

High-Order Conservative Solution Transfer for Anisotropically Adapted Tetrahedral Meshes

T. Levý* and G. May**

Corresponding author: levyto@kme.zcu.cz

* Department of Mechanics, Faculty of Applied Sciences,
University of West Bohemia, Pilsen, Czech Republic.

** Division of Natural and Applied Sciences,
Duke Kunshan University, Kunshan, China.

Abstract: We present a conservative high-order solution transfer procedure for three-dimensional tetrahedral meshes. The method is formulated as a local Galerkin projection between non-matching meshes and relies on a robust supermesh construction based on tetrahedron–tetrahedron intersections. Particular emphasis is placed on the geometric robustness of the three-dimensional supermesh construction, as well as on conservation and preservation of formal accuracy. The transfer procedure is assessed in two complementary settings: first, by repeated projection of analytical functions between non-matching meshes, and second, within a time-accurate hybridized discontinuous Galerkin (HDG) framework with predictor-based anisotropic mesh adaptation. Numerical experiments confirm optimal high-order convergence, preservation of integral quantities, and stability under successive applications in fully three-dimensional adaptive simulations.

Keywords: Three-dimensional solution transfer, Conservative Galerkin projection, Supermesh construction, Anisotropic mesh adaptation, Hybridized discontinuous Galerkin.

1. Introduction

High-order numerical methods are increasingly used for the simulation of unsteady flow problems involving localized and evolving solution features. In this context, the hybridized discontinuous Galerkin (HDG) method provides an efficient high-order spatial discretization by reducing globally coupled degrees of freedom through hybridization [1]. Combined with implicit time-integration schemes such as diagonally implicit Runge–Kutta (DIRK) methods, HDG enables stable and time-accurate simulations of convection-diffusion and compressible flow problems [2]. For such problems, adaptive mesh techniques are particularly attractive, since they allow spatial resolution to be redistributed dynamically according to the evolving solution structure [3].

In time-dependent adaptive computations, however, mesh modification introduces an additional algorithmic difficulty: the numerical solution must be transferred repeatedly between successive non-matching meshes. This transfer is not merely a technical interpolation step. For piecewise polynomial high-order approximations, it must preserve conservation, maintain the formal order of accuracy of the overall discretization, and remain robust under repeated application throughout the time-integration process. The importance of conservative remapping in moving- and adaptive-mesh settings has been recognized for decades, going back to classical works such as Dukowicz and Kodis [4] and Grandy [5].

A broad class of conservative remapping and interpolation procedures has been developed for mesh-to-mesh transfer. Classical overlap-based conservative remapping methods, as in [5], established the importance of conservation through explicit element-overlap constructions. In parallel, related developments on general polyhedral meshes [6] further emphasized the role of conservative and accurate remapping on complex cell topologies. In the context of anisotropic mesh adaptation, several conser-

vative interpolation strategies were later developed primarily for low-order solution representations and Hessian-based metric frameworks, for example by Alauzet and Mehrenberger [7] and in subsequent related works [8, 9, 10, 11]. These methods are particularly relevant from the adaptive CFD perspective, but they do not directly address the repeated transfer of general high-order discontinuous polynomial fields.

For high-order approximations, Galerkin projection provides a natural L^2 -optimal transfer strategy, and supermesh-based formulations have proved particularly suitable for conservative projection between non-matching meshes. In this direction, Farrell et al. [12] introduced conservative interpolation between unstructured meshes via supermesh construction, and Farrell and Maddison [13] extended the local Galerkin projection viewpoint to three-dimensional meshes. A related but distinct high-order line of work is represented by the transfer operators of Pazner and Kolev [14], who developed conservative and accurate mappings between finite element spaces of different polynomial orders. Although the present paper focuses on transfer between meshes within the same polynomial approximation space, these results illustrate the broader relevance of conservative high-order transfer operators.

