Journal of Computational Physics 420 (2020) 109712

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

# Conservative high-order discontinuous Galerkin remap scheme on curvilinear polyhedral meshes

Konstantin Lipnikov [a,*], Nathaniel Morgan [b]

[a] Applied Mathematics and Plasma Physics Group, Theoretical Division, Mail Stop B284, Los Alamos National Laboratory, Los Alamos, NM 87545, United States of America
[b] Methods and Algorithms Group, X-Computational Physics Division, Mail Stop B284, Los Alamos National Laboratory, Los Alamos, NM 87545, United States of America

A R T I C L E   I N F O

*Article history:*
Received 7 January 2020
Received in revised form 7 July 2020
Accepted 7 July 2020
Available online 15 July 2020

*Keywords:*
Advection-based remap
High-order schemes
Polytopal meshes
Discontinuous Galerkin
Curvilinear meshes
Data remap

A B S T R A C T

A data transfer (remap) between two meshes is an important step of each arbitrary Lagrangian-Eulerian (ALE) simulation. We develop a conservative scheme for remapping high-order discontinuous Galerkin fields on high-order polytopal meshes with curved faces. This scheme uses a virtual element function to define the remap velocity. We show that the optimal accuracy is achieved when the remap problem is written and is solved as a coupled system of two conservative equations. The properties of the proposed scheme are studied numerically for smooth and discontinuous fields on cubic and prismatic meshes.

Published by Elsevier Inc.

## 1. Introduction

Lagrangian hydrodynamic codes use often general polytopal meshes and second-order discretization schemes. To increase predictive power of future codes, researchers are looking at high-order numerical schemes on high-order meshes [24,28]. Indirect arbitrary Lagrangian-Eulerian (ALE) hydrodynamics methods adopt a three-step approach to evolve the solution and mesh positions. The first step is a Lagrange calculation that deforms the mesh, the second step creates an improved mesh (e.g., a smoother mesh), and the third step remaps the physical quantities to the improved mesh. The last two steps are as critical for the overall simulation accuracy as the first step, since they must preserve characteristic mesh features as well as important mathematical and physical properties of the Lagrangian solution, see e.g. [21,11,12,29] and references therein. The focus of this paper is on remapping high-order discontinuous Galerkin (DG) solutions.

The first DG method for neutron transport equation was introduced in [32]. Typical DG methods have a number of properties important in applications. They are locally conservative, can handle complex geometries, support arbitrary (polytopal) meshes, and provide high-order accuracy. DG methods are used nowadays in a huge variety of engineering applications including compressible [7,14,27] and incompressible [31] fluid dynamics, granular flows [19], semiconductor device simulation [13], viscoelasticity [6], transport in porous media [2,4], and Lagrangian hydrodynamics on linear [30,34,24,27] and high-order (curvilinear) [35,28,15] meshes, see also references inside. DG methods have been employed to solve various other nonlinear equations: viscosity solutions of the general Hamilton-Jacobi equation are considered in [23], the Korteweg-deVries equations are studied in [33], the level set equation is analyzed in [26].

[*] Corresponding author.
*E-mail addresses:* lipnikov@lanl.gov (K. Lipnikov), nmorgan@lanl.gov (N. Morgan).

https://doi.org/10.1016/j.jcp.2020.109712
0021-9991/Published by Elsevier Inc.

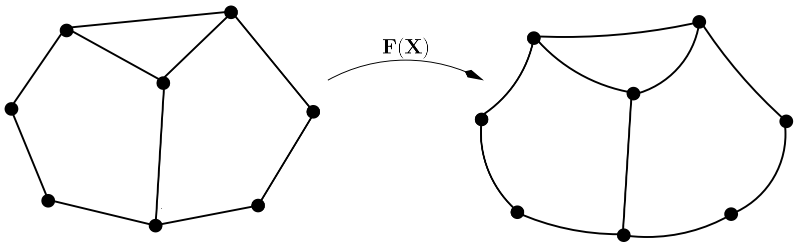

**Fig. 1.** Illustration of map $\mathbf{F}(\mathbf{X})$ from a reference (left) to a physical (right) mesh cell. Three visible faces are shown. Cell vertices are marked with solid disks.

For DG methods, there exist two major approaches to a conservative data remap. The first approach is based on the $L^2$ finite element projectors. Its practical implementation requires the intersection of two meshes which is a very challenging problem for high-order meshes with curved faces. But this approach is needed when mesh topology has changed during the mesh optimization step, see e.g. [29].

The second approach, that we employ in this paper, is to write the data remap as a dynamic process governed by a linear transport equation. If the difference between meshes is small, on the order of the mesh size, then the remap may be written as a simple flux-form transport algorithm [16]. The swept region algorithms work under similar assumption [21]. We consider a more general case where such an assumption is not valid. This PDE-type approach is natural for meshes with strongly curved faces and was analyzed in two-dimensions in [25]. Specifics of the two-dimensional remap problem is that the mesh deformation could be defined point-wise on mesh edges. Using exterior data, the deformation map is extended inside each polygonal cells using non-polynomials virtual element functions. The practical implementation uses virtual element projections that produce a polynomial approximation of an un-computable virtual element function. The virtual element projector minimizes energy norm of a polynomial which is a useful property for the stability analysis. When the cell is a triangle or a quadrilateral, the resulting map coincides with the conventional finite element map, see e.g. [15,3]. Finally, we mention a finite volume scheme from [12]. It is one of the rare schemes that deals with a conservative remapping on curvilinear grids.

In three-dimensions, the deformation map is defined pointwise on mesh edges; hence, it should be extended first inside mesh faces and then inside mesh cells. Development and analysis of the resulting 3D remap algorithm is the first novelty of this paper. This algorithm uses the serendipity virtual element method (VEM) to build a remap velocity. The VEM is a non-trivial generalization of the finite element to polytopal meshes [8,5,9] which emerged recently from the mimetic finite difference method [10]. Useful property of the VEM is that it remains stable numerically for non-convex and degenerate cells.

The mathematical formulation of the advection-based remap problem leads to a system of two coupled conservation equations for volume and mass. One equation describes dynamics of determinant of the Jacobian matrix, the other one describes the change in the remapped quantity. We show why the discrete volume conservation equation improves the remap error by one order and verify this with numerical experiments. This could be considered as the second novelty of this paper.

The paper outline is as follows. In Section 2, we introduce the PDE formulation of the advection-based remap problem. In Section 3, we present the numerical scheme and describe briefly construction of the virtual element projectors. In Section 4, we summarize briefly scheme's properties. In Section 5, we present results of numerical experiments on cubic and prismatic meshes for two different mesh deformation maps.

## 2. Advection based remap

Let us derive a PDE model for the advection-based remap problem. Consider a two meshes $\Omega_{X,h}$ and $\Omega_{x,h}$ called also the source and target meshes. In the context of the Lagrangian hydrodynamics, the first mesh is a Lagrangian mesh, while the second one is a modified (a higher quality) mesh. For simplicity, we assume that both meshes cover the computational domain $\Omega$ without gaps and overlaps. We also assume that the meshes are topologically equivalent, i.e. there exists a bijective map $\mathbf{F}\colon \Omega \to \Omega$ that maps cells to cells, faces to faces, edges to edges, and nodes to nodes, see Fig. 1.

Let $\mathbf{X} = (X_1, X_2, X_3)$ and $\mathbf{x} = (x_1, x_2, x_3)$ denote the position vectors on meshes $\Omega_{X,h}$ and $\Omega_{x,h}$, respectively. Hereafter, we refer to $\mathbf{X}$ as the reference coordinate system. For simplicity, we assume that the reference mesh $\Omega_{X,h}$ consists of polyhedral cells with flat faces and $\Omega_{x,h}$ consists of generalized polytops with curvilinear faces. We make the following additional assumptions.

**(A1)** The map $\mathbf{F}(\mathbf{X})$ is a vector polynomial of order $k$ on each mesh edge.

**(A2)** The map $\mathbf{F}(\mathbf{X})$ is sufficiently smooth so it could be approximated (with the optimal order) by a vector polynomial of order $k$ on each mesh face and inside each mesh cell.

Using the pseudo-time $\tau$, let us define a space-time map:

$$\mathbf{x}(\tau, \mathbf{X}) = \mathbf{X} + \tau(\mathbf{F}(\mathbf{X}) - \mathbf{X}), \qquad 0 \le \tau \le 1.$$

A remap "velocity" $\mathbf{u}$ can be defined as the rate of change of this map:

$$\mathbf{u}(\mathbf{X}) = \frac{\partial \mathbf{x}}{\partial \tau} = \mathbf{F}(\mathbf{X}) - \mathbf{X}. \tag{1}$$

Similar to a physical Lagrangian motion, this velocity defines a particle trajectory. A change of field $\rho$ along this trajectory is governed by the material derivative. By the nature of the remap problem, the field is not changing at each space point, i.e. $\partial\rho/\partial\tau = 0$. Using this and definition of the material derivative, we obtain the following advection equation:

$$\frac{d\rho}{d\tau} = \frac{\partial\rho}{\partial\tau} + \mathbf{u}\cdot\nabla_x\rho = \mathbf{u}\cdot\nabla_x\rho. \tag{2}$$

Hereafter, the operator's subscript indicates the coordinate frame. Note that remap equations for vector fields are derived similarly.

