}_j(x) \in G$ for all $x \in S_j$ and keep the cell average $\bar{\boldsymbol{U}}_j$ at the same time. This limiter can be used after the multi-resolution WENO limiter because it does not lose the essentially non-oscillatory property by compressing the polynomial toward the cell average. Furthermore, this limiter will not destroy conservation and accuracy, as shown in detail in [41].

To conclude this part, we emphasize that the major procedures of the high order positivity-preserving DG remapping method are completed. In the following section, we will test some numerical examples to validate the accuracy, non-oscillatory, and positivity-preserving properties of our remapping algorithm. Finally, we will implement our remapping technique in an indirect ALE approach and demonstrate its performance on the simulation of the Euler system in one and two dimensions.

3 Numerical results for the remapping algorithm

In this section, we test the performance of our remapping algorithm in the two-dimensional case for properties such as high order accuracy, positivity-preserving and essentially non-

oscillation. We will not describe the one-dimensional case here since the results are similar. We mimic the distorted mesh by smoothly moving meshes and randomly moving meshes, which are generated by perturbing the internal vertices. For simplicity, we set the computational area as a rectangle $[x_l, x_r] \times [y_l, y_r]$. Suppose (x_p^m, y_p^m) is the coordinate of an interior node and the superscript m represents the times of remapping. (x_p^0, y_p^0) is the coordinate of the corresponding node of the initial mesh. The two different ways of mesh movements are as follows.

- The smoothly moving mesh [31]

$$(x_p^m, y_p^m) = (x_p^0, y_p^0) + C_s \frac{m}{M_r} \sin\left(\frac{2\pi x_p^0}{x_r - x_l}\right) \sin\left(\frac{2\pi y_p^0}{y_r - y_l}\right),$$

  where $C_s = 0.1$. M_r is the total number of remapping.

- The randomly moving mesh

$$(x_p^m, y_p^m) = (x_p^0, y_p^0) + C_r h(r_x^m, r_y^m),$$

  where $C_r = 0.6$ and h is the smallest element diameter. $r_x^m, r_y^m \in [-0.5, 0.5]$ are random numbers.

Our remapping algorithm is validated using a suite of the cyclic remapping process [30], which starts from the initial mesh and returns to it after remapping on the moving mesh ten times.

3.1 Accuracy test

We tested four types of the remapping algorithms to verify the convergence: one without any limiter, one with the multi-resolution WENO limiter, one with the positive-preserving limiter, and one with the multi-resolution WENO limiter plus the positivity-preserving limiter. The numerical results of the four remapping algorithms are respectively denoted as u_h, u_h^W, u_h^P, and $u_h^{W,P}$. The following smooth function with periodic boundary condition is

selected for the accuracy test,

$$u(x,y) = \cos^8(2\pi x)\cos^8(2\pi y) + 10^{-15}, \quad 0 \leqslant x, y \leqslant 1.$$

The numerical calculation is performed on a set of uniform refined triangular meshes obtained by discretizing the computational domain into $10\times 10, 20\times 20, 40\times 40, 80\times 80$ and 160×160 uniform quadrilateral meshes, respectively, and then splitting each quadrilateral into two equal triangles. We compute the error measured in the L^1, L^2, and L^∞ norms for numerical solution u_h, i.e.,

$$\|\epsilon\|_{L^1} = \frac{1}{\sum\limits_{K\in\mathcal{T}_h} S_K} \sum_{K\in\mathcal{T}_h} \left(\frac{1}{N_p} \sum_{s=1}^{N_p} |u(x_K^s, y_K^s) - u_h(x_K^s, y_K^s)| S_K \right),$$

$$\|\epsilon\|_{L^2} = \sqrt{\frac{1}{\sum\limits_{K\in\mathcal{T}_h} S_K} \sum_{K\in\mathcal{T}_h} \left(\frac{1}{N_p} \sum_{s=1}^{N_p} \left(u(x_K^s, y_K^s) - u_h(x_K^s, y_K^s)\right)^2 S_K \right)},$$

$$\|\epsilon\|_{L^\infty} = \max_{K\in\mathcal{T}_h} \max_s |u(x_K^s, y_K^s) - u_h(x_K^s, y_K^s)|.$$

where N_p is the number of quadrature points and the quadrature points $\{(x_K^s, y_K^s)\}$ on the triangle K are obtained by the affine transformation (2.8) from the reference triangle $\mathcal{K}$ to K. In practice, we take $N_p = 21$ and the corresponding points on $\mathcal{K}$ are $\left\{ \left(\frac{i}{5}, \frac{j}{5}\right), j = 0, \cdots, 5, i = 0, \cdots, 5-j \right\}$.

In Table 3.1, we summarize the remapping errors on the smoothly moving mesh and numerical rates of convergence. We also compute the initial L^2 projection $u_h^E(x,y)$ of $u(x,y)$ on the initial mesh and calculate the error $||u_h^E - u||$. In addition, we show the proportion of the cells being modified by the positivity-preserving limiter, denoted as 'N_{pp}'. The last column 'M' indicates the total number of remapping steps. Table 3.2 shows the corresponding results on the randomly moving mesh. From Table 3.1 and Table 3.2, we can clearly see that our high order DG remapping method achieves the designed order of accuracy in the L^1, L^2 and L^∞ norms respectively, no matter whether the limiters are applied or not. Moreover, when using the WENO limiter in the remapping algorithm for accuracy testing, we artificially mark each triangular cell as a troubled cell instead of using the detection technique, in order to fully observe accuracy after the limiter is applied everywhere. Obviously, the

new multi-resolution WENO limiter maintains the designed order of accuracy. Furthermore, compared to the high order remapping algorithm without the positivity-preserving limiter, the number of remapping steps in the high order positivity-preserving remapping algorithm is mostly the same or only increases slightly.

Additionally, we provide the flipping mesh setup shown below to confirm that our remapping method is not constrained by mesh movement, i.e., it can remap between the old and new meshes when the mesh nodes move beyond the size of their neighboring cells. We give the initial mesh configuration by discretizing the computational region into non-uniform rectangular cells and then splitting each quadrilateral into two equal triangles. The mesh size of the rectangular cells are

$$h_x^1 < h_x^2 < \cdots < h_x^{N_x},\ h_x^{N_x} = 2h_x^1,$$

where the mesh size $h_x^i = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}$ and the y direction is divided equally. We design the flipping mesh in the same way and its rectangular mesh satisfies

$$\tilde{h}_x^1 = h_x^{N_x}, \cdots, \tilde{h}_x^{N_x} = h_x^1.$$

We remap from the initial mesh to the flipping mesh and return to the initial mesh for 10 times. Table 3.3 shows the results of these three types of remapping algorithms on the flipping mesh. All of them have the expected third-order accuracy, which indicates that our remapping method can also have good accuracy when dealing with large mesh movements.

3.2 Non-oscillatory and positivity-preserving tests

In this subsection, we test the following discontinuous function on the randomly moving meshes to verify the performance of our remapping algorithm, in particular, the essentially non-oscillatory and positivity-preserving properties. We choose a cubical pyramid function

Table 3.1: Errors and orders of accuracy of the remapping algorithms on the smoothly moving meshes. The mesh is composed of $2 \times N_x \times N_y$ triangular cells.

$\|\|u_h^E - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	5.688E-03		1.602E-02		1.706E-01		-	-
$2 \times 20 \times 20$	7.208E-04	2.98	2.030E-03	2.98	2.425E-02	2.81	-	-
$2 \times 40 \times 40$	9.253E-05	2.96	2.651E-04	2.94	3.434E-03	2.82	-	-
$2 \times 80 \times 80$	1.165E-05	2.99	3.351E-05	2.98	4.407E-04	2.96	-	-
$2 \times 160 \times 160$	1.459E-06	3.00	4.200E-06	3.00	5.538E-05	2.99	-	-
$\|\|u_h - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	6.308E-03		1.710E-02		1.994E-01		-	37
$2 \times 20 \times 20$	8.823E-04	2.84	2.201E-03	2.96	2.929E-02	2.77	-	70
$2 \times 40 \times 40$	1.197E-04	2.88	3.013E-04	2.87	4.142E-03	2.82	-	130
$2 \times 80 \times 80$	1.517E-05	2.98	4.062E-05	2.89	5.172E-04	3.00	-	256
$2 \times 160 \times 160$	1.856E-06	3.03	5.159E-06	2.98	8.654E-05	2.58	-	505
$\|\|u_h^W - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	6.650E-03		1.735E-02		1.843E-01		-	37
$2 \times 20 \times 20$	9.173E-04	2.86	2.315E-03	2.91	3.288E-02	2.49	-	70
$2 \times 40 \times 40$	1.214E-04	2.92	3.060E-04	2.92	4.151E-03	2.99	-	130
$2 \times 80 \times 80$	1.519E-05	3.00	4.064E-05	2.91	5.172E-04	3.00	-	256
$2 \times 160 \times 160$	1.856E-06	3.03	5.159E-06	2.98	8.654E-05	2.58	-	505
$\|\|u_h^P - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	8.556E-03		1.899E-02		1.199E-01		0.556	37
$2 \times 20 \times 20$	9.028E-04	3.24	2.213E-03	3.10	2.929E-02	2.03	0.323	70
$2 \times 40 \times 40$	1.198E-04	2.91	3.013E-04	2.88	4.142E-03	2.82	0.157	130
$2 \times 80 \times 80$	1.517E-05	2.98	4.062E-05	2.89	5.172E-04	3.00	0.064	260
$2 \times 160 \times 160$	1.856E-06	3.03	5.159E-06	2.98	8.654E-05	2.58	0.017	576
$\|\|u_h^{W,P} - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	9.715E-03		2.116E-02		1.352E-01		0.501	37
$2 \times 20 \times 20$	9.373E-04	3.37	2.326E-03	3.19	3.288E-02	2.04	0.323	70
$2 \times 40 \times 40$	1.215E-04	2.95	3.060E-04	2.93	4.151E-03	2.99	0.157	130
$2 \times 80 \times 80$	1.519E-05	3.00	4.064E-05	2.91	5.172E-04	3.00	0.064	260
$2 \times 160 \times 160$	1.856E-06	3.03	5.159E-06	2.98	8.654E-05	2.58	0.017	576