In our previous work [15], a conservative supermesh-based transfer methodology was developed and analyzed for two-dimensional non-matching meshes. That methodology was subsequently incorporated into a time-accurate anisotropic mesh adaptation framework for unsteady problems in [16]. Although the numerical assessment in the present paper is carried out in the same adaptive framework based on the HDG discretization, the transfer itself is not restricted to HDG and applies more generally to piecewise polynomial solution representations. The present contribution therefore continues a line of development in which conservative supermesh-based transfer, previously established in two dimensions and in time-accurate adaptive computations, is extended to three-dimensional tetrahedral meshes.

This extension is geometrically non-trivial. While in two dimensions the intersection of two elements yields a polygon whose connectivity can be recovered relatively simply from the localized vertices alone, in three dimensions the intersection of two tetrahedra produces a convex polyhedron with potentially intricate topology. The corresponding robust supermesh construction requires localization of all intersection points, reconstruction of the intersection polyhedron, and subsequent subdivision into tetrahedral integration sub-elements.

From this perspective, the main challenge in three dimensions is no longer the algebraic form of the transfer operator itself, but the robustness and efficiency of the underlying geometric primitives. This view is consistent with recent developments in computational geometry and interface reconstruction, such as the robust polyhedron-intersection framework of Chiodi and Desjardins [17] and the recent tetrahedron–tetrahedron intersection work of McCoid and Gander [18].

The present work develops a conservative high-order solution transfer methodology for anisotropically adapted three-dimensional tetrahedral meshes. The transfer is formulated as a local Galerkin projection and evaluated over a supermesh constructed from tetrahedron–tetrahedron intersections. Particular emphasis is placed on the geometric robustness of the three-dimensional supermesh construction, including localization of intersection points, reconstruction of the intersection polyhedron, and its subdivision into tetrahedral integration sub-elements. The resulting transfer procedure is first verified independently of any governing equation and is then assessed within a time-accurate adaptive framework based on the HDG spatial discretization, through a model scalar convection-diffusion problem.

2. Formulation of Conservative Solution Transfer

As in the two-dimensional setting, the conservative solution transfer is formulated as a local Galerkin projection between non-matching meshes. This formulation is independent of the spatial dimension. The extension to three dimensions therefore affects primarily the geometric part of the procedure, namely the construction of the supermesh associated with tetrahedral intersections.

To introduce the notation, let $\mathcal{T}_h^1$ and $\mathcal{T}_h^2$ denote two non-matching tetrahedral meshes of a computational domain $\Omega \subset \mathbb{R}^3$, representing the source and target tessellations. Their topology may differ arbitrarily, for example as a result of anisotropic mesh adaptation. For $i = 1, 2$, let

$$V_h^i = \{v \in L^2(\Omega) : v|_\kappa \in \mathcal{P}^p(\kappa), \forall \kappa \in \mathcal{T}_h^i\} \quad (1)$$

denote the associated piecewise polynomial space of degree p , and let $w_h^1 \in V_h^1$ be a given high-order discontinuous approximation on $\mathcal{T}_h^1$. The objective is to construct an approximation $w_h^2 \in V_h^2$ on $\mathcal{T}_h^2$ such that conservation and high-order accuracy are preserved.

A natural choice is to seek an approximation w_h^2 of w_h^1 on $\mathcal{T}_h^2$ that minimizes the L^2 -norm error, i.e.

$$\|w_h^1 - w_h^2\|_{L^2(\Omega)} = \min_{v \in V_h^2} \|w_h^1 - v\|_{L^2(\Omega)}. \quad (2)$$

This corresponds to the L^2 projection of w_h^1 onto the space V_h^2 defined on the target mesh $\mathcal{T}_h^2$. Owing to the discontinuous character of the approximation, the problem can be written locally on each element $\kappa \in \mathcal{T}_h^2$ as

$$\|w_h^1 - w_h^2\|_{L^2(\kappa)} = \min_{v \in \mathcal{P}^p(\kappa)} \|w_h^1 - v\|_{L^2(\kappa)}. \quad (3)$$