To derive a conservative formulation, we need the equation for volume change. Let $j$ be the determinant of the Jacobian $\mathbf{J} = \nabla_X\mathbf{x}$. We assume that $j > 0$ for $\tau \le 1$. The rate of change of the determinant, which captures the volume evolution, is (see e.g. [35]):

$$\frac{dj}{d\tau} = j\,\mathrm{div}_x\mathbf{u}. \tag{3}$$

This paper proposes a new method to solve the advection equation and the volume evolution with high-order accuracy on 3D curvilinear polytopal meshes (e.g., Voronoi-type grids with curved cell faces). Hereafter, we pursue the approach where the equations are solved in the reference coordinate system. Let us recall formulas for the change of coordinates in differential operators, see [35] for derivations:

$$\nabla_x\rho = \mathbf{J}^{-T}\,\nabla_X\rho, \qquad \mathrm{div}_x\mathbf{u} = \frac{1}{j}\mathrm{div}_X(j\mathbf{J}^{-1}\mathbf{u}). \tag{4}$$

Multiplying equation (2) by $j$, equation (3) by $\rho$, changing coordinates, and finally summing them up, we obtain the conservative formulation in the reference coordinates:

$$\frac{d(j\rho)}{d\tau} = j\,\mathbf{u}\cdot\mathbf{J}^{-T}\nabla_X\rho + \rho\,\mathrm{div}_X(j\mathbf{J}^{-1}\mathbf{u}) = \mathrm{div}_X(\rho\, j\mathbf{J}^{-1}\mathbf{u}). \tag{5}$$

Introducing an auxiliary velocity $\mathbf{v} = j\mathbf{J}^{-1}\mathbf{u}$, we conclude that the remap problem consists of two conservative equations:

$$\frac{d(j\rho)}{d\tau} = \mathrm{div}_X(\rho\,\mathbf{v}), \qquad \frac{dj}{d\tau} = \mathrm{div}_X\mathbf{v}. \tag{6}$$

Note that the obtained equations could also be derived using the Reynolds Transport Theorem. The second equation is actually a continuous form of the volume conservation equation, also known as the Geometric Conservation Law (GCL), which plays an important role in remapping. We show later, that it increases accuracy of the remap scheme. In other words, a numerical scheme for the first equation should use approximation of $j$ provided by the second equation.

## 3. Numerical scheme

### 3.1. Semi-discrete formulation

Since $\mathbf{F}$ is the bijective map, we use same symbols to denote related topological objects on meshes $\Omega_{X,h}$ and $\Omega_{x,h}$. We use $c$ for a cell, $f$ for a face, and $e$ for an edge. When the space-time position of a mesh object is important, we write $c(\tau)$, with $c(0) \in \Omega_{X,h}$ and $c(1) \in \Omega_{x,h}$. Let $\mathbf{n}_f(\tau, \mathbf{x})$ (or simply $\mathbf{n}_f$) be the unit normal vector to face $f$ at point $\mathbf{x}$ whose orientation is fixed once and for all. Furthermore, let $\mathbf{x}_c(\tau)$ denote the centroid of cell $c$, and $\mathbf{X}_c = \mathbf{x}_c(0)$. Finally, let $|c(\tau)|$ denote the cell volume and $|f(\tau)|$ denotes the face area.

By our assumption, mesh $\Omega_{X,h}$ has planar faces. In the ALE framework, this assumption could be simplified by requiring that only a mesh at the beginning of a simulation has flat faces. The recent progress in the theory of virtual element projectors allows us to drop even this assumption at expense of a more complex construction than that presented below.

In cell $c$, we define a moving basis $\psi_i^c(\tau, \mathbf{x})$ which has polynomial representation only at time $\tau = 0$, see formulas below. We assume that the basis function remains constant along particle trajectories, i.e.

$$\frac{d\psi_i^c}{d\tau} = 0. \tag{7}$$

At time zero, the basis functions are given by the Taylor polynomials. The lower-order basis functions are

$$\psi_0^c = 1, \quad \psi_1^c = a_1^c(X_1 - X_{c,1}), \quad \psi_2^c = a_2^c(X_2 - X_{c,2}), \quad \psi_3^c = a_3^c(X_3 - X_{c,3}).$$

Higher-order basis functions have the following form:

$$\psi_i^c = a_i^c\left((X_1 - X_{c,1})^{\alpha_i}(X_2 - X_{c,2})^{\beta_i}(X_3 - X_{c,3})^{\gamma_i} - b_i^c\,\psi_0^c\right), \quad i > 2,$$

where $\alpha_i$, $\beta_i$, and $\gamma_i$ are non-negative integers. The scaling factors $a_i^c$ and the orthonormalization factors $b_i^c$ are calculated from these two conditions:

$$\int_{c(0)} \psi_0^c\,\psi_i^c\,\mathrm{d}V = \delta_{0,i}\,|c(0)|, \qquad \int_{c(0)} \psi_i^c\,\psi_i^c\,\mathrm{d}V = |c(0)|, \tag{8}$$

where $\delta_{0,i}$ is the Kronecker symbol. The scaling factors improve spectral properties of mass matrices that appear later. Their proper selection increases the stability domain of an explicit time integration scheme.

A discrete space-time fields $\rho_h$ and $j_h$ in cell $c$ have the following expansions in the Taylor basis:

$$\rho_h|_c(\mathbf{x}) = \rho_0^c + \rho_1^c\,\psi_1^c(\mathbf{x}) + \rho_2^c\,\psi_2^c(\mathbf{x}) + \rho_3^c\,\psi_3^c(\mathbf{x}) + \ldots,$$
$$j_h|_c(\mathbf{x}) = j_0^c + j_1^c\,\psi_1^c(\mathbf{x}) + j_2^c\,\psi_2^c(\mathbf{x}) + j_3^c\,\psi_3^c(\mathbf{x}) + \ldots.$$

For a DG scheme of formal order $k+1$, each expansion is a polynomial of order $k$ with $(k+1)(k+2)(k+3)/6$ terms.

We use conventional definitions of the jump and average operators: Consider a face $f$ shared by two cells $c_1$ and $c_2$ and the normal vector pointing from $c_1$ to $c_2$. Then,

$$[\psi] = \psi|_{c_1} - \psi|_{c_2}, \quad \{\psi\} = \frac{1}{2}\left(\psi|_{c_1} + \psi|_{c_2}\right). \tag{9}$$

These definitions are extended to boundary faces by dropping out the $c_2$-terms.

Let us multiply both equations in (6) by a test function $\psi$ and integrate by parts. Since we work with discontinuous functions, we perform calculations cell-by-cell and then sum up the results. For the first equation, we have

$$\frac{d}{d\tau}\sum_c \int_c j\rho\psi\,\mathrm{d}V = -\sum_c \int_c \left(\mathbf{v}\cdot\nabla_X\psi\right)\rho\,\mathrm{d}V + \sum_f \int_f (\mathbf{v}\cdot\mathbf{n}_f)\rho\,[\psi]\,\mathrm{d}S. \tag{10}$$

Since $\mathbf{u}\cdot\mathbf{n}_f = 0$ on the domain boundary, $\mathbf{v}\cdot\mathbf{n}_f = j\mathbf{J}^{-1}\mathbf{u}\cdot\mathbf{n}_f = 0$ and the last term uses only internal faces. For the second equation in (6), we have

$$\frac{d}{d\tau}\sum_c \int_c j\psi\,\mathrm{d}V = -\sum_c \int_c \left(\mathbf{v}\cdot\nabla_X\psi\right)\mathrm{d}V + \sum_f \int_f (\mathbf{v}\cdot\mathbf{n}_f)\,[\psi]\,\mathrm{d}S. \tag{11}$$

Note that at the continuous level, this equation follows immediately from the previous one.

A semi-discrete formulation is obtained by replacing functions $\rho$ and $j$ by their cell-based Taylor expansions $\rho_h$ and $j_h$, respectively. To enforce the conservation law, we introduce a unique value of function $\rho$ on face $f$, which leads to the unique definition of a mass flux between cells. Since velocity $\mathbf{u}$ is unknown inside mesh cells, we approximate it using specially designed polynomial projectors $\Pi_{c,k}(\mathbf{u})$, $k > 0$, described later. The matrix $j\mathbf{J}^{-1}$ is approximated using definition of the Jacobian matrix which gives $\mathbf{J} = \mathbf{I} + \tau\nabla_X\mathbf{u}$. Hence, we take

$$\mathbf{J}_h = \mathbf{I} + \tau\nabla_X\Pi_{c,k}(\mathbf{u}). \tag{12}$$

We note that entries in the matrix of cofactors $\mathbf{C} \equiv j\mathbf{J}^{-1}$ are quadratic functions of entries in $\mathbf{J}$. Thus, a polynomial approximation $\mathbf{J}_h$ of $\mathbf{J}$ leads to a polynomial approximation $\mathbf{C}_c$ of $\mathbf{C}$. More details are provided below.