Table 3.2: Errors and orders of accuracy of the remapping algorithms on the randomly moving meshes. The mesh is composed of $2 \times N_x \times N_y$ triangular cells.

$\|\|u_h^E - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	5.688E-03		1.602E-02		1.706E-01		-	-
$2 \times 20 \times 20$	7.208E-04	2.98	2.030E-03	2.98	2.425E-02	2.81	-	-
$2 \times 40 \times 40$	9.253E-05	2.96	2.651E-04	2.94	3.434E-03	2.82	-	-
$2 \times 80 \times 80$	1.165E-05	2.99	3.351E-05	2.98	4.407E-04	2.96	-	-
$2 \times 160 \times 160$	1.459E-06	3.00	4.200E-06	3.00	5.538E-05	2.99	-	-
$\|\|u_h - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	6.620E-03		1.623E-02		1.918E-01		-	35
$2 \times 20 \times 20$	8.485E-04	2.96	2.043E-03	2.99	3.184E-02	2.59	-	41
$2 \times 40 \times 40$	1.066E-04	2.99	2.561E-04	3.00	4.398E-03	2.86	-	44
$2 \times 80 \times 80$	1.286E-05	3.05	3.140E-05	3.03	5.573E-04	2.98	-	85
$2 \times 160 \times 160$	1.559E-06	3.04	3.840E-06	3.03	7.334E-05	2.93	-	176
$\|\|u_h^W - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	6.897E-03		1.642E-02		1.830E-01		-	35
$2 \times 20 \times 20$	8.623E-04	3.00	2.110E-03	2.96	3.376E-02	2.44	-	41
$2 \times 40 \times 40$	1.071E-04	3.01	2.579E-04	3.03	4.404E-03	2.94	-	44
$2 \times 80 \times 80$	1.287E-05	3.06	3.142E-05	3.04	5.573E-04	2.98	-	85
$2 \times 160 \times 160$	1.559E-06	3.04	3.840E-06	3.03	7.334E-05	2.93	-	176
$\|\|u_h^P - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	8.206E-03		1.751E-02		1.176E-01		0.556	35
$2 \times 20 \times 20$	8.683E-04	3.24	2.066E-03	3.08	3.184E-02	1.89	0.463	41
$2 \times 40 \times 40$	1.062E-04	3.03	2.561E-04	3.01	4.398E-03	2.86	0.300	44
$2 \times 80 \times 80$	1.286E-05	3.05	3.140E-05	3.03	5.573E-04	2.98	0.142	85
$2 \times 160 \times 160$	1.559E-06	3.04	3.840E-06	3.03	7.334E-05	2.93	0.046	176
$\|\|u_h^{W,P} - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	8.457E-03		1.834E-02		1.626E-01		0.524	35
$2 \times 20 \times 20$	8.834E-04	3.26	2.132E-03	3.10	3.375E-02	2.27	0.463	41
$2 \times 40 \times 40$	1.067E-04	3.05	2.579E-04	3.05	4.404E-03	2.94	0.300	44
$2 \times 80 \times 80$	1.287E-05	3.05	3.142E-05	3.04	5.573E-04	2.98	0.142	85
$2 \times 160 \times 160$	1.559E-06	3.04	3.840E-06	3.03	7.334E-05	2.93	0.046	176

Table 3.3: Errors and orders of accuracy of the remapping algorithms on the flipping moving meshes. The mesh is composed of $2 \times N_x \times N_y$ triangular cells.

$\|\|u_h^E - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	6.093E-03		1.794E-02		2.554E-01		-	-
$2 \times 20 \times 20$	7.998E-04	2.93	2.408E-03	2.90	4.811E-02	2.41	-	-
$2 \times 40 \times 40$	1.027E-04	2.96	3.108E-04	2.95	6.535E-03	2.88	-	-
$2 \times 80 \times 80$	1.291E-05	2.99	3.923E-05	2.99	8.472E-04	2.95	-	-
$2 \times 160 \times 160$	1.615E-06	3.00	4.911E-06	3.00	1.052E-04	3.01	-	-
$\|\|u_h - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	1.300E-02		2.726E-02		2.315E-01		-	220
$2 \times 20 \times 20$	2.375E-03	2.45	5.424E-03	2.33	6.092E-02	1.93	-	410
$2 \times 40 \times 40$	3.735E-04	2.67	9.227E-04	2.56	1.062E-02	2.52	-	790
$2 \times 80 \times 80$	5.626E-05	2.73	1.505E-04	2.62	1.821E-03	2.54	-	1550
$2 \times 160 \times 160$	8.011E-06	2.81	2.250E-05	2.74	2.735E-04	2.73	-	3070
$\|\|u_h^W - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	1.676E-02		3.407E-02		2.172E-01		-	220
$2 \times 20 \times 20$	2.431E-03	2.78	5.769E-03	2.56	6.713E-02	1.69	-	410
$2 \times 40 \times 40$	3.790E-04	2.68	9.414E-04	2.62	1.108E-02	2.60	-	790
$2 \times 80 \times 80$	5.636E-05	2.75	1.508E-04	2.64	1.831E-03	2.60	-	1550
$2 \times 160 \times 160$	8.012E-06	2.81	2.251E-05	2.74	2.740E-04	2.74	-	3070
$\|\|u_h^P - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	1.748E-02		3.776E-02		2.554E-01		0.202	220
$2 \times 20 \times 20$	2.266E-03	2.95	5.352E-03	2.82	6.112E-02	2.06	0.174	410
$2 \times 40 \times 40$	3.572E-04	2.67	9.166E-04	2.55	1.062E-02	2.52	0.145	790
$2 \times 80 \times 80$	5.542E-05	2.69	1.503E-04	2.61	1.821E-03	2.54	0.089	1550
$2 \times 160 \times 160$	7.984E-06	2.80	2.250E-05	2.74	2.736E-04	2.73	0.045	3106
$\|\|u_h^{W,P} - u\|\|$								
Mesh size	L^1	order	L^2	order	L^∞	order	N_{pp}	M
$2 \times 10 \times 10$	2.070E-02		4.353E-02		2.615E-01		0.191	220
$2 \times 20 \times 20$	2.357E-03	3.13	5.735E-03	2.92	6.732E-02	1.96	0.174	410
$2 \times 40 \times 40$	3.627E-04	2.70	9.355E-04	2.62	1.108E-02	2.60	0.145	790
$2 \times 80 \times 80$	5.552E-05	2.71	1.506E-04	2.64	1.831E-03	2.60	0.089	1550
$2 \times 160 \times 160$	7.985E-06	2.80	2.251E-05	2.74	2.740E-04	2.74	0.045	3106

[20] as the test function

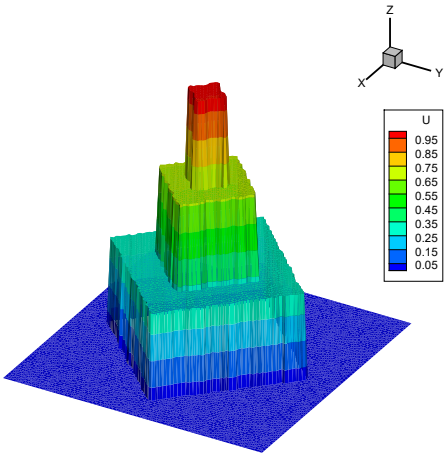
$$u(x,y)=\begin{cases}1 & R\leqslant 0.08\\ 0.66 & 0.08<R\leqslant 0.2\\ 0.33 & 0.2<R\leqslant 0.38\\ 10^{-15} & R>0.38\end{cases},\quad R=|x-0.5|+|y-0.5|,\ 0\leqslant x,y\leqslant 1\ .$$

We conduct the test on unstructured triangular meshes with 14120 cells. Four remapping procedures are performed depending on whether or not the limiters are employed: without any limiter, with the multi-resolution WENO limiter, with the positivity-preserving limiter, and with two limiters. The remapping results of these algorithms are shown in Figure 2 and the values at 64 points at the cut line $y = 0.5$ and $x = y$ are shown in Figure 3 and Figure 4.