The local problem can be equivalently formulated in a weak sense as a Galerkin projection: for each $\kappa \in \mathcal{T}_h^2$, find w_h^2 such that

$$\int_\kappa w_h^2 \varphi \, d\mathbf{x} = \int_\kappa w_h^1 \varphi \, d\mathbf{x}, \quad \forall \varphi \in \mathcal{P}^p(\kappa). \quad (4)$$

Choosing the basis functions $\varphi_2^{(i)}$ spanning $\mathcal{P}^p(\kappa)$ as test functions, the local Galerkin projection can be written as

$$\int_\kappa w_h^2 \varphi_2^{(i)} \, d\mathbf{x} = \int_\kappa w_h^1 \varphi_2^{(i)} \, d\mathbf{x}, \quad i = 1, \dots, N_2^\kappa, \quad (5)$$

where $N_2^\kappa = \dim(\mathcal{P}^p(\kappa))$ and $\varphi_2^{(i)} \equiv \varphi_2^{(i)}(\mathbf{x})$, $\mathbf{x} \in \kappa$. This formulation yields a locally conservative projection, since the constant function belongs to $\mathcal{P}^p(\kappa)$ for any polynomial degree. In the present work, hierarchical Dubiner basis functions are employed [19].

For clarity, we explicitly indicate the element dependence of the basis functions and coefficients in the following expansions. Both fields, w_h^1 and w_h^2 , can be represented on each element by linear combinations of the corresponding basis functions as

$$w_h^1(\mathbf{x}) = \sum_{j=1}^{N_1^{\bar{\kappa}}} W_{1,\bar{\kappa}}^{(j)} \varphi_{1,\bar{\kappa}}^{(j)}(\mathbf{x}), \quad \mathbf{x} \in \bar{\kappa}, \quad \bar{\kappa} \in \mathcal{T}_h^1, \quad (6)$$

$$w_h^2(\mathbf{x}) = \sum_{i=1}^{N_2^\kappa} W_{2,\kappa}^{(i)} \varphi_{2,\kappa}^{(i)}(\mathbf{x}), \quad \mathbf{x} \in \kappa, \quad \kappa \in \mathcal{T}_h^2, \quad (7)$$

where $W_{1,\bar{\kappa}}^{(j)}$ are the expansion coefficients associated with the source element $\bar{\kappa} \in \mathcal{T}_h^1$, $\varphi_{1,\bar{\kappa}}^{(j)}$ are the basis functions spanning $\mathcal{P}^p(\bar{\kappa})$, and $N_1^{\bar{\kappa}} = \dim(\mathcal{P}^p(\bar{\kappa}))$. The coefficients $W_{2,\kappa}^{(i)}$ associated with the target element $\kappa \in \mathcal{T}_h^2$ are to be determined.

Since the meshes do not share the same topology, the right-hand side of (5) cannot be evaluated directly. The function w_h^1 is defined on $\mathcal{T}_h^1$, whose elements do not align with those of $\mathcal{T}_h^2$, and the integrals

must therefore be evaluated over the intersections of $\kappa \in \mathcal{T}_h^2$ with elements of $\mathcal{T}_h^1$. To this end, a *local* supermesh is introduced. Each target element κ is subdivided into tetrahedral subregions induced by these intersections, and the projection integrals are computed as sums over the resulting sub-elements.

For each pair $(\kappa_1, \kappa_2) \in \mathcal{T}_h^1 \times \mathcal{T}_h^2$ with non-empty intersection, the region $\kappa_{12} = \kappa_1 \cap \kappa_2$ forms a *convex polyhedron*, which is subsequently subdivided into tetrahedra for the purpose of numerical integration. The local supermesh is the conforming tetrahedral mesh formed by all such sub-elements associated with a target element $\kappa \in \mathcal{T}_h^2$. We denote it by

$$\mathcal{H}_\kappa = \{K_\ell, \ell = 1, \dots, N_\kappa\}, \quad (8)$$

where N_κ is the number of local supermesh elements.