Uniqueness of the surface fluxes implies that we cannot use $\mathbf{C}_c$ in surface integrals. Instead, we introduce another approximation of $j\mathbf{J}^{-1}$, now on face $f$, denoted by $\mathbf{C}_f$. Notice that we need to approximate the matrix-vector product $j\mathbf{J}^{-T}\mathbf{n}_f$ which depends only on the velocity value on this face. As we show in the next subsection, vector $\mathbf{C}_f^T\mathbf{n}_f$, $f \in \partial c$, will be used in the hierarchical construction of matrix $\mathbf{C}_c$. Finally, we have

$$\frac{d}{d\tau}\sum_c \int_c j_h\rho_h\psi_h\,\mathrm{d}V = -\sum_c \int_c \left(\mathbf{C}_c\,\Pi_{c,k}(\mathbf{u})\cdot\nabla_X\psi_h\right)\rho_h\,\mathrm{d}V + \sum_f \int_f (\Pi_{f,k}(\mathbf{u})\cdot\mathbf{C}_f^T\,\mathbf{n}_f)\rho_h^*\,[\psi_h]\,\mathrm{d}S,$$

where $\Pi_{f,k}(\mathbf{u})$ is a specially designed polynomial projector on face $f$. Let us introduce a discrete velocity $\mathbf{v}_h$ such that $\mathbf{v}_h = \mathbf{v}_c \equiv \mathbf{C}_c\,\Pi_{c,k}(\mathbf{u})$ in mesh cells and $\mathbf{v}_h = \mathbf{v}_f \equiv \mathbf{C}_f\,\Pi_{f,k}(\mathbf{u})$ on mesh faces. Then,

$$\frac{d}{d\tau}\sum_c \int_c j_h \rho_h \psi_h \, \mathrm{d}V = -\sum_c \int_c \left(\mathbf{v}_c \cdot \nabla_X \psi_h\right) \rho_h \, \mathrm{d}V + \sum_f \int_f (\mathbf{v}_f \cdot \mathbf{n}_f) \rho_h^* \, [\psi_h] \, \mathrm{d}S. \tag{13}$$

The discrete form of the second equation is

$$\frac{d}{d\tau}\sum_c \int_c j_h \psi_h \, \mathrm{d}V = -\sum_c \int_c \left(\mathbf{v}_c \cdot \nabla_X \psi_h\right) \mathrm{d}V + \sum_f \int_f (\mathbf{v}_f \cdot \mathbf{n}_f) \, [\psi_h] \, \mathrm{d}S. \tag{14}$$

The stability argument suggests to select the downwind value for $\rho_h^*$:

$$(\mathbf{v}_f \cdot \mathbf{n}_f)\rho_h^* = (\mathbf{v}_f \cdot \mathbf{n}_f)\{\rho_h\} - \frac{1}{2}|\mathbf{v}_f \cdot \mathbf{n}_f| \, [\rho_h]. \tag{15}$$

By our assumption, $\mathbf{u}$ is a vector polynomial on each mesh edge. It is natural to require that projectors $\Pi_{c,k}(\mathbf{u})$ and $\Pi_{f,k}(\mathbf{u})$ be exact for vector polynomials of order $k$. This is the necessary condition for achieving an optimal approximation of $\mathbf{u}$ in certain functional spaces that completely characterized by velocity data on the mesh skeleton. With this requirement, all integrals in the weak formulation use polynomials and could be computed with the divergence theorem. Integration of surface fluxes based on formula (15) requires more comments. We can either split face $f$ into triangles and apply this formula at each Gauss quadrature point or define one upwind direction for the whole face. In our numerical experiments, we use the second approach.

### 3.2. Fully discrete scheme

#### 3.2.1. Time discretization

We use the third order TVD Runge-Kutta (RK) scheme [18] to derive a fully discrete scheme, although other RK schemes can be also used. When we rewrite the weak formulation (13)-(14) in a matrix form, each RK substep looks like a forward Euler step. Let $\boldsymbol{\rho}_h^{n,i}$, $\mathbf{j}_h^{n,i}$ be the algebraic vectors of coefficients in all Taylor expansions at time level $n$ and the RK substep $i$. Note that $\rho_h^{n,0} = \rho_h^n$ and $j_h^{n,0} = j_h^n$. The left hand side of equation (13) produces the matrix $\mathbf{M}_\rho(\mathbf{j}_h^{n,i})$ which is the conventional mass matrix for the weighted $L^2$ product with the weight $j_h$. Note that $\mathbf{M}_\rho(\mathbf{j}^{n,0}) = \mathbf{I}$. Similarly, the left hand side of equation (14) produces a matrix $\mathbf{M}_j$ that does not depend on the pseudo-time and can be computed once. Then each RK substep is

$$\begin{aligned} \frac{\mathbf{M}_\rho(\mathbf{j}^{n,i+1})\boldsymbol{\rho}_h^{n,i+1} - \mathbf{M}_\rho(\mathbf{j}_h^{n,i})\boldsymbol{\rho}_h^{n,i}}{\Delta\tau} &= \mathbf{G}_\rho^{n,i}, \\ \mathbf{M}_j \frac{\mathbf{j}_h^{n,i+1} - \mathbf{j}_h^{n,i}}{\Delta\tau} &= \mathbf{G}_j^{n,i}, \end{aligned} \tag{16}$$

where $\mathbf{G}_\rho^{n,i}$, $\mathbf{G}_j^{n,i}$ are the discrete functionals combining volumetric and surface terms at the previous RK substeps, and $\Delta\tau$ is the pseudo-time step.

Note that introduction of the auxiliary variable $\mathbf{w}_h^{i,n} = \mathbf{M}_\rho(\mathbf{j}_h^{n,i})\boldsymbol{\rho}_h^{n,i}$ and the auxiliary functional $\widehat{\mathbf{G}}_j^{n,i} = \mathbf{M}_j^{-1}\mathbf{G}_j^{n,i}$ may help to reuse the existing RK codes.

#### 3.2.2. Numerical implementation

This subsection describes a brief over of the remap approach, the equations that must be solved, and the order in which the equations are solved. The first concept in our approach is to express/approximate everything (the maps and unknowns) as Taylor-series polynomials in terms of the Lagrangian coordinates $\mathbf{X}$. The second concept is to make approximations that are exact for the polynomials of order $k$. The remap equations (6) and spatial maps require a mesh velocity along the cell face and inside the cell, which are both calculated following the ideas from VEM. Each velocity component on a cell face $f$ is projected onto a 2D Taylor-series polynomial $u_f = \sum_i \psi_i(\mathbf{X}) u_{i,f}$ using

$$\int_f \nabla_X u_f \cdot \nabla_X \psi_q \, \mathrm{d}S = -\int_f \overline{u}_f (\nabla_X^2 \psi_q) \, \mathrm{d}S + \oint_{\partial f} (\nabla_X \psi_q \cdot \mathbf{n}) u \, \mathrm{d}L \qquad \forall \psi_q, \tag{17}$$

which is equivalent to formula (25) below with a VEM projector in place of $u_f$. Here $\overline{u}_f$ is a least-square fit to the mesh velocity values on the face skeleton. The mesh velocity on the skeleton is known and corresponds to the mesh smoothing process. All integrals are over the mesh prior to the remap (locations given by the coordinates $\mathbf{X}$). Solving this system of equations gives the coefficients $u_{i,f}$ needed to describe the mesh velocity polynomial for the face.

Each component of the mesh velocity inside a cell $c$ is projected onto a 3D Taylor-series polynomial in the cell, $u_c = \sum_i \psi_i(\mathbf{X}) u_{i,c}$, using

$$\int_c \nabla_X u_c \cdot \nabla_X \psi_q \, \mathrm{d}V = -\int_c \overline{u}_c (\nabla_X^2 \psi_q) \, \mathrm{d}V + \oint_{\partial c} (\nabla_X \psi_q \cdot \mathbf{n}) u_f \, \mathrm{d}S \qquad \forall \psi_q, \tag{18}$$

which is equivalent to formula (22) below with a VEM projector in place of $u_c$. Here $\overline{u}_c$ is a least-squares fit to the velocity values on the cell skeleton, and $u_f$ is the polynomial velocity on the cell face. Solving this system of equations gives the coefficients $u_{i,c}$ needed to describe the mesh velocity polynomial for the interior of the cell.

Given three velocity components from equation (18), we calculate the polynomial vector mesh velocity $\mathbf{u}_h$. The polynomial approximation of Jacobian $\mathbf{J}$ is given by $\mathbf{J}_h = \mathbf{I} + \tau \nabla_X \mathbf{u}_h$. Instead of using $j_h = \det(\mathbf{J}_h)$, we solve a separate equation to evolve the polynomial coefficients in $j_c = \sum_i \psi_i(\mathbf{X}) j_i$ forward in time using the second equation in (16). Finally, we solve the first equation in (16) for an unknown polynomial field coefficients in $\rho_c = \sum_i \psi_i(\mathbf{X}) \rho_i$. The pseudo time derivatives are approximated using the third-order accurate TVD RK method.