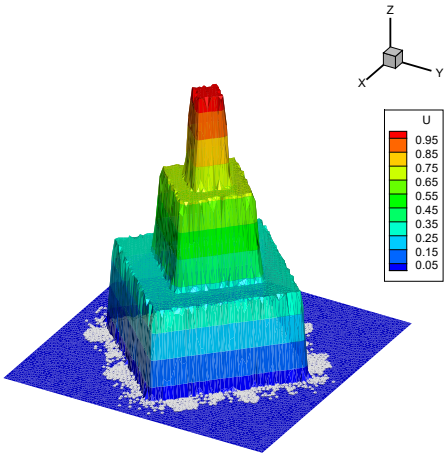
Compared with the remapping results without any limiter, the remapping results with the multi-resolution WENO limiter can maintain the essentially non-oscillatory property near the discontinuous regions. Besides, by applying the WENO limiter to the remapping algorithm, it is impossible to eliminate the number of cells that generate negative cell averages (marked in white in Figure 2). Thus a positive-preserving limiter is necessary to compute numerical examples with discontinuous solutions. As one can see the remapping results produced by the remapping algorithm with the positive-preserving limiter can ensure that the solutions remain positive. During the remapping process of the remapping algorithm with two limiters, there are about 24.3% of the cells which have been modified by the multi-resolution WENO limiter and about 9.4% of the cells which have been modified by the positivity-preserving limiter. Overall, our remapping results with these two limiters can handle the overshoots near the discontinuities and preserve positivity well.

4 Numerical tests for the ALE simulation

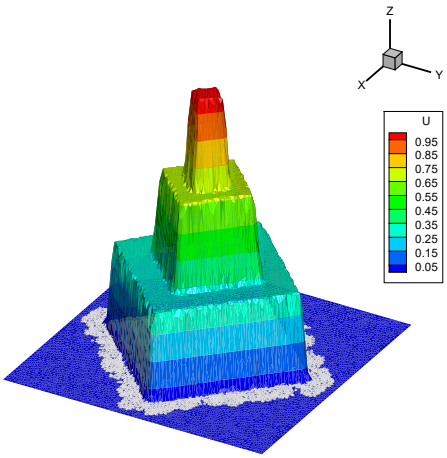
This section aims to verify the good performance of our remapping algorithm applied in the indirect ALE method for the compressible Euler equations in one and two space dimensions. We refer to the indirect ALE method in [23], which is constructed based on the ALE-DG



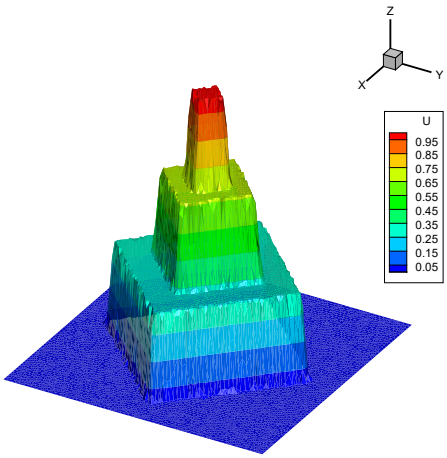
(a) The initial function



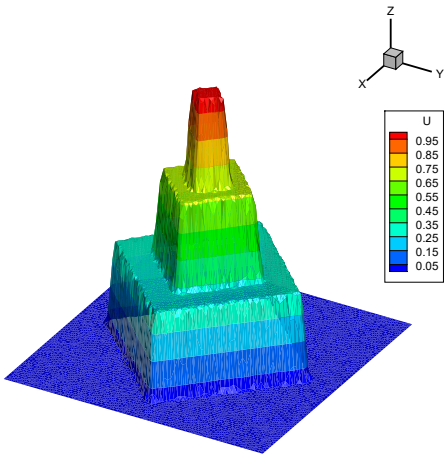
(b) Remapping results without limiter



(c) Remapping results with WENO limiter



(d) Remapping results with PP limiter



(e) Remapping results with two limiters

Figure 2: The non-oscillatory and positivity-preserving tests for the discontinuous profiles by remapping method. The white symbols in the subgraphs (b) and (c) represent the cells with negative cell averages.

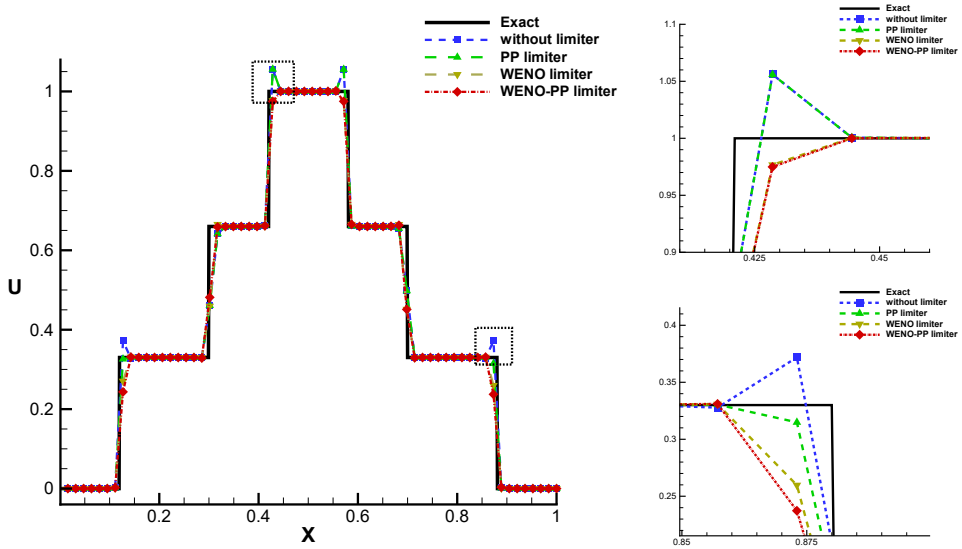


Figure 3: The remapping results at the cut line $y = 0.5$. Top right: the zoomed-in subfigure at $x \in [0.41, 0.46]$. Bottom right: the zoomed-in subfigure at $x \in [0.85, 0.89]$.

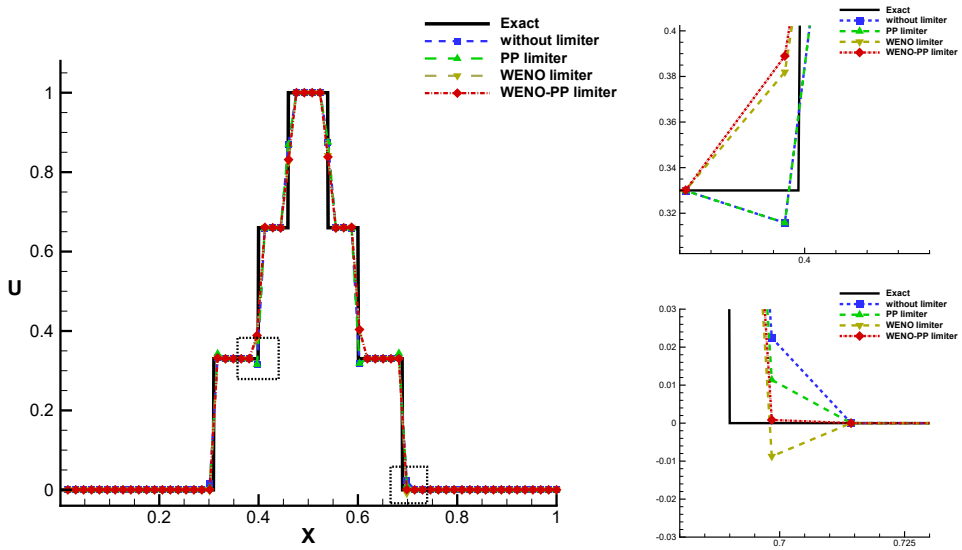


Figure 4: The remapping results at the cut line $x = y$. Top right: the zoomed-in subfigure at $x \in [0.38, 0.42]$. Bottom right: the zoomed-in subfigure at $x \in [0.68, 0.73]$.

method developed in [17,18]. We will use this Lagrangian type DG scheme to solve the Euler equations with periodic rezone and remapping steps to avoid the failure of the simulation led by computational mesh undergoing distortion or large deformation. For each numerical test problem we will use different rezone strategies. In order to obtain better numerical results, both the multi-resolution WENO limiter and the positive-preserving limiter will be used in the remapping algorithm. The remapping of vector functions can be performed component by component, except for the positivity-preserving procedure of density and internal energy, which would require us to consider the vector of conserved variables as one entity. To achieve this, we first utilize the positivity-preserving limiter for the density ρ, then we limit the momentum, total energy and the modified density to ensure positivity of the internal energy e.