Substituting the expansions (6) and (7) into (5) and decomposing the element integral on the right-hand side into a sum over the local supermesh elements, $\int_\kappa = \sum_{\ell=1}^{N_\kappa} \int_{K_\ell}$, yields

$$\int_\kappa \sum_{k=1}^{N_2^\kappa} W_{2,\kappa}^{(k)} \varphi_{2,\kappa}^{(k)} \varphi_{2,\kappa}^{(i)} \, d\mathbf{x} = \sum_{\ell=1}^{N_\kappa} \int_{K_\ell} \sum_{j=1}^{N_1^{\bar{\kappa}_\ell}} W_{1,\bar{\kappa}_\ell}^{(j)} \varphi_{1,\bar{\kappa}_\ell}^{(j)} \varphi_{2,\kappa}^{(i)} \, d\mathbf{x}, \quad i = 1, \dots, N_2^\kappa. \quad (9)$$

For each supermesh element $K_\ell \in \mathcal{H}_\kappa$, let $\bar{\kappa}_\ell \in \mathcal{T}_h^1$ denote the unique source element such that $K_\ell \subset \bar{\kappa}_\ell$. The source basis functions and coefficients in each integral are then understood with respect to the corresponding element $\bar{\kappa}_\ell$. In the present work, the same polynomial degree p is assumed on both meshes, so that the number of basis functions is identical on all source and target elements. We therefore write $N_1 \equiv N_1^{\bar{\kappa}_\ell}$ and $N_2 \equiv N_2^\kappa$. This leads to the local linear system

$$\mathbf{M}_2^\kappa \mathbf{W}_2^\kappa = \sum_{\ell=1}^{N_\kappa} \mathbf{M}_{1,2}^{K_\ell} \mathbf{W}_1^{\bar{\kappa}_\ell} \quad (10)$$

for unknown coefficients $\mathbf{W}_2^\kappa = (W_{2,\kappa}^{(1)}, \dots, W_{2,\kappa}^{(N_2)})^T$ of the target element κ , $\mathbf{M}_2^\kappa \in \mathbb{R}^{N_2 \times N_2}$ is the local mass matrix on the target element κ , $\mathbf{M}_{1,2}^{K_\ell} \in \mathbb{R}^{N_2 \times N_1}$ is the mixed mass matrix associated with the supermesh element K_ℓ coupling the source and target basis functions, and $\mathbf{W}_1^{\bar{\kappa}_\ell} = (W_{1,\bar{\kappa}_\ell}^{(1)}, \dots, W_{1,\bar{\kappa}_\ell}^{(N_1)})^T$ are the known coefficients of the source element $\bar{\kappa}_\ell$. The matrix entries are given by

$$(\mathbf{M}_2^\kappa)_{ij} = \int_\kappa \varphi_{2,\kappa}^{(i)} \varphi_{2,\kappa}^{(j)} \, d\mathbf{x}, \quad i, j = 1, \dots, N_2, \quad (11)$$

and

$$(\mathbf{M}_{1,2}^{K_\ell})_{ij} = \int_{K_\ell} \varphi_{2,\kappa}^{(i)} \varphi_{1,\bar{\kappa}_\ell}^{(j)} \, d\mathbf{x}, \quad i = 1, \dots, N_2, \quad j = 1, \dots, N_1. \quad (12)$$

The algebraic form of the transfer problem is therefore identical to the two-dimensional case. The main additional complexity in three dimensions lies in the geometric construction of the local supermesh: while intersections of triangles yield polygons that can be triangulated directly, intersections of tetrahedra produce convex polyhedra with more involved topology and connectivity. In particular, the intersection points may arise from vertex inclusion, edge–face intersections, and edge–edge intersections, and the resulting polyhedra must be reconstructed and subdivided in a robust manner. The corresponding geometric construction is described in the following section.

3. Three-Dimensional Supermesh Construction

The algebraic formulation of the conservative solution transfer derived in the previous section requires the construction of a local supermesh over which the mixed projection integrals can be evaluated. In three dimensions, this task reduces to the geometric treatment of intersections between tetrahedral elements from the source and target meshes.