### 3.3. *Virtual element projector on a polytopal cell*

Here we highlight important aspects of the mathematical theory behind computation of projectors $\Pi_{c,k}(\mathbf{u})$ and $\Pi_{f,k}(\mathbf{u})$. In other words, given a cell $c$ and a velocity polynomial on mesh edges, we want to build polynomial approximation of velocity inside $c$. The construction is done independently for each velocity component. We assume that $k > 0$. For the case $k = 0$, we build projectors of order 1.

Let $u$ be a velocity component in a virtual element space $\mathcal{V}_k(c)$ that we formalize along with our derivations. From the approximation view point, this space must contain the polynomial space $\mathcal{P}^k(c)$ and the projectors $\Pi_{c,k}(\mathbf{u})$ and $\Pi_{f,k}(\mathbf{u})$ must be exact for polynomials from this space. The theory of virtual elements introduces multiple projectors. Our construction uses orthogonal projectors that minimize the $H^1$ semi-norm of the resulting polynomial.

The classical definition of the orthogonal projector reads: Find polynomial $\Pi_{c,k}(u) \in \mathcal{P}^k(c)$ such that

$$\int_c \nabla \Pi_{c,k}(u) \cdot \nabla q \, \mathrm{d}V = \int_c \nabla u \cdot \nabla q \, \mathrm{d}V \qquad \forall q \in \mathcal{P}^k(c), \tag{19}$$

subject to

$$\int_c \Pi_{c,k}(u) \, \mathrm{d}V = \int_c u \, \mathrm{d}V, \quad \text{if} \quad k > 1, \tag{20}$$

or

$$\int_{\partial c} \Pi_{c,k}(u) \, \mathrm{d}V = \int_{\partial c} u \, \mathrm{d}V, \quad \text{if} \quad k = 1. \tag{21}$$

Integrating by parts the right-hand side of (19), we obtain the equivalent definition of the projector:

$$\int_c \nabla \Pi_{c,k}(u) \cdot \nabla q \, \mathrm{d}V = -\int_c u \, \Delta q \, \mathrm{d}V + \int_{\partial c} u (\nabla q \cdot \mathbf{n}) \, \mathrm{d}S \qquad \forall q \in \mathcal{P}^k(c). \tag{22}$$

We develop further the surface integral by adding the first requirement to the space $\mathcal{V}_k(c)$. Let $\mathcal{B}(f)$ denote the trace of $\mathcal{V}_k(c)$ on face $f \in \partial c$. Then, we require that the $L^2$ projector $\Pi^0_{f,k-1}(u_{|f})$ is computable (knowing velocity only on edges of face $f$) for any $u_{|f} \in \mathcal{B}(f)$, i.e.

$$\int_f \Pi^0_{f,k-1}(u|_f) \, p \, \mathrm{d}S = \int_f u \, p \, \mathrm{d}S \quad \forall p \in \mathcal{P}^{k-1}(f). \tag{23}$$

The second requirement to the space $\mathcal{V}_k(c)$ is that moments of $u$ are computable upto order $k-2$. This implies that the second integral in (22) is computable. This also implies that we know the orthogonal $L^2$ projector $\Pi^0_{c,k-2}$:

$$\int_c \Pi^0_{c,k-2}(u) \, p \, \mathrm{d}V = \int_c u \, p \, \mathrm{d}V \quad \forall p \in \mathcal{P}^{k-2}(c).$$

#### 3.3.1. *Two-dimension space $\mathcal{B}(f)$*

Nowadays, there exist enough publications on construction of the virtual space $\mathcal{B}(f)$ and the $L^2$ projector $\Pi^0_{f,k-1}$. For this reason, we present major steps (sufficient to reproduce the scheme) in the construction and avoid precise definitions of virtual spaces. We refer the reader to [9] for these definitions and other technical details. Recall, all steps should be exact for polynomials of order $k$.

Let $v = u|_f$. The staring point is the least-square algorithm which compute a polynomial of order $k$ using velocity data on edges of face $f$. We formally write this algorithm as the linear operator $\mathcal{I}_f^{LS,k}(v)$. For a triangular face, the least-square algorithm is stable up to $k = 2$. For a quadrilateral, this bound becomes $k = 3$. For a convex pentagon, this bound is $k = 4$. In our opinion $k = 3$ is more than enough for hydrodynamic codes. We define the moments in the right-hand side of (23) as follows:

$$\int_f v\, p\, \mathrm{d}S = \int_f \mathcal{I}_f^{LS,k}(v)\, p\, \mathrm{d}S \qquad \forall p \in \mathcal{P}^{k-1}(f). \tag{24}$$

Thus, the first requirement for space $\mathcal{B}(f)$ is that the least-square operator becomes the desired $L^2$ projector, i.e. $\Pi^0_{f,k-1} = \mathcal{I}_f^{LS,k}$.

Now we build the desired projector $\Pi_{f,k}(\mathbf{u})$. The integration by parts gives

$$\int_f \nabla\Pi_{f,k}(v)\cdot\nabla p\, \mathrm{d}S = -\int_f v\, \Delta p\, \mathrm{d}S + \int_{\partial f} v(\nabla p\cdot\mathbf{n})\, \mathrm{d}L \qquad \forall p \in \mathcal{P}^k(f). \tag{25}$$

To fix the constant part of this projector we impose conditions similar to conditions (20) and (21). By the definition of $\mathcal{B}(f)$, the moments of $v \in \mathcal{B}(f)$ are computable upto order $k-1$. Thus, we have only computable integrals in the right-hand side. The projector is computed by inverting the Gramm-Schmidt matrix appearing in the left-hand side. This projector has the following property. Taking $p = \Pi_{f,k}(v)$, we bound the left-hand side as follows:

$$\|\nabla\Pi_{f,k}(v)\|^2_{L^2(f)} = \int_f \nabla v\cdot\nabla\Pi_{f,k}(v)\, \mathrm{d}S \le \|\nabla\Pi_{f,k}(v)\|_{L^2(f)}\, \|\nabla v\|_{L^2(f)}, \tag{26}$$

i.e. this projector reduces the energy norm of the resulting polynomial. Note that $\mathcal{I}^{LS,k}$ does not have this property.

**Remark 3.1.** The virtual space $\mathcal{B}(f)$ is not unique. For instance, we can use a different space that satisfies (24) with $k$ instead of $k-1$. This new space allows us to compute the $L^2$ and $H^1$ projectors with the optimal approximation properties. In practice, the space with less additional constraints in (24) shows better approximation properties.

### 3.3.2. *The $L^2$ projector $\Pi^0_{c,k-2}$*

To define the orthogonal projector $\Pi^0_{c,k-2}$, we again use the least-square approach to build a polynomial of order $k$ in cell $c$. But now we use all velocity data on edges of $c$. We formally write this algorithm as $\mathcal{I}_c^{LS,k}(u)$ and define $\Pi^0_{c,k-2} = \mathcal{I}_c^{LS,k}(u)$.

The virtual element space $\mathcal{V}_{c,k}$ with the above properties allows us to calculate the desired projector $\Pi_{c,k}$ from this formula:

$$\int_c \nabla\Pi_{c,k}(u)\cdot\nabla q\, \mathrm{d}V = -\int_c \mathcal{I}_c^{LS,k}(u)\, \Delta q\, \mathrm{d}V + \sum_{f\in\partial c}\int_f \mathcal{I}_f^{LS,k}(u)(\nabla q\cdot\mathbf{n})\, \mathrm{d}S \qquad \forall q \in \mathcal{P}^k(c). \tag{27}$$

The global <u>continuous</u> virtual element space $\mathcal{V}_k$ is defined by cell-based spaces $\mathcal{V}_{c,k}$. Let $v \in \mathcal{V}_k$. The degrees of freedom are moment of $v$ on mesh edges up order $k$. The space properties are as follows:

1. $v \in \mathcal{P}^k(e)$ for any mesh edge $e$;
2. $v \in \mathcal{B}(f)$ for any mesh face $f$ and such that $\mathcal{I}_f^{LS,k}(v) = \Pi^0_{f,k-1}(v)$;
3. $\mathcal{I}_c^{LS,k}(v) = \Pi^0_{c,k-2}(v)$.

To find the trace of $v$ on face $f$ we need to solve two-dimension Poisson equation on $f$ with the Dirichlet boundary data and the specially designed source term, see [26] for more detail. Similarly, a three-dimensional Poisson equation in cell $c$ has to be solved to find the actual shape of $v$. Despite the implicit definition of the virtual element space, the local projectors $\Pi_{f,k}$ and $\Pi_{c,k}$ can be computed from the degrees of freedom. These projectors are all what we need to build a scheme.

**Remark 3.2.** We could reproduce the finite elements deformation maps for a hexahedral cell by building projectors of order $k+1$. However, this only increases the cost of the scheme. Hence, the proposed scheme is not equivalent to a finite element scheme in [3] on hexahedral meshes.