4.1 Numerical tests for the one-dimensional Euler equation

In this subsection, we consider the compressible Euler equation of gas dynamics in one dimension:

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix}+\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2+p\\ u(E+p)\end{pmatrix}=0$$

where ρ is the density, u is the fluid velocity, E is the total energy, $p=(\gamma-1)(E-\frac{1}{2}\rho u^2)$ is the pressure and γ is a constant that depends on the particular gas under consideration.

Example 4.1. We first consider the accuracy test in one dimension [12]. The initial condition is given as

$$\rho(x,0)=\frac{1+0.2\sin(x)}{2\sqrt{3}},\quad u(x,0)=\sqrt{\gamma}\rho(x,0),\quad p(x,0)=\rho(x,0)^{\gamma}.$$

The computational domain is set to be $[0,2\pi]$ with periodic boundary conditions. We follow the setting in the reference [17] that when periodic boundary conditions are applied in the ALE framework, the first point $x_{\frac{1}{2}}$ and last point $x_{N+\frac{1}{2}}$ move at the same speed. The values at the boundary points are obtained from the periodicity of the solution, i.e., $u^-_{h,\frac{1}{2}}=u^-_{h,N+\frac{1}{2}}$

and $u^+_{h,N+\frac{1}{2}} = u^+_{h,\frac{1}{2}}$. Similar strategy is adopted in two dimensions. We take $\gamma = 3$, which allows us to verify that $2\sqrt{3}\rho(x,t)$ is the exact solution of the Burgers equation:

$$\mu_t + \left(\frac{\mu^2}{2}\right)_x = 0, \quad \mu(x,0) = 1 + 0.2\sin(x),$$

and the exact solutions u and p satisfy:

$$u(x,t) = \sqrt{\gamma}\rho(x,t), \quad p(x,t) = \rho(x,t)^\gamma.$$

We present the numerical results for density at $t = 0.3$ obtained by the indirect ALE approach with our remapping algorithm. After every 10 Lagrangian steps, we rezone the old mesh to the new uniform mesh and apply our remapping procedure. Table 4.1 shows the error on the different sizes of the mesh $32, 64, 128, 256, 512$ and it can be shown that the ALE scheme have achieved the designed third-order accuracy.

Table 4.1: Numerical convergence results of the indirect ALE-DG scheme for one-dimensional Euler equation at $t = 0.3$.

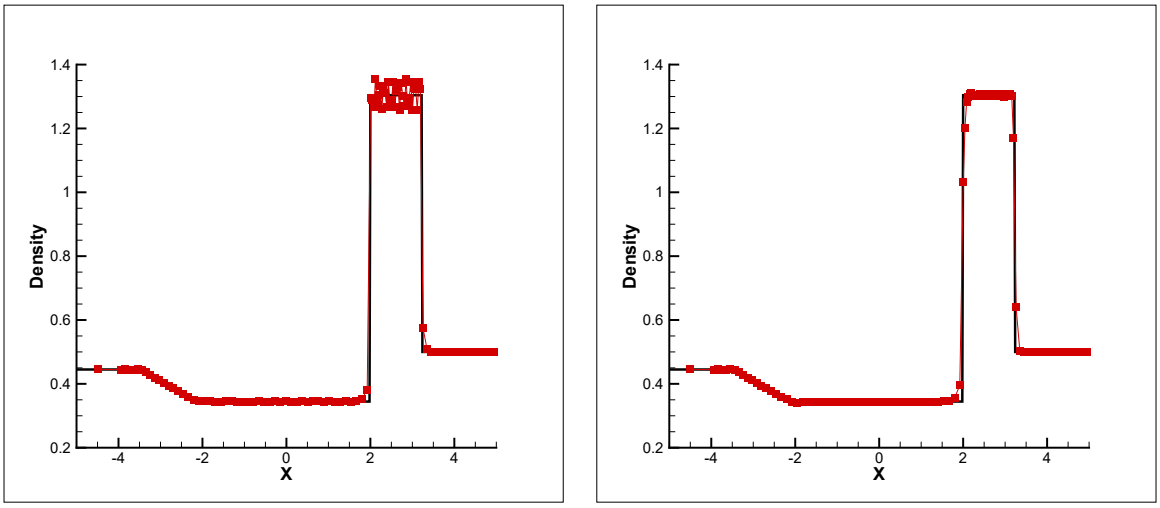
Mesh	L^1	order	L^2	order	L^∞	order
32	2.612E-06		4.009E-06		1.593E-05	
64	3.248E-07	3.01	4.910E-07	3.03	1.892E-06	3.07
128	4.046E-08	3.00	6.052E-08	3.02	2.155E-07	3.13
256	5.251E-09	2.95	7.776E-09	2.96	2.409E-08	3.16
512	7.656E-10	2.78	1.121E-09	2.79	3.649E-09	2.72

Example 4.2. Next we consider the Lax problem of the Euler system with the initial condition

$$(\rho, u, p) = \begin{cases} (0.445, 0.698, 3.528)\,, & x < 0 \\ (0.5, 0, 0.571)\,, & x > 0 \end{cases} \qquad \gamma = 1.4.$$

The computational domain is $[-5, 5]$ and the number of cells is $N = 100$. We use the indirect ALE-DG method to compute the solution until $t = 1.3$ and perform the remapping and the rezoning step every 20 time steps when $t > 1.0$. To demonstrate the applicability of the multi-resolution WENO limiter in the simulation of compressible fluid, we calculate the Lax problem using the indirect ALE methods with and without this limiter in remapping step.

In Figure 5 we see that the ALE-DG method equipped with the multi-resolution WENO limiter not only maintains the advantages of the Lagrangian type scheme, that is, there very few transition points along the contact discontinuity, but also handles the overshoots very well.



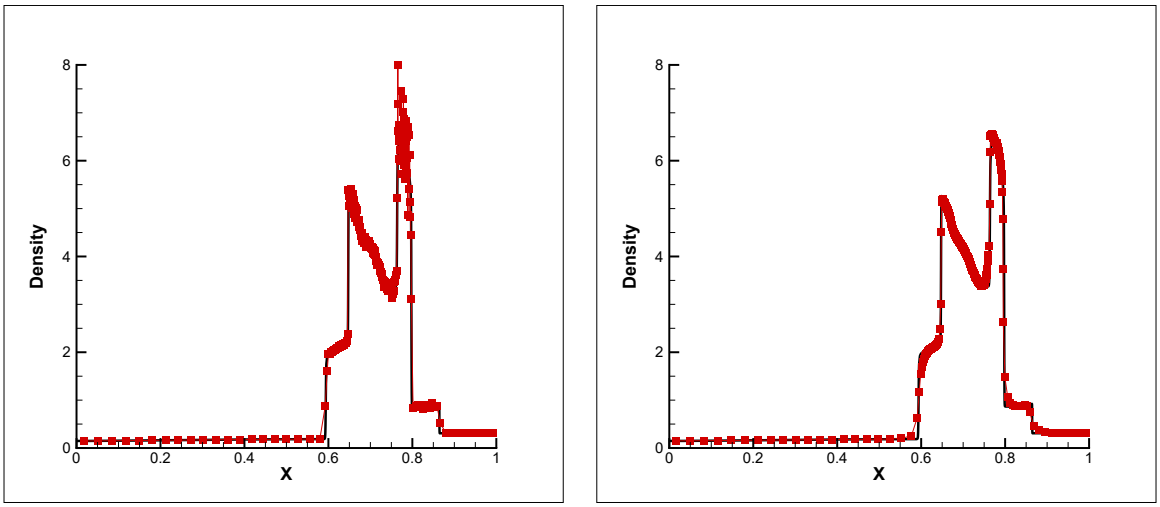
(a) ALE-DG (b) ALE-DG method with WENO limiter

Figure 5: Comparison of solutions for the Lax problem calculating by the ALE-DG method with (right) and without (left) the multi-resolution WENO limiter. The black solid line is the reference solution. The red points represent the ALE-DG solution.

Example 4.3. We now consider the interaction of two blast waves, with the initial condition

$$(\rho, u, p) = \begin{cases} (1, 0, 1000)\,, & x \in [0, 0.1) \\ (1, 0, 0.01)\,, & x \in [0.1, 0.9) \\ (1, 0, 100)\,, & x \in [0.9, 1] \end{cases} \quad \gamma = 1.4.$$

We apply our remapping procedure and the rezoning method every 50 time steps after $t > 0.03$. The numerical results of the density for two ALE-DG schemes at time $t = 0.038$ with $N = 200$ cells are displayed in Figure 6. Compared to the reference solution and the ALE-DG solution without limiter, the indirect ALE method with the multi-resolution WENO limiter handles the overshoots effectively.