In contrast to the planar case, where element intersections yield polygons, see previous works [15, 16], the intersection of two tetrahedra produces a convex polyhedron whose topology may vary depending on the relative configuration of the elements. Robust geometric treatment is therefore essential, since inaccuracies in the identification or subdivision of these intersection regions may compromise conservation and degrade the formal accuracy of the transfer.

The construction of the three-dimensional local supermesh is decomposed into four main steps. First, the geometric setting and the localization of candidate intersecting elements are introduced. Next, all intersection points between two tetrahedra are identified. These points are then used to reconstruct the intersection polyhedron, which is subsequently subdivided into tetrahedral sub-elements suitable for numerical integration.

3.1 Geometric Setting

For each target element $\kappa \in \mathcal{T}_h^2$, the local supermesh is constructed from all source elements $\bar{\kappa} \in \mathcal{T}_h^1$ that intersect κ . To identify these elements efficiently, an advancing front search is employed. The search is initialized by determining a seed element in the source mesh that contains the centroid of the target element. This is achieved by iterating through the source elements and applying a point-in-tetrahedron test based on the dot product between face normals and vectors connecting face vertices to the target centroid. Once the seed element is identified, neighboring elements are explored iteratively to detect all source elements intersecting κ .

Once an intersecting pair consisting of a target element $\kappa \in \mathcal{T}_h^2$ and a source element $\bar{\kappa} \in \mathcal{T}_h^1$ has been identified, the construction of the corresponding portion of the local supermesh follows the sequence of steps illustrated in Fig. 1. Starting from the intersecting tetrahedra in the physical space, both elements are mapped to a reference configuration, where the intersection points are identified. These points are then used to reconstruct the intersection polyhedron, which is subsequently subdivided into tetrahedral sub-elements suitable for numerical integration. Finally, the resulting tessellation is mapped back to the physical space. The individual steps are described below.

First, to simplify the geometric treatment and improve numerical robustness, both tetrahedra are transformed to the reference space associated with the target element $\kappa \in \mathcal{T}_h^2$. We consider the reference tetrahedron with vertices $(0, 0, 0)$, $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$. The mapping from a point $\mathbf{x}_{\text{ref}}$ in the reference space to a point $\mathbf{x}$ in the physical space is given by the affine transformation

$$\mathbf{x} = \mathbf{x}_0 + \mathbf{B}\mathbf{x}_{\text{ref}}, \quad (13)$$

where $\mathbf{x}_0 = (x_0, y_0, z_0)$ is one vertex of the target element κ , and

$$\mathbf{B} = \begin{bmatrix} x_1 - x_0 & x_2 - x_0 & x_3 - x_0 \\ y_1 - y_0 & y_2 - y_0 & y_3 - y_0 \\ z_1 - z_0 & z_2 - z_0 & z_3 - z_0 \end{bmatrix}, \quad (14)$$

with $\mathbf{x}_1, \mathbf{x}_2$, and $\mathbf{x}_3$ denoting the remaining vertices of κ . The inverse mapping

$$\mathbf{x}_{\text{ref}} = \mathbf{B}^{-1}(\mathbf{x} - \mathbf{x}_0) \quad (15)$$

is then used to transform the vertices of the source element $\bar{\kappa} \in \mathcal{T}_h^1$ into the reference space. In what follows, the reference image of the target element κ is denoted by $\mathcal{A}$, whereas the mapped source tetrahedron is denoted by $\mathcal{B}$.