### 3.4. *Limiting of discontinuous solutions*

Remap of a discontinuous solution requires a carefully designed limiting strategy to avoid oscillations around discontinuities. As the proof of principle, we use the Barth-Jespersen limiter, although we see no obvious obstacles in using more complex limiters such as the accuracy preserving hierarchical limiter from [22].

Let $\hat{\rho}_h$ be the limited solution. As discussed in [25], specifics of the remap problem is that the limiting strategy consists of two steps that we briefly summarize here. First, we limit $\rho_h$ in each cell by multiplying high-order terms by a single non-negative factor:

$$\hat{\rho}_h\big|_c = \rho_0^c + \alpha_c\big(\rho_1^c\,\psi_1^c + \rho_2^c\,\psi_2^c + \rho_3^c\,\psi_3^c + \dots\big), \quad 0 \le \alpha_c \le 1.$$

The factor $\alpha_c$ is calculated such that the limited function satisfies the following inequalities:

$$\min_{c'\in\mathcal{F}_c}\{\rho_0^{c'}\} \le \hat{\rho}_h\big|_c(\mathbf{X}_k) \le \max_{c'\in\mathcal{F}_c}\{\rho_0^{c'}\}, \quad \forall \mathbf{X}_k \in \mathcal{G}_c,$$

where $\mathcal{F}_c$ is the set of cells that have at least one common point with cell $c$ and $\mathcal{G}_c$ is a set of limiting points. We define $\mathcal{G}_c$ as the set of Gauss points on edges of $c$. We use one Gauss point for $k<2$ and two Gauss points for $k=2$. Second, we restore the conservation law which implies that the integral of $\rho_h j_h$ must be preserved during the limiting strategy, see the next section. This is achieved by shifting the mean value of the limited function in each mesh cell:

$$\hat{\rho}_0^c = \alpha_c\,\rho_0^c + (1-\alpha_c)\frac{1}{|c(\tau)|}\int\limits_{c(\tau)} \rho_h\, j_h\, \mathrm{d}V. \tag{28}$$

## 4. Properties of the scheme

### 4.1. *The conservation law*

Taking $\psi_h = 1$ in formula (13), we have

$$\frac{d}{d\tau}\sum_c \int\limits_c \rho_h\, j_h\, \mathrm{d}V = 0.$$

Thus, the weighted integral of $\rho_h$ is conserved. Using the orthogonality property of basis functions at time $\tau=0$ and the fact that $j_h = 1$ at this time moment, we rewrite this equation as follows:

$$\sum_c \int\limits_c \rho_h^{n+1}\, j_h\, \mathrm{d}V = \sum_c (\rho_0^c)^0\,|c|. \tag{29}$$

### 4.2. *Justification of the volume equation*

The purpose of this section is not to provide a detailed error analysis but give a simple explanation of the impact of equation (14) on critical error terms. Our focus will be on the spatial error, so that we restrict discussion to semi-discrete weak formulations. We refer the reader to [17,20] where the importance of the discrete geometric conservation law is studied for finite volume schemes.

Since equation (14) is decoupled from (13), we can employ the existing error analysis for hyperbolic problem (see, e.g. [36]) that gives

$$\|j - j_h\|_{L^2(\Omega)} \le C h^{k_1} \tag{30}$$

where $k_1$ is a function of $k$ and $h$ is the characteristic mesh size. Hereafter, $C$ denotes a generic constant.

Let us introduce a trilinear form

$$\mathcal{L}(\alpha,\beta,\gamma) = \int\limits_c \alpha\,\beta\,\gamma\, \mathrm{d}V.$$

Let us take $\psi = \psi_h = \varepsilon_h$ in weak formulations (10) and (13), where $\varepsilon_h = \rho_h - \Pi_k^0(\rho)$. We take the difference of these equations and analyze each term separately. For the accumulation term, we have

$$\mathcal{L}(j,\varepsilon_h,\rho) - \mathcal{L}(j_h,\varepsilon_h,\rho_h) = -\mathcal{I}_1 + \mathcal{I}_2 + \mathcal{I}_3 + \mathcal{I}_4,$$

where

$$\begin{aligned}\mathcal{I}_1 &= \mathcal{L}(j, \varepsilon_h, \varepsilon_h), & \mathcal{I}_2 &= \mathcal{L}(j - j_h, \varepsilon_h, \varepsilon_h),\\ \mathcal{I}_3 &= \mathcal{L}(j - j_h, \varepsilon_h, \Pi^0_{c,k}(\rho)), & \mathcal{I}_4 &= \mathcal{L}(j, \varepsilon_h, \rho - \Pi^0_{c,k}(\rho)).\end{aligned}$$

The first term is a part of the error energy definition. The other terms describe how the error grows. The second term is smaller than the first term due to (30), so it changes only the generic constant in the error estimate. The fourth term has the optimal approximation property, $O(h^k)$, due to properties of the $L^2$ projector. The third term has sub-optimal approximation properties since $j_h$ depends on derivatives of $\mathbf{u}_h$. Here we need the second equation. Let us take $\psi = \psi_h = \varepsilon_h \Pi_0^{c,0}(\rho)$ in weak formulations (11) and (14). The difference of two equations gives

$$\mathcal{L}(j, \varepsilon_h \Pi^0_0(\rho), 1) - \mathcal{L}(j_h, \varepsilon_h \Pi^0_0(\rho), 1) = \mathcal{L}(j - j_h, \varepsilon_h \Pi^0_{c,0}(\rho), 1) = \mathcal{L}(j - j_h, \varepsilon_h, \Pi^0_{c,0}(\rho)),$$

since $\Pi^0_0(\rho)$ is a constant. Subtracting the last formula form $\mathcal{I}_3$, we obtain

$$\widetilde{\mathcal{I}}_3 = \mathcal{I}_3 - \mathcal{L}(j - j_h, \varepsilon_h, \Pi^0_{c,0}(\rho)) = \mathcal{L}(j - j_h, \varepsilon_h, \Pi^0_k(\rho) - \Pi^0_{c,0}(\rho)). \tag{31}$$

The difference of two $L^2$ projectors gives addition factor $h$ to the error estimate. The same strategy can be applied to the remaining terms in the weak formulation. We verify this gain of accuracy with numerical experiments.

## 5. Numerical experiments

To generate a modified mesh $\Omega_{x,h}$, we consider two deformation maps given by the vorticial motion (VM map) and the compression/expansion map (CE map). In the VM map, the nodes of a given mesh are moved according to the following ODEs:

$$\begin{aligned}\dot{x}_1 &= 0.2 \sin(\pi x_1) \cos(\pi x_2) \sin(\pi x_3/2),\\ \dot{x}_2 &= -0.2 \cos(\pi x_1) \sin(\pi x_2) \sin(\pi x_3/2),\\ \dot{x}_3 &= 0.0,\end{aligned}$$

with the initial values $\mathbf{x}(0) = \mathbf{X}$ and the final integration time 1. The VM map preserves cell volume; however, only approximately in the discrete setting. Indeed, the remap velocity $\mathbf{u}$ linearizes the continuum motion which introduces a small error.

The CE map is given by the quartic velocity:

$$\mathbf{u} = \frac{1}{2} X_1\, X_2\, X_3\, (\mathbf{1} - \mathbf{X}).$$

We consider two sequences of cubic and prismatic meshes. The original and two modified meshes are shown in Fig. 2 for modest mesh resolutions. Note that prismatic cells degenerate near domain boundary, some dihedral angles are 180°.

Calculation of the discrete $L^2$ error, denoted by $\epsilon(\rho_h)$, deserves a few comments. We cannot use centroid $\mathbf{x}_c$ of cell $c \in \Omega_{x,h}$, since the basis functions cannot be evaluated there. But we can use values of basis functions at cell vertices $\mathbf{x}_v$ where the remap velocity is well defined. We define the discrete error as follows:

$$\epsilon(\rho_h) = \Big[ \sum_{c \in \Omega_{x,h}} \frac{|c|}{\mathcal{N}_c} \sum_{v \in c} |\rho(\mathbf{x}_v) - \rho_h(\mathbf{x}_v))|^2 \Big]^{1/2}, \tag{32}$$

where $\mathcal{N}_c$ is the number of vertices in cell $c$.

The visualization is done with the Paraview software [1] using the solution mean values $\rho_0^c$. The curvature of mesh edges for $k > 1$ is not resolved in the presented figures, although it is present in the calculations.

### 5.1. Numerical integration over polytopal cells

Efficiency of the proposed technology is based on the feasibility to integrate polynomial functions over arbitrarily-shared cells. Let $q$ be a monomial of degree $n$. The Euler homogeneous function theorem reads

$$\int_c q\, \mathrm{d}V = \frac{1}{3+n} \sum_{f \in \partial c} \int_f (\mathbf{X} \cdot \mathbf{n}_f)\, q\, \mathrm{d}S. \tag{33}$$

Polynomial $(\mathbf{X} \cdot \mathbf{n}_f)\, q$ is a linear combination of monomials; hence, we can employ the two-dimensional version of the Euler homogeneous function theorem for each monomial. This leaves only edge integrals of polynomials that could be computed efficiently using the Gauss-Legendre quadrature formulas.