(a) The ALE-DG method (b) The ALE-DG method with WENO limiter

Figure 6: Comparison of solutions for the blast wave problem calculating by the ALE-DG method with (right) and without (left) the multi-resolution WENO limiter. The black solid line is the reference solution. The red points represent the ALE-DG solution.

4.2 Numerical tests for the two-dimensional Euler equations

Consider the two-dimensional Euler system

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E\end{pmatrix}+\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(E+p)\end{pmatrix}+\frac{\partial}{\partial y}\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(E+p)\end{pmatrix}=0$$

where ρ, u and v denote the density, x-direction velocity and y-direction velocity, respectively. E is the total energy, $p=(\gamma-1)(E-\frac{1}{2}\rho(u^2+v^2))$ is the pressure and γ is a constant that depends on the particular gas under consideration.

Example 4.4. First we conduct an accuracy test for the 2D Euler equations [12] on $[0,4\pi]\times[0,4\pi]$. The initial condition is:

$$\rho(x,y,0)=\frac{1+0.2\sin(\frac{x+y}{2})}{2\sqrt{3}},\quad u(x,y,0)=v(x,y,0)=\sqrt{\gamma}\rho(x,y),\quad p(x,y,0)=\rho(x,y,0)^{\gamma}.$$

We set $\gamma=3$, and it could be easily verified that $\sqrt{6}\rho(x,y,t)$ is the exact solution of the

2D Burgers equation:

$$\mu_t + \left(\frac{\mu^2}{2}\right)_x + \left(\frac{\mu^2}{2}\right)_y = 0, \quad \mu(x, y, 0) = 1 + 0.2\sin(\frac{x+y}{2}),$$

and

$$u(x, y, t) = v(x, y, t) = \sqrt{\frac{\gamma}{2}}\rho(x, y, t), \quad p(x, y, t) = \rho(x, y, t)^\gamma.$$

To assess the convergence order of the indirect ALE-DG scheme, the numerical calculation is performed on a set of initially uniform refined triangular meshes. The initial computational domain is discretized into $10 \times 10, 20 \times 20, 40 \times 40, 60 \times 60$ and 80×80 uniform quadrilateral meshes, respectively. And then each quadrilateral is split into two triangles. The final computational time is 0.3. We perform the accuracy test in an Eulerian regime, including setting the new mesh as the initial mesh after each Lagrangian step and then applying the remapping algorithm. We record the CPU time measured in seconds on a ThinkCentre with 8GB RAM. We show the numerical results of density in Table 4.2. It can be seen that the convergence order have achieved the designed third-order accuracy.

Table 4.2: Numerical convergence results of the indirect ALE-DG scheme in the Eulerian regime for two-dimensional Euler equations at $t = 0.3$.

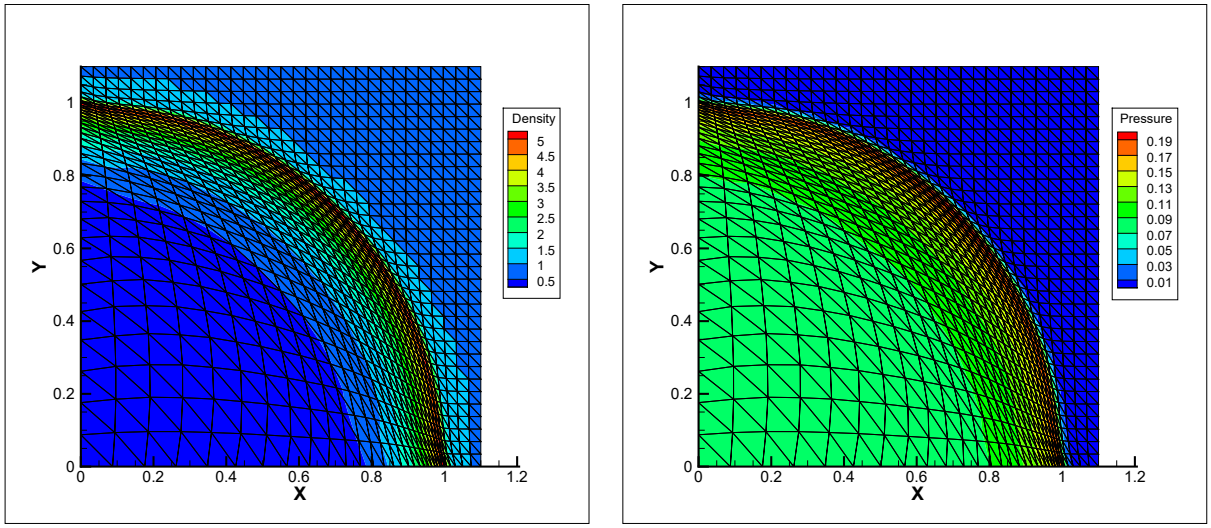
Mesh size	L^1	order	L^2	order	L^∞	order	CPU(s)
$2 \times 10 \times 10$	4.695E-05		6.060E-05		1.880E-04		1.06
$2 \times 20 \times 20$	8.187E-06	2.52	1.078E-05	2.49	3.575E-05	2.39	8.22
$2 \times 40 \times 40$	1.095E-06	2.90	1.523E-06	2.82	5.485E-06	2.70	65.00
$2 \times 60 \times 60$	3.343E-07	2.93	4.544E-07	2.98	1.617E-06	3.01	221.69
$2 \times 80 \times 80$	1.462E-07	2.87	1.990E-07	2.87	6.842E-07	2.99	537.20

Example 4.5. The Sedov problem [35] describes the evolution of a blast wave generated by an intense explosion at the origin $(0, 0)$. The computational domain is a $[0, 1.1] \times [0, 1.1]$ rectangle which is discretized into 2048 triangular cells with $\gamma = 1.4$. The initial state is

$$\rho = 1, \ u = 0, \ v = 0.$$

The internal energy of the system is 10^{-13} almost everywhere except the only cell K contained

the origin where it has a value of $e = \frac{0.244816}{|K|}$. Reflective boundary conditions are applied on the four boundaries.



(a) density (b) pressure

Figure 7: The Sedov problem of the ALE-DG scheme.

We use the indirect ALE method with our remapping algorithm to run the Sedov problem to the final time $t = 1$. The rezone step, which uses a simple smoothing strategy on the inner points, and the remapping step are implemented every 20 time steps after $t = 0.5$ in the ALE computation. The density and pressure profiles of the Sedov problem is shown in the Figure 7. As one can see, the ALE-DG scheme captures the shock precisely and the mesh quality is well after adjusting the inner mesh. The density simulation as a function of the radius is presented in Figure 8 and one can observe that our results are comparable to the reference results. The radial nature of the solution is well preserved.

Example 4.6. Next, we consider the Noh problem [32] with the initial condition as

$$\rho = 1, \quad u_r = -1, \quad e = 10^{-14},$$

where u_r is the radial velocity and take $\gamma = 5/3$. The computational domain is $[0, 1] \times [0, 1]$. For the left and bottom boundaries, reflective boundary conditions are considered, whereas for the right and top boundaries, free boundary conditions are considered. We perform the

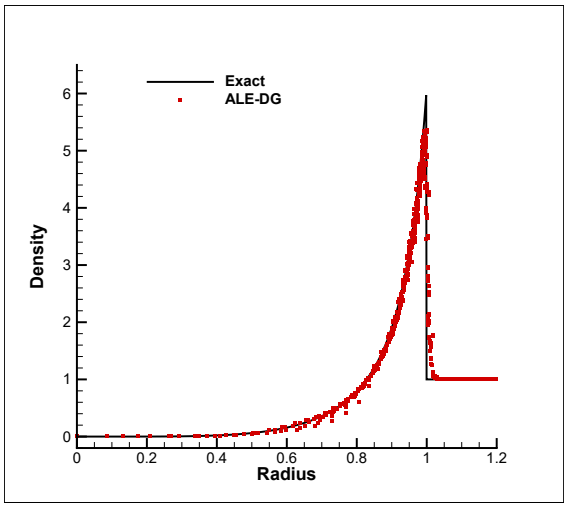


Figure 8: Scatter plot of the density value for each cell as a function of the radial position.

rezoning and the remapping procedures every 20 time steps. The ALE-DG method calculates the Noh problem in two cases, namely, whether the remapping step includes the multi-resolution WENO limiter or not. We present the density contour and scatter plot results at time $t = 0.6$ with 2048 triangular cells in Figure 9. By comparing with the ALE-DG method without WENO limiter, we can see that the density peak obtained by the new indirect ALE-DG scheme with the multi-resolution WENO limiter in the remapping algorithm is in good agreement with the exact solution. As the initial internal energy is extremely close to 0 and the numerical solutions may be negative, making the scheme unstable, the positivity-preserving limiter is required for this test. We have recorded the cells modified by the positivity-preserving limiter during the remapping and these cells have been labeled in white color in Figure 10. According to statistics, the multi-resolution WENO limiter is utilized on around 2.17% of the cells and the positivity-preserving limiter is utilized on around 14.7% of the cells. Numerical experiments indicate the good behavior of the resulting scheme in its robustness and sharp shock transition.