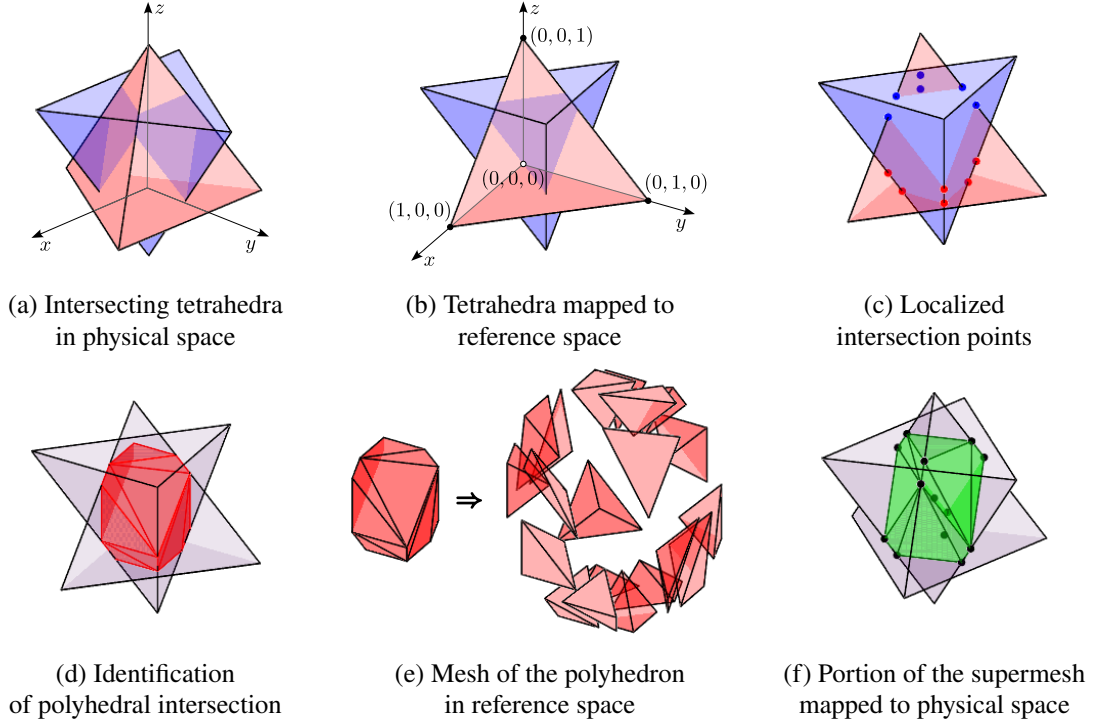


Figure 1: Steps for constructing the portion of the supermesh corresponding to the intersection of tetrahedra from the previous (blue) and current (red) mesh. First, the intersecting tetrahedra (a) are transformed into the reference space (b), where the intersection points are detected (c). Next, the convex hull of these points is constructed to recover the intersection polyhedron (d). This polyhedron is then subdivided into a set of tetrahedra (e), and the resulting tessellation is mapped back to the physical space (f).

To improve the robustness of the subsequent intersection procedure, a clipping correction is applied to the vertices of $\mathcal{B}$. More precisely, each vertex of $\mathcal{B}$ is successively checked against the faces, edges, and vertices of the reference tetrahedron $\mathcal{A}$ and, whenever its distance to one of these geometric entities falls below a prescribed tolerance (here 10^{-10}), it is snapped to the corresponding entity, see Fig. 2. Although this correction introduces only a negligible perturbation of the geometry, it significantly improves numerical robustness. In particular, it prevents the creation of near-degenerate intersection polyhedra, reduces the number of spurious small-volume sub-elements, and stabilizes the subsequent localization and reconstruction steps.

Once both tetrahedra have been represented in the reference space, the intersection localization algorithm is applied to determine the vertices of the polyhedron $\mathcal{A} \cap \mathcal{B}$. At this stage, only the set of intersection points is obtained, while the connectivity of these points within the polyhedral structure remains to be determined. The details of the localization procedure are presented in Sec. 3.2.

The next step is to reconstruct the intersection polyhedron from the localized points. Since the intersection of two tetrahedra is convex, this reconstruction can be formulated as a three-dimensional convex hull problem. The resulting polyhedral description provides the face structure and vertex connectivity of the intersection region. This part of the algorithm is discussed in Sec. 3.3.