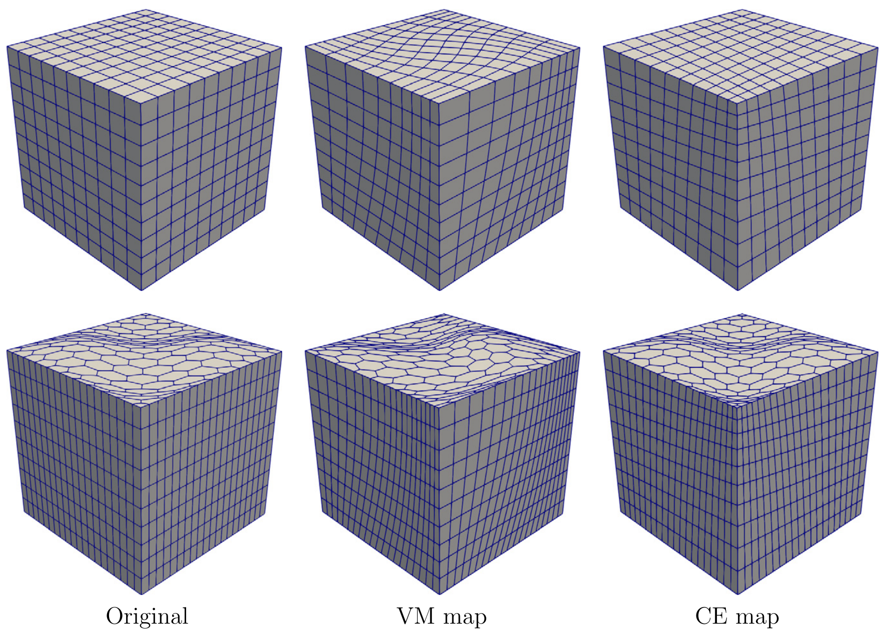

**Fig. 2.** Top row: original (left) and two modified (right) cubic meshes. Bottom row: original (left) and two modified (right) prismatic meshes.

**Table 1**
VM map: the discrete $L^2$ and maximum errors for DG($\mathcal{P}^k$) schemes on a sequence of cubic meshes.

| $1/h$ | DG($\mathcal{P}^0$) | | DG($\mathcal{P}^1$) | | DG($\mathcal{P}^2$) | |
|---|---|---|---|---|---|---|
| | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ |
| 10 | 1.700e-1 | 4.147e-1 | 4.013e-2 | 1.126e-1 | 4.662e-3 | 1.470e-2 |
| 20 | 8.746e-2 | 2.095e-1 | 1.078e-2 | 3.669e-2 | 5.766e-4 | 1.888e-3 |
| 40 | 4.434e-2 | 1.057e-1 | 3.001e-3 | 1.597e-2 | 7.438e-5 | 3.138e-4 |
| rate | 0.969 | 0.986 | 1.871 | 1.409 | 2.985 | 2.775 |

## 5.2. Smooth solution: logically cubic meshes

Let $\Omega$ be the unit cube. To study the formal order of convergence of the proposed scheme, we consider a smooth solution given by

$$\rho(X_1, X_2, X_3) = \sin(3X_1)\sin(6X_2)\sin(4X_3). \tag{34}$$

In the first experiment, we compare the unlimited DG($\mathcal{P}^k$) solutions for cubic meshes and the VM map. The results are collected in Table 1. Hereafter, the convergence rates are calculated using the linear regression algorithm. We suppress the time integration error by using the time step that is smaller than that required for scheme stability. The time step on the coarsest mesh is 0.025 and is reduced twice for each mesh refinement. Notice that with each increment of $k$, we gain almost one order of accuracy on the intermediate mesh.

In the second experiment, we compare the DG($\mathcal{P}^k$) solutions for the CE map. The results collected in Table 2 confirm our previous conclusions. On cubic meshes, the projector $\Pi_{c,k}$ does not recover the conventional finite element map, i.e. $\Pi_{c,k}(\mathbf{u})$ is discontinuous across mesh faces. Still, our results are in agreement with the results reported in [3] for the finite element method on curvilinear meshes. Indeed, the virtual element space is continuous by construction; hence, the optimal convergence rate is expected.

## 5.3. Smooth solution: prismatic meshes

In this section, we consider again the smooth solution (34) but the sequence of prismatic meshes shown in Fig. 2. In two sets of experiments, we compare the DG($\mathcal{P}^k$) solutions for the VM and CE maps. The results are collected in Tables 3 and 4, respectively.

The benefits of using higher-order DG schemes are evident also in the case of polyhedral meshes. The lack of optimal convergence of the DG($\mathcal{P}^0$) scheme requires a rigorous analysis which is beyond the scope of this paper.

**Table 2**
CE map: the discrete $L^2$ and maximum errors for DG($\mathcal{P}^k$) schemes on a sequence of cubic meshes.

| $1/h$ | DG($\mathcal{P}^0$) | | DG($\mathcal{P}^1$) | | DG($\mathcal{P}^2$) | |
|---|---|---|---|---|---|---|
| | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ |
| 10 | 1.693e-1 | 3.898e-1 | 3.869e-2 | 1.063e-1 | 4.685e-3 | 1.387e-2 |
| 20 | 8.611e-2 | 2.179e-1 | 9.941e-3 | 2.994e-2 | 5.521e-4 | 2.122e-3 |
| 40 | 4.336e-2 | 1.136e-1 | 2.529e-3 | 7.802e-3 | 6.632e-5 | 2.969e-4 |
| rate | 0.983 | 0.889 | 1.968 | 1.884 | 3.071 | 2.773 |

**Table 3**
VM map: the discrete $L^2$ and maximum errors for DG($\mathcal{P}^k$) schemes on a sequence of prismatic meshes.

| #cells | DG($\mathcal{P}^0$) | | DG($\mathcal{P}^1$) | | DG($\mathcal{P}^2$) | |
|---|---|---|---|---|---|---|
| | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ |
| $10^3$ | 1.718e-1 | 5.425e-1 | 4.070e-2 | 1.675e-1 | 4.191e-3 | 3.960e-2 |
| $20^3$ | 1.033e-1 | 4.315e-1 | 1.128e-2 | 5.070e-2 | 6.331e-4 | 7.737e-3 |
| $40^3$ | 5.942e-2 | 3.394e-1 | 2.956e-3 | 1.363e-2 | 8.107e-5 | 8.059e-4 |
| rate | 0.767 | 0.338 | 1.892 | 1.810 | 2.846 | 2.809 |

**Table 4**
CE map: the discrete $L^2$ and maximum errors for DG($\mathcal{P}^k$) schemes on a sequence of prismatic meshes.

| #cells | DG($\mathcal{P}^0$) | | DG($\mathcal{P}^1$) | | DG($\mathcal{P}^2$) | |
|---|---|---|---|---|---|---|
| | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ |
| $10^3$ | 1.590e-1 | 5.146e-1 | 4.174e-2 | 1.986e-1 | 4.709e-3 | 4.116e-2 |
| $20^3$ | 8.763e-2 | 3.064e-1 | 1.153e-2 | 6.014e-2 | 6.735e-4 | 6.509e-3 |
| $40^3$ | 4.620e-2 | 1.981e-1 | 3.017e-3 | 1.722e-2 | 8.741e-5 | 9.397e-4 |
| rate | 0.892 | 0.687 | 1.895 | 1.763 | 2.876 | 2.726 |

**Table 5**
Direct calculation of $j_h$: the discrete $L^2$ and maximum errors for DG($\mathcal{P}^k$) schemes on a sequence of cubic meshes.

| $1/h$ | DG($\mathcal{P}^0$) | | DG($\mathcal{P}^1$) | | DG($\mathcal{P}^2$) | |
|---|---|---|---|---|---|---|
| | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ |
| 10 | 1.700e-1 | 4.147e-1 | 4.173e-2 | 1.380e-1 | 4.800e-3 | 1.487e-2 |
| 20 | 8.746e-2 | 2.095e-1 | 1.412e-2 | 6.006e-2 | 6.418e-4 | 2.095e-3 |
| 40 | 4.434e-2 | 1.057e-1 | 6.222e-3 | 2.778e-2 | 1.024e-4 | 3.308e-4 |
| rate | 0.969 | 0.986 | 1.373 | 1.156 | 2.775 | 2.745 |

Let us compare the DG schemes for a given level of accuracy, see Table 4. On a Linux cluster with 24 cores, the DG($\mathcal{P}^0$) scheme on the finest mesh runs approximately 20 times longer than the DG($\mathcal{P}^1$) scheme of the coarsest mesh. The DG($\mathcal{P}^1$) scheme on the finest mesh runs approximately 6 times longer than the DG($\mathcal{P}^2$) scheme of the coarsest mesh. Significant CPU time is spent on calculation of the remap velocities $\mathbf{v}_f$ and $\mathbf{v}_c$. Since these are polynomial functions not only in space but also in the pseudo time $\tau$, we recommend to compute coefficients in the $\tau$-polynomial only once per simulation.

## 5.4. Impact of the volume equation

In the next experiment, we calculate $j_h$ directly from the remap velocity, so effectively, we drop the volume conservation equation. We consider only the VM map. The results are shown in Tables 5 and 6.