Example 4.7. The Saltzman test case [11] describes the motion of a piston which is impinging on a fluid at rest contained in the initial computational domain given by $[0, 1] \times$

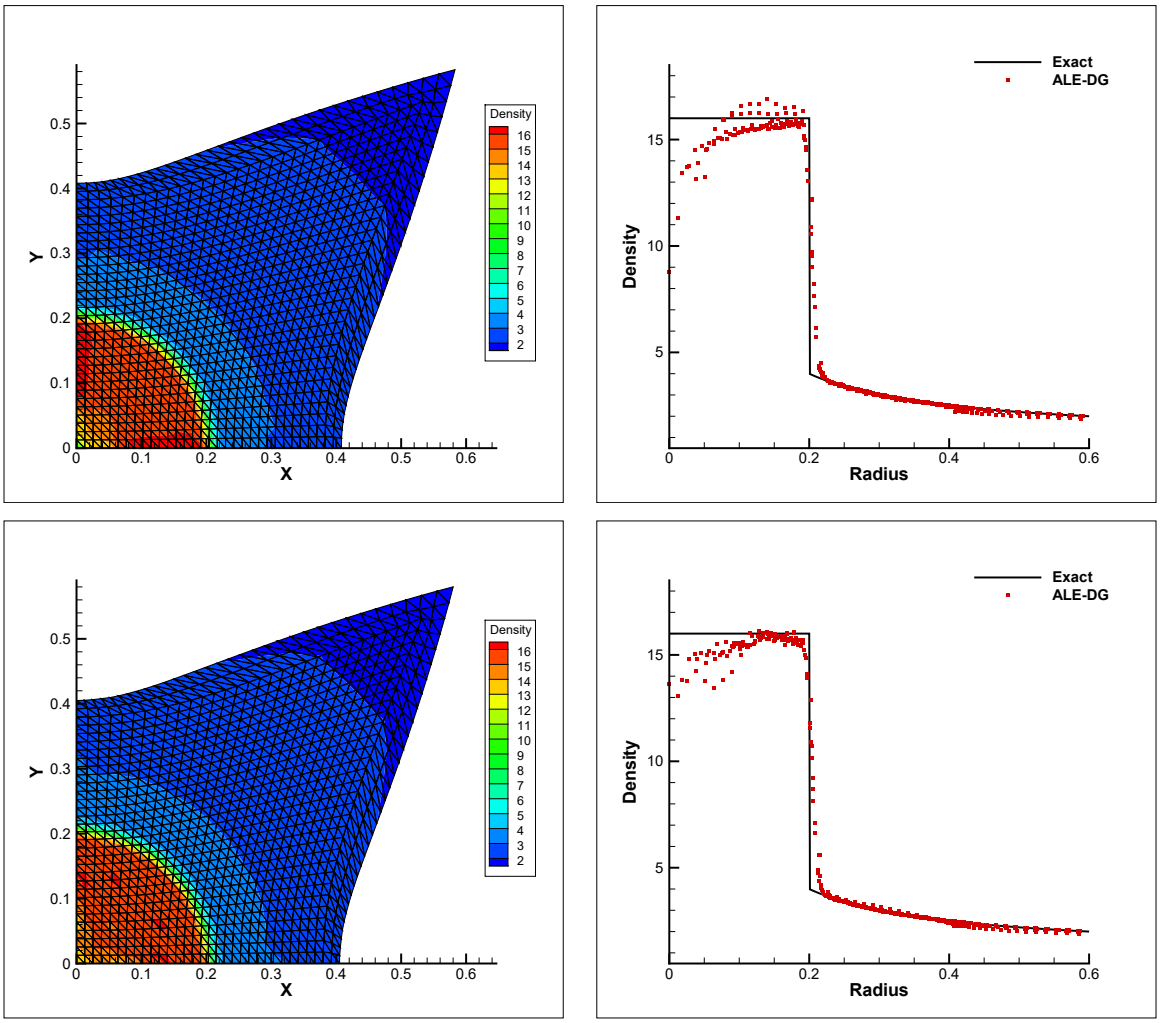


Figure 9: The Noh problem of the ALE-DG scheme. Left: density distribution with mesh configuration ; Right: scatter plot of numerical scheme compared with the exact solution; Top row: The result of the ALE-DG method without WENO limiter in remapping step; Bottom row: The result of the the ALE-DG method with WENO limiter in remapping step.

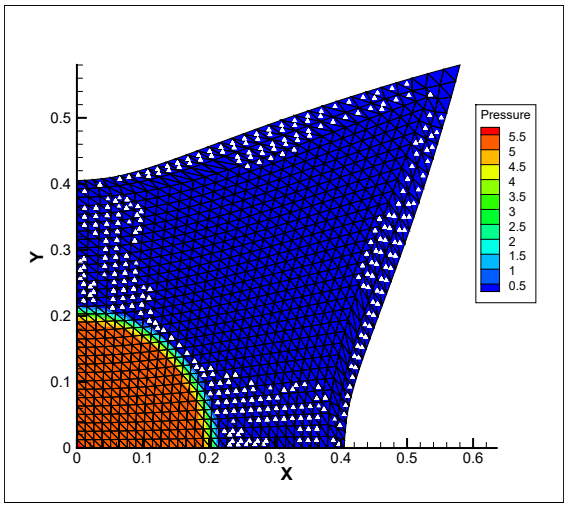


Figure 10: The pressure contour of the Noh problem. The white symbols represent the cells that have been modified by the positivity-preserving limiter during the remapping phase.

$[0, 0.1]$. The initial condition is

$$\rho = 1,\ u = 0,\ v = 0,\ e = 10^{-10},$$

and the adiabatic gas constant $\gamma = 5/3$. The left boundary is a moving piston with the fixed velocity $u = 1$. Reflective boundary conditions are considered everywhere except for the piston. The initial mesh with 640 triangular cells is shown in Figure 11. The moving piston generates a strong shock wave and the cells which lie near the piston are highly compressed during the simulation. Therefore, the indirect ALE method is the best choice to solve the Salzmann problem.

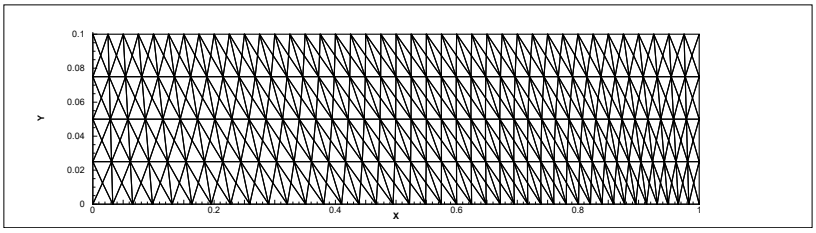
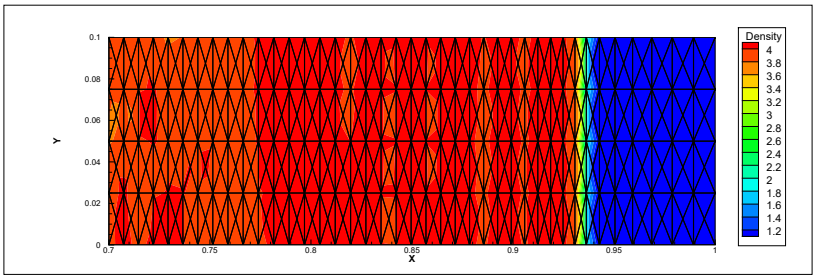
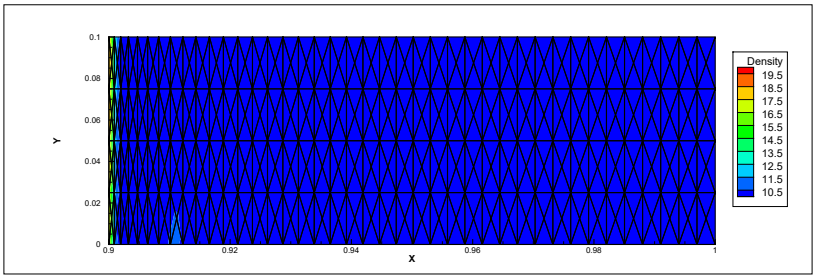


Figure 11: The initial Saltzman mesh.

In the rezone step of this problem, we preserve the y-coordinates unchanged and modify

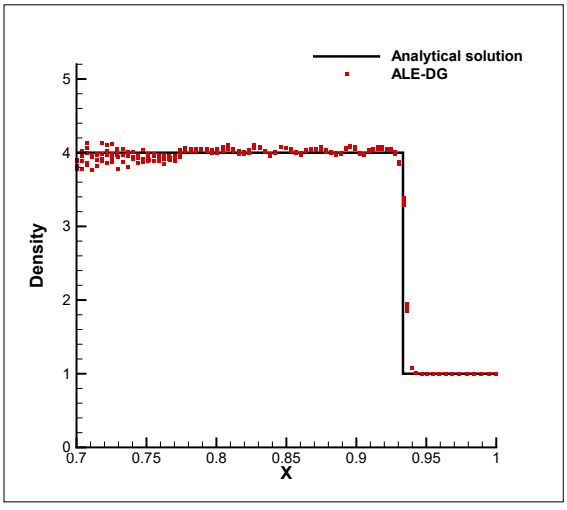


(a) $t = 0.7$

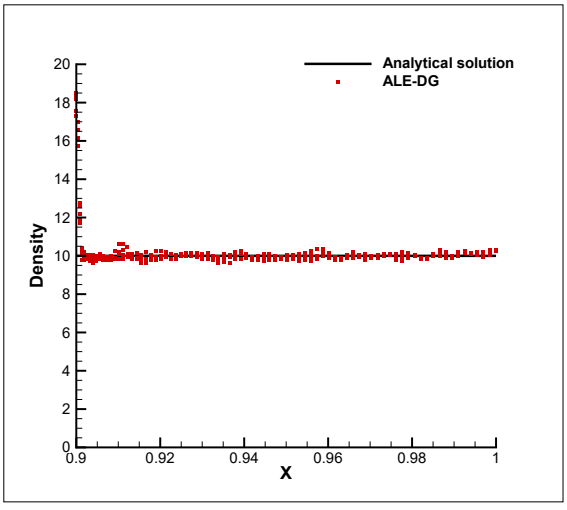


(b) $t = 0.9$

Figure 12: The density contour of the Saltzman problem of the ALE-DG scheme at output times $t = 0.7$ (top) and $t = 0.9$ (bottom).



(a) $t = 0.7$



(b) $t = 0.9$

Figure 13: Scatter plot of density values of the Saltzman problem and comparison against the analytical solution.

the inner point $P_l(x_l, y_l)$ in the x- direction as

$$\tilde{x}_l = \frac{1}{m}\sum_{s=1}^{m} x_{l,s}$$

where $x_{l,s}$, $1 \leqslant s \leqslant m$ represent the x-coordinates of the neighbors of P_l. After adjusting the computational mesh and applying the remapping procedure every 10 time steps, we calculate the Saltzman problem to the time $t = 0.7$ and $t = 0.9$ using the indirect ALE-DG scheme and display the density contours in Figure 12. The density scatter plots are shown in Figure 13, along with a comparison against the analytical solution. We can observe that the calculated results agree favorably with the analytical solution. It demonstrates the robustness of the ALE method with our remapping algorithm in handling large deformation problems.

Example 4.8. Finally, we consider the problem proposed by Dukowicz and Meltz [11] which is also a piston driven problem. The computational domain is composed of two regions with different densities but equal pressure. At the initial time, the states and regions are

$$\begin{aligned}
&\text{Left region}: \ \rho_L = 1, u_L = 0, v_L = 0, p_L = 1,\\
&\text{Coordinate of vertexes}: \ (0,0),(1,0),(1+1.5\sqrt{3},1.5),(0,1.5),\\
&\text{Right region}: \ \rho_R = 1.5, u_R = 0, v_R = 0, p_R = 1,\\
&\text{Coordinate of vertexes}: \ (1,0),(3,0),(3+1.5\sqrt{3},1.5),(1+1.5\sqrt{3},1.5).
\end{aligned}$$

In both regions $\gamma = 1.4$. The schematic diagram of Dukowicz problem is shown in Figure 14. Figure 15 shows the initial computational mesh with 4000 triangular cells. The upper and lower boundaries are reflective and the left boundary is a piston, which moves from the left with velocity 1.48.

We rezone the mesh by maintaining the y- coordinates unchanged and redistributing the interior points P_l in the x- direction such that they are evenly spaced apart. This time, the mesh rezone method and our remapping procedure are implemented every 30 time steps. We run this problem to a time of 1.3 just before the shock would leave the right region. The density contour calculated by the indirect ALE-DG scheme is displayed in Figure 16, which is in good agreement with the exact solution in [11]. It is obvious that the interface along

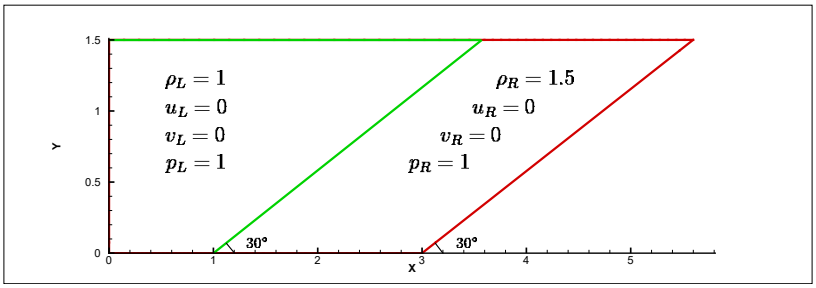


Figure 14: Sketch map for Dukowicz problem.

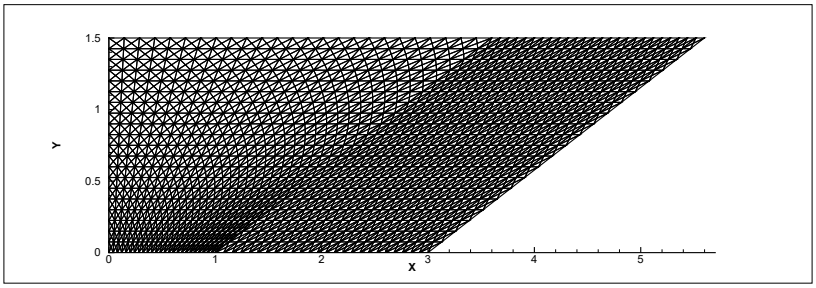


Figure 15: The initial Dukowicz mesh with 4000 triangular cells.

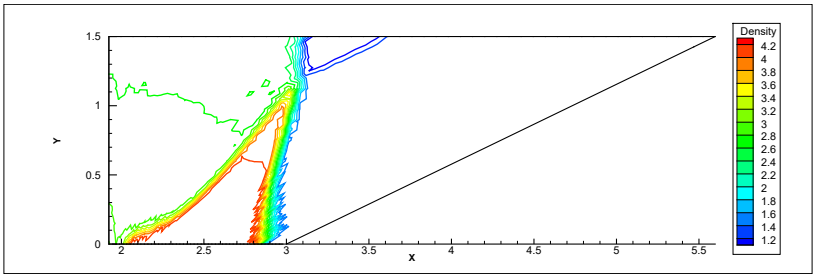


Figure 16: The Dukowicz problem of the ALE-DG scheme at $t = 1.3$.

with the incident and the transmitted shocks is sharply kept.

5 Concluding remarks

In this paper, for the simulation of the compressible fluid flow with the indirect ALE method based on DG framework, we develop a high order, essentially non-oscillatory and positivity-preserving DG remapping method in one and two dimensions. This remapping approach is based on solving the trivial equation $\frac{\partial u}{\partial t} = 0$ on a moving mesh, with the old mesh before remapping as the initial mesh at $t = 0$ and the new mesh after remapping as the final mesh at $t = T$. The relatively mild smoothness requirement (Lipschitz continuity) on the mesh movement velocity, which ensures the high order accuracy of the scheme, is used to calculate the final pseudo-time T. Our remapping algorithm does not need to calculate the intersection regions and is applicable for meshes with the same topology but otherwise arbitrary mesh deformation. Different from the other remapping algorithms, the final pseudo-time T in our remapping algorithm depends on the magnitude of the mesh movement. Therefore, our new algorithm is more efficient in dealing with remapping problems with mild grid movement. We apply the multi-resolution WENO limiter to suppress spurious oscillations near strong discontinuities and simultaneously keep the original order of accuracy in smooth regions. To further guarantee that the remapping results retain the positivity of the remapping variables, we apply the positivity-preserving limiter valid under the suitable time step constraint. Numerical examples have been presented to verify the convergence order, non-oscillation and positivity preservation of the scheme. Finally, the above good properties are also verified on the indirect ALE method with our remapping algorithm simulating 1D and 2D Euler equations. In our future work we will extend this remapping algorithm to three dimensions and design 3D higher-order conservative positivity-preserving ALE methods based on our remapping algorithm.

References

[1] R. W. Anderson, V. A. Dobrev, T. V. Kolev and R. N. Rieben, *Monotonicity in high-order curvilinear finite element arbitrary Lagrangian–Eulerian remap*, International Journal for Numerical Methods in Fluids, 77, 2015, 249–273.

[2] R. W. Anderson, V. A. Dobrev, T. V. Kolev, R. N. Rieben and V. Z. Tomov, *High-order multi-material ALE hydrodynamics*, SIAM Journal on Scientific Computing, 40, 2018, B32-B58.