Observe the reduced convergence rate compared to the results in Tables 1 and 3. Approximately one order has been lost on prismatic meshes. Also notice that lose of the convergence rate is less drastic for cubic meshes, especially for the DG($\mathcal{P}^2$) scheme. We attribute this behavior to additional error cancellation on structured meshes for the divergence-free map. Indeed, for the CE map the convergence rates for the DG($\mathcal{P}^2$) scheme are 2.432 and 2.052 (compare with Table 2).

**Table 6**
Direct calculation of $j_h$: the discrete $L^2$ and maximum errors for DG($\mathcal{P}^k$) schemes on a sequence of prismatic meshes.

| *#cells* | DG($\mathcal{P}^0$) | | DG($\mathcal{P}^1$) | | DG($\mathcal{P}^2$) | |
|---|---|---|---|---|---|---|
| | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ | $L^2$ | $L^\infty$ |
| $10^3$ | 1.718e-1 | 5.425e-1 | 5.132e-2 | 4.903e-1 | 6.282e-3 | 7.043e-2 |
| $20^3$ | 1.103e-1 | 4.316e-1 | 2.259e-2 | 2.807e-1 | 1.542e-3 | 2.217e-2 |
| $40^3$ | 5.942e-2 | 3.394e-1 | 1.105e-2 | 1.517e-1 | 3.966e-4 | 5.989e-3 |
| rate | 0.766 | 0.338 | 1.108 | 0.846 | 1.993 | 1.778 |

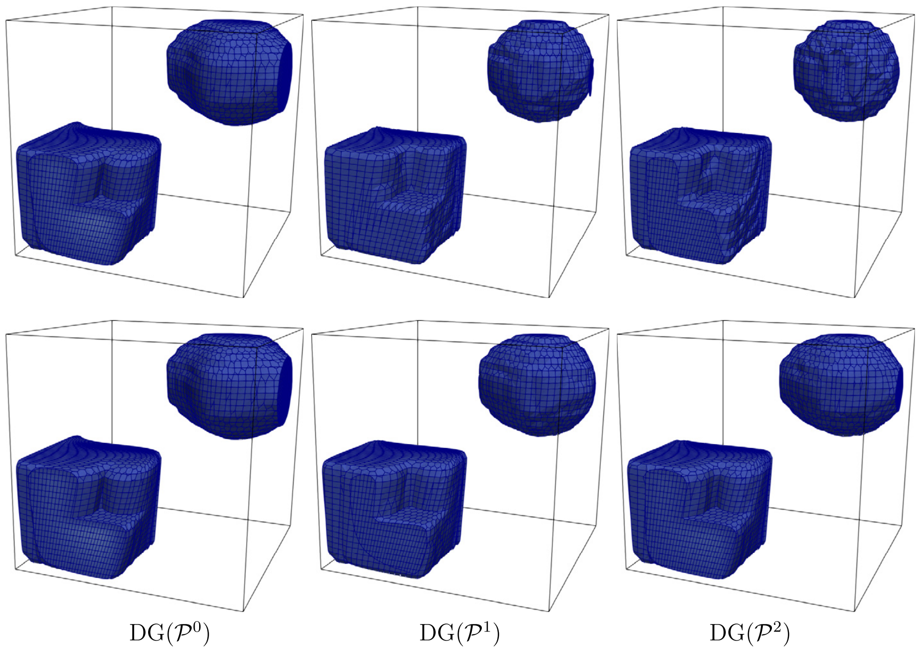

**Fig. 3.** Top row: unlimited solution for the finest prismatic mesh and VM map. Bottom row: limited solution.

## 5.5. *Discontinuous solution*

Discontinuity in a solution is typical for hyperbolic systems. Numerical diffusion and unphysical oscillations around discontinuity effect significantly the quality of the remapped solution. Limiters and a shock detection algorithms are typically used to restrict solution dissipation to shock location and to suppress oscillations. We consider the Barth-Jespersen limiter with the correction suggested in [25], see also Sec. 3.4.

The computational domain is again the unit cube. The solution is given by the characteristic functions of the notched cube and sphere. The cube has edge size $a_0 = 0.4$ and is centered at point (0.25, 0.25, 0.25). The notch is formed by removing a smaller cube with edge size $a_1 = 0.2$ centered at point (0.15, 0.15, 0.15). The sphere has radius $a_3 = 0.2$ and is centered at point (0.75, 0.75, 0.75).

We consider the meshes introduced above. To make the numerical experiment relevant for applications, we break the pseudo-time interval (0, 1) into 10 uniform segments and perform 10 advection solves. The reference coordinate system is the same for all solves and is given at time 0.

In Figs. 3 and 4 we compare limited and unlimited solutions on the finest prismatic meshes obtained with VM and CE maps, respectively. We plot solution isovolume for $\rho_c > 0.01$. Solution bounds are reported in Table 7. In all experiments, the mass conservation law is satisfied up to machine precision.

The lack of limiters leads to moderate overshoots and undershoots around the solution discontinuity. The violation of bounds is between 0.05 and 0.11, depending on the experiment, and is slightly less for the DG($\mathcal{P}^2$) solution. The Barth-Jespersen limiter suppresses the oscillations. The reported violation of bounds is between 0 and 0.0003, due to the fact that our limiter is not mathematically monotone for the remap problem.

As expected, the Barth-Jespersen limiter smears solution near discontinuity. The experiment shows significant shape improvement for $k = 1$ and $k = 2$. Notice, that both high-order DG schemes looks similar in the eye-ball norm; although on average the limiting was more aggressive for the DG($\mathcal{P}^1$) scheme. This indicates that higher-order limiters may be needed for higher-order DG schemes. Also, the initial piecewise constant solution profile is not well suited for showing impact of

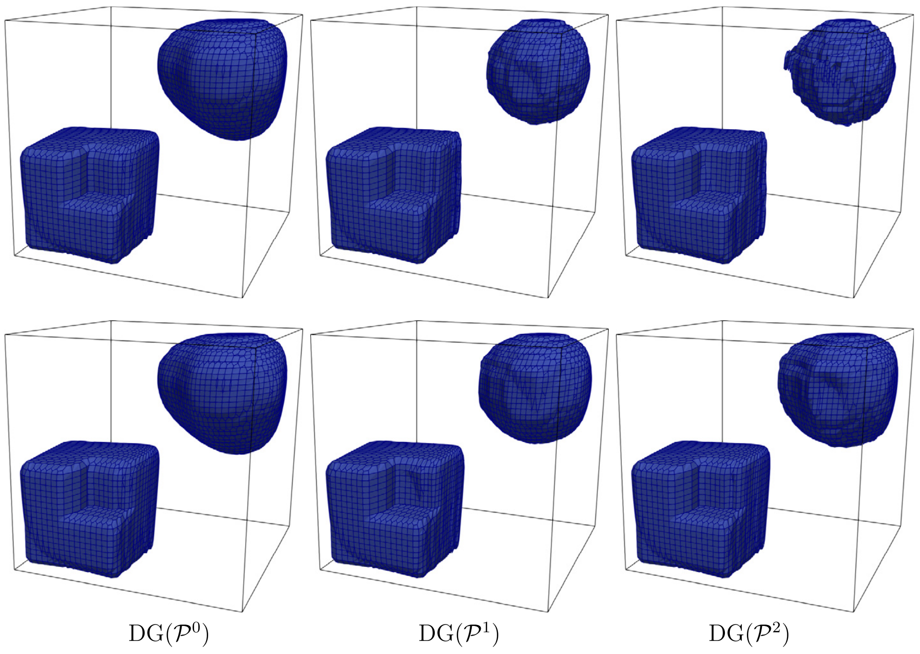

**Fig. 4.** Top row: unlimited solution for the finest prismatic mesh and CE map. Bottom row: limited solution.

**Table 7**
Bounds on the mean values in the remapped solution.

| | DG($\mathcal{P}^0$) | | DG($\mathcal{P}^1$) | | DG($\mathcal{P}^2$) | |
|---|---|---|---|---|---|---|
| | $\rho_{\max}$ | $\rho_{\min}$ | $\rho_{\max}$ | $\rho_{\min}$ | $\rho_{\max}$ | $\rho_{\min}$ |
| cubes VM | 1.0 | 0.0 | 1.103 | -0.0706 | 1.083 | -6.86e-2 |
| cubes VM lim | 1.0 | 0.0 | 1.0 | 0.0 | 1.0 | -1.54e-28 |
| cubes CE | 1.0 | 0.0 | 1.107 | -0.0583 | 1.075 | -0.0548 |
| cubes CE lim | 1.0 | 0.0 | 1.0 | -6.96e-31 | 1.0 | 0.0 |
| prisms VM | 1.0 | 0.0 | 1.093 | -8.44e-2 | 1.076 | -8.16e-2 |
| prisms VM lim | 1.0 | 0.0 | 1.0 | -2.63e-4 | 1.0 | -1.13e-4 |
| prisms CE | 1.0 | 0.0 | 1.113 | -6.67e-2 | 1.082 | -7.37e-2 |
| prisms CE lim | 1.0 | 0.0 | 1.0 | -1.79e-43 | 1.0 | -1.20e-35 |

higher-order schemes. To show this impact, additional work is needed on shock captured schemes and visualization tools. Our preliminary two-dimensional experiments revealed that the shock detectors developed for the Lagrangian hydrodynamics require additional analysis for the remap problem which is beyond the scope of this work. Finally, the visualization tool Paraview interpolates cell-centered values to mesh nodes which is a diffusive process.

## 6. Conclusion

The remap problem is an important step in most arbitrarily Lagrangian-Eulerian (ALE) hydrodynamics simulations. We proposed a new technology to remap high-order discontinuous Galerkin solutions between two polytopal meshes with curved edges and faces. The proposed scheme uses globally continuous uncomputable virtual element remap velocity and its computable polynomial projections in each mesh cell. The projectors are computed from velocity information on cell edges.

For smooth solutions, we demonstrated the $k+1$ order of convergence of the DG($\mathcal{P}^k$), $k=1,2$, scheme on cubic and prismatic meshes in both discrete $L^2$ and maximum norms. The numerical experiments have shown significant computational advantage for using these schemes compared to the DG($\mathcal{P}^0$) scheme. For discontinuous solutions, we verified numerically the modified scalar Barth-Jespersen limiter proposed in [25]. The numerical experiments on polytopal meshes have shown significant reduction of overshoots and undershoots in the limited solution.

## CRediT authorship contribution statement

Konstantin Lipnikov: Conceptualization; Methodology; Writing, Reviewing and Editing; Code Implementation.

Nathaniel Morgan: Conceptualization; Methodology; Writing, Reviewing and Editing.

## Declaration of competing interest

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

## Acknowledgements

Research presented in this article was supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project number 20170051DR. Los Alamos National Laboratory is operated by Triad National Security, LLC for the U.S. Department of Energy NNSA under Contract No. 89233218CNA000001.

## References

[1] J. Ahrens, B. Geveci, C. Law, Paraview: an end-user tool for large data visualization, in: Visualization Handbook, Elsevier, 2005, pp. 717–731.
[2] V. Aizinger, C. Dawson, B. Cockburn, P. Castillo, Local discontinuous Galerkin method for contaminant transport, Adv. Water Resour. 24 (2000) 73–87.
[3] R. Anderson, V. Dobrev, T. Kolev, R. Rieben, Monotonicity in high-order curvilinear finite element arbitrary Lagrangian-Eulerian remap, Int. J. Numer. Methods Fluids 77 (2015) 249–273.
[4] P.F. Antonietti, C. Facciola, A. Russo, M. Verani, Discontinuous Galerkin approximation of flows in fractured porous media on polytopic grids, SIAM J. Sci. Comput. 41 (1) (2019) A109–A138.
[5] B. Ayuso de Dios, K. Lipnikov, G. Manzini, The nonconforming virtual element method, M2AN: Math. Model. Numer. Anal. 50 (3) (2016) 879–904.
[6] F. Baaijens, Application of low-order discontinuous Galerkin methods to the analysis of viscoelastic flows, J. Non-Newton. Fluid Mech. 52 (2) (1994) 37–57.
[7] F. Bassi, S. Rebay, High-order accurate discontinuous finite element solution of the 2D Euler equations, J. Comput. Phys. 138 (1997) 251–285.
[8] L. Beirão da Veiga, F. Brezzi, A. Cangiani, G. Manzini, L.D. Marini, A. Russo, Basic principles of virtual element methods, Math. Models Methods Appl. Sci. 23 (2013) 119–214.
[9] L. Beirão da Veiga, F. Brezzi, L.D. Marini, A. Russo, Serendipity nodal VEM spaces, Comput. Fluids 141 (2016) 2–12.
[10] L. Beirao da Veiga, K. Lipnikov, G. Manzini, The Mimetic Finite Difference Method for Elliptic PDEs, Springer, 2014, 408 pp.
[11] D. Benson, An efficient, accurate, simple ALE method for nonlinear finite element programs, Comput. Methods Appl. Mech. Eng. 72 (1989) 305–350.
[12] B. Boutin, E. Deriaz, P. Hoch, P. Navaro, Extension of ALE methodology to unstructured conical meshes, ESAIM Proc. 22 (2011) 31–55.
[13] Z. Chen, B. Cockburn, C. Gardner, J. Jerome, Quantum hydrodynamic simulation of hysteresis in the resonant tunneling diode, J. Comput. Phys. 117 (1995) 274–280.
[14] B. Cockburn, C.-W. Shu, The Runge-Kutta discontinuous Galerkin finite element method for conservation laws V: multidimensional systems, J. Comput. Phys. 141 (1998) 199–224.
[15] V. Dobrev, T. Kolev, R. Rieben, High-order curvilinear finite element methods for Lagrangian hydrodynamics, SIAM J. Sci. Comput. 34 (5) (2012) 606–641.
[16] J. Dukowicz, J. Baumgardner, Incremental remapping as a transport/advection algorithm, J. Comput. Phys. 160 (2000) 318–335.
[17] C. Farhat, P. Geuzaine, C. Grandmont, The discrete geometric conservation law and the nonlinear stability of ALE schemes for the solution of flow problems on moving grids, J. Comput. Phys. 174 (2) (2001) 669–694.
[18] S. Gottlieb, C.-W. Shu, Total variation dimishing Runge-Kutta schemes, Math. Comput. 67 (221) (1998) 73–85.
[19] P. Gremaud, J. Matthews, Discontinuous Galerkin methods for Friedrichs' systems. I. General theory, J. Comput. Phys. 166 (2) (2001) 63–83.
[20] H. Guillard, C. Farhat, On the significance of the geometric conservation law for flow computations on moving meshes, Comput. Methods Appl. Mech. Eng. 190 (2000) 1467.
[21] M. Kucharik, M. Shashkov, B. Wendroff, An efficient linearity-and-bound-preserving remapping methods, J. Comput. Phys. 188 (2003) 462–471.
[22] D. Kuzmin, A vertex-based hierarchical slope limiter for $p$-adaptive discontinuous Galerkin methods, J. Comput. Phys. 233 (12) (2010) 3077–3085.
[23] O. Lepsky, C. Hu, C.-W. Shu, Analysis of the discontinuous Galerkin method for Hamilton-Jacobi equations, Appl. Numer. Math. 33 (2000) 423–434.
[24] Z. Li, X. Yu, Z. Jia, The cell-centered discontinuous Galerkin method for Lagrangian compressible Euler equations in two-dimensions, Comput. Fluids 96 (2014) 152–164.
[25] K. Lipnikov, N. Morgan, A high-order conservative remap for discontinuous Galerkin schemes on curvilinear polygonal meshes, J. Comput. Phys. 399 (2019) 108931.
[26] K. Lipnikov, N. Morgan, A high-order discontinuous Galerkin method for level set problems on polygonal meshes, J. Comput. Phys. 397 (2019) 108834.
[27] X. Liu, N. Morgan, D. Burton, A Lagrangian discontinuous Galerkin hydrodynamic method, Comput. Fluids 163 (2018) 68–85.
[28] X. Liu, N. Morgan, D. Burton, A high-order Lagrangian discontinuous Galerkin hydrodynamic method for quadratic cells using a subcell mesh stabilization scheme, J. Comput. Phys. 386 (1) (2019) 110–157.
[29] R. Loubère, P.-H. Maire, M. Shashkov, J. Breil, S. Galera, ReALE: a reconnection-based arbitrary Lagrangian-Eulerian method, J. Comput. Phys. 229 (2010) 4724–4761.
[30] R. Loubère, J. Ovadia, R. Agrall, A Lagrangian discontinuous Galerkin-type method on unstructured meshes to solve hydrodynamics problems, Int. J. Numer. Methods Fluids 44 (2004) 645–663.
[31] N. Nguyen, J. Peraire, B. Cockburn, An implicit high-order hybridizable discontinuous Galerkin method for the incompressible Navier-Stokes equations, J. Comput. Phys. 230 (4) (2011) 1147–1170.
[32] W. Reed, T. Hill, Triangular mesh methods for the neutron transport equation, Technical Report LA-UR-73-479, Los Alamos National Laboratory, 1973.
[33] C.-W. Shu, J. Yan, A local discontinuous Galerkin method for kdv type equations, SIAM J. Numer. Anal. 40 (2002) 769–791.
[34] F. Vilar, Cell-centered discontinuous Galerkin discretization for two-dimensional Lagrangian hydrodynamics, Comput. Fluids 64 (2012) 64–73.
[35] F. Vilar, P.-H. Maire, R. Abgrall, A discontinuous Galerkin discretization for solving two-dimensional gas dynamics equations written unter total Lagrangian formulation on general unstructured grids, J. Comput. Phys. 276 (2014) 188–234.
[36] Q. Zhang, C.-W. Shu, Stability analysis and a priori error estimates of the third order explicit Runge-Kutta discontinuous Galerkin method for scalar conservation laws, SIAM J. Numer. Anal. 48 (3) (2010) 1038–1063.