[3] G. Blanchard and R. Loubere, *High order accurate conservative remapping scheme on polygonal meshes using a posteriori MOOD limiting*, Computers & Fluids, 136, 2016, 83-103.

[4] M. Castro, B. Costa and W. S. Don, *High order weighted essentially non-oscillatory WENO-Z schemes for hyperbolic conservation laws*, Journal of Computational Physics, 230, 2011, 1766-1792.

[5] J. Cheng and C.-W. Shu, *A high order accurate conservative remapping method on staggered meshes*, Applied Numerical Mathematics, 58, 2008, 1042-1060.

[6] B. Cockburn, S.-Y. Lin, and C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for scalar conservation laws III: One dimensional systems*, Journal of Computational Physics, 84, 1989, 90-113.

[7] B. Cockburn and C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for scalar conservation laws V: Multidimensional systems*, Journal of Computational Physics, 141, 1998, 199-224.

[8] Y. Di, R. Li, T. Tang, and P. Zhang, *Moving mesh finite element methods for the incompressible Navier-Stokes equations*, SIAM Journal on Scientific Computing, 26, 2005, 1036-1056.

[9] J. K. Dukowicz and J. R. Baumgardner, *Incremental remapping as a transport/advection algorithm*, Journal of Computational Physics, 160, 2000, 318-335.

[10] J. K. Dukowicz and J. W. Kodis, *Accurate conservative remapping (rezoning) for arbitrary Lagrangian-Eulerian computations*, SIAM Journal on Scientific and Statistical Computing, 8, 1987, 305-321.

[11] J. K. Dukowicz and B. Meltz, *Vorticity errors in multidimensional Lagrangian codes*, Journal of Computational Physics, 99, 1992, 115–134.

[12] G. Fu and C.-W. Shu, *A New Troubled-Cell Indicator for Discontinuous Galerkin Methods for Hyperbolic Conservation Laws*, Journal of Computational Physics, 347, 2017, 305-327.

[13] P. Fu, G. Schnucke and Y. Xia, *Arbitrary Lagrangian-Eulerian discontinuous Galerkin method for conservation laws on moving simplex meshes*, Mathematics of Computation, 88, 2019, 2221-2255.

[14] P. Fu and Y. Xia, *The positivity preserving property on the high order arbitrary Lagrangian-Eulerian discontinuous Galerkin method for Euler equations*, Journal of Computational Physics, 470, 2022 111600.

[15] X. Gu, Y. Li, J. Cheng and C.-W. Shu, *A high order positivity-preserving conservative WENO remapping method based on a moving mesh solver*, Journal of Computational Physics, 473, 2023, 111754.

[16] Y. Gu, D. Luo, Z. Gao and Y. Chen, *An Adaptive Moving Mesh Method for the Five-equation Model*, Communications in Computational Physics, 32, 2021, 189-221.

[17] C. Klingenberg, G. Schnucke and Y. Xia, *Arbitrary Lagrangian-Eulerian discontinuous Galerkin method for conservation laws: Analysis and application in one dimension*, Mathematics of Computation, 86, 2017, 1203-1232.

[18] C. Klingenberg, G. Schnucke and Y. Xia, *An arbitrary Lagrangian-Eulerian local discontinuous Galerkin method for Hamilton-Jacobi equations*, Journal of Scientific Computing, 73, 2017, 906-942.

[19] L. Krivodonova, J. Xin, J.-F. Remacle, N. Chevaugeon and J.E. Flaherty, *Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws*, Applied Numerical Mathematics, 48, 2004, 323-338.

[20] M. Kucharik, M. Shashkov and B. Wendroff, *An efficient linearity-and-bound-preserving remapping method*, Journal of Computational Physics, 188, 2003, 462-471.

[21] N. Lei, J. Cheng and C.-W. Shu, *A high order positivity-preserving conservative WENO remapping method on 2D quadrilateral meshes*, Computer Methods in Applied Mechanics and Engineering, 373, 2021, 113497.

[22] N. Lei, J. Cheng and C.-W. Shu, *A high order positivity-preserving conservative WENO remapping method on 3D tetrahedral meshes*, Computer Methods in Applied Mechanics and Engineering, 395, 2022, 115037.

[23] N. Lei, J. Cheng and C.-W. Shu, *A high order positivity-preserving polynomial projection remapping method*, Journal of Computational Physics, 474, 2023, 111826.

[24] R. Li and T. Tang, *Moving mesh discontinuous Galerkin method for hyperbolic conservation laws*, Journal of Computational Physics, 27, 2006, 347-363.

[25] R. Li, T. Tang and P. Zhang, *Moving mesh methods in multiple dimensions based on harmonic maps*, Journal of Computational Physics, 170, 2001, 562-588.

[26] R. Li, T. Tang and P. Zhang, *A Moving mesh finite element algorithm for singular problems in two and three space dimensions*, Journal of Computational Physics, 177, 2002, 365-393.

[27] Y. Li, J. Cheng, Y. Xia and C.-W. Shu, *High order arbitrary Lagrangian-Eulerian finite difference WENO scheme for Hamilton-Jacobi equations*, Communications in Computational Physics, 26, 2019, 1530-1574.

[28] K. Lipnikov and N. Morgan, *A high-order conservative remap for discontinuous Galerkin schemes on curvilinear polygonal meshes*, Journal of Computational Physics, 399, 2019, 108931.

[29] K. Lipnikov and N. Morgan, *Conservative high-order discontinuous Galerkin remap scheme on curvilinear polyhedral meshes*, Journal of Computational Physics, 420, 2020, 109712.

[30] L. G. Margolin and M. Shashkov, *Second-order sign-preserving conservative interpolation (remapping) on general grids*, Journal of Computational Physics, 184, 2003, 266-298.

[31] L. G. Margolin and M. Shashkov, *Second-order sign-preserving remapping on general grids*, Report of Los Alamos National Laboratory, LA-UR-02-525.

[32] W.F. Noh, *Errors for calculations of strong shocks using an artificial viscosity and an artificial heat flux*, Journal of Computational Physics, 72, 1987, 78–120.

[33] A. L. Ortega and G. Scovazzi, *A geometrically-conservative, synchronized, flux-corrected remap for arbitrary Lagrangian–Eulerian computations with nodal finite elements*, Journal of Computational Physics, 230, 2011, 6709-6741.

[34] J. Qiu. and C.-W. Shu, *Runge-Kutta Discontinuous Galerkin Method Using WENO Limiters*, SIAM Journal on Scientific Computing, 26, 2015, 907–929.

[35] L.I. Sedov, *Similarity and Dimensional Methods in Mechanics*, Academic Press, New York, 1959.

[36] Z. Shen and G. Lv, *A remapping method based on multi-point flux corner transport upwind advection algorithm*, Journal of Computational Mathematics, 31, 2013, 592-619.

[37] C.-W. Shu, *TVB uniformly high-order schemes for conservation laws*, Mathematics of Computation, 49, 1987, 105-121.

[38] J. Velechovsky, J. Breil and R. Liska, *Flux corrected remapping using piecewise parabolic reconstruction for 2D cell-centered ALE methods*, International Journal for Numerical Method in Fluids, 76, 2014, 575-586.

[39] W. Wu, A. M. Zhang and M. Liu, *A cell-centered indirect Arbitrary-Lagrangian-Eulerian discontinuous Galerkin scheme on moving unstructured triangular meshes with topological adaptability*, Journal of Computational Physics, 438, 2021, 110368.

[40] M. Zhang, W. Huang and J. Qiu, *High-order conservative positivity-preserving DG-interpolation for deforming meshes and application to moving mesh DG simulation of radiative transfer*, SIAM Journal on Scientific Computing, 42, 2020, A3109-A3135.

[41] X. Zhang and C.-W. Shu, *On positivity preserving high order discontinuous Galerkin schemes for compressible Euler equations on rectangular meshes*, Journal of Computational Physics, 229, 2010, 8918-8934.

[42] X. Zhang, Y. Xia and C.-W. Shu, *Maximum-principle-satisfying and positivity-preserving high order discontinuous Galerkin schemes for conservation laws on triangular meshes*, Journal of Computational Physics, 50, 2012, 29–62.

[43] X. Zhao, X. Yu, M. Duan, Q. Fang and S. Zou, *An Arbitrary Lagrangian-Eulerian RKDG method for compressible Euler equations on unstructured meshes: Single-material flow.* Journal of Computational Physics, 396, 2019, 451–469.

[44] J. Zhu, J. Qiu and C.-W. Shu, *High-order Runge-Kutta discontinuous Galerkin methods with a new type of multi-resolution WENO limiters*, Journal of Computational Physics, 404, 2020, 109105.

[45] J. Zhu, J. Qiu and C.-W. Shu, *High-order Runge-Kutta discontinuous Galerkin methods with a new type of multi-resolution WENO limiters on triangular meshes*, Applied Numerical Mathematics, 153, 2020, 519-539.