After the intersection polyhedron has been reconstructed, it is subdivided into tetrahedral sub-elements in order to enable the evaluation of the Galerkin projection integrals by standard quadrature rules. This yields a tetrahedral tessellation of the region $\mathcal{A} \cap \mathcal{B}$ in the reference space, see Sec. 3.4. Finally, the resulting tetrahedral sub-elements are mapped back to the physical space by the affine transformation (13), thereby producing the contribution of the pair $(\bar{\kappa}, \kappa)$ to the local supermesh.

For a fixed target element $\kappa \in \mathcal{T}_h^2$, the above procedure is applied to all source elements $\bar{\kappa} \in \mathcal{T}_h^1$ identified as intersecting by the advancing front search. Repeating this construction for every target element of the new mesh yields the collection of local supers meshes required for the conservative transfer between successive meshes.

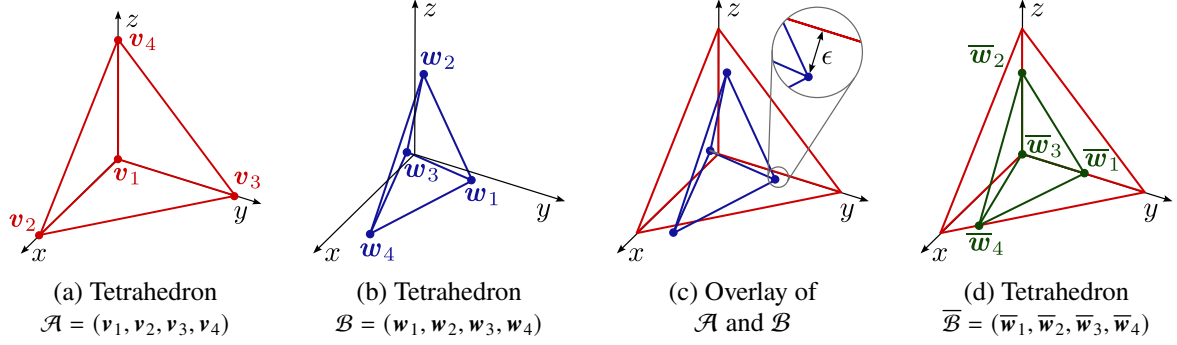


Figure 2: Illustration of the clipping correction applied to tetrahedron $\mathcal{B}$ after mapping into the reference space. (a) The reference tetrahedron $\mathcal{A}$ with vertices $v_1 = (0, 0, 0)$, $v_2 = (1, 0, 0)$, $v_3 = (0, 1, 0)$, and $v_4 = (0, 0, 1)$, representing the target element. (b) The image of source element $\bar{\kappa} \in \mathcal{T}_h^1$ mapped to the reference space, denoted as tetrahedron $\mathcal{B}$. (c) Overlay of tetrahedra $\mathcal{A}$ and $\mathcal{B}$. (d) Modified tetrahedron $\bar{\mathcal{B}}$ after applying geometric clipping. Assuming that vertices w_1, w_2, w_3 , and w_4 lie within distance $\epsilon < 10^{-10}$ from, respectively, edge (v_1, v_2) , edge (v_1, v_4) , vertex v_1 , and edge (v_2, v_3) , they are snapped to the corresponding geometric entities. This correction ensures numerical robustness in further geometric computations. Note that the distances and proportions are not drawn to scale.

3.2 Tetrahedra Intersection Points Localization

The intersection of two tetrahedra, if non-empty, is a convex polyhedron. The goal of the present step is therefore to localize all vertices of the intersection polyhedron $\mathcal{C} = \mathcal{A} \cap \mathcal{B}$, where $\mathcal{A}$ and $\mathcal{B}$ denote the two tetrahedra in the reference space introduced in Sec. 3.1. These vertices arise from a finite number of geometric configurations: a vertex of one tetrahedron may lie inside the other, an edge of one tetrahedron may intersect a face of the other, or, in degenerate cases, two edges may intersect. A few representative examples of tetrahedron–tetrahedron intersections are shown in Fig. 3, including several measure-zero degenerate configurations. Such cases must be handled robustly, since they still contribute valid vertices of the intersection polyhedron and may otherwise lead to inconsistencies in its subsequent reconstruction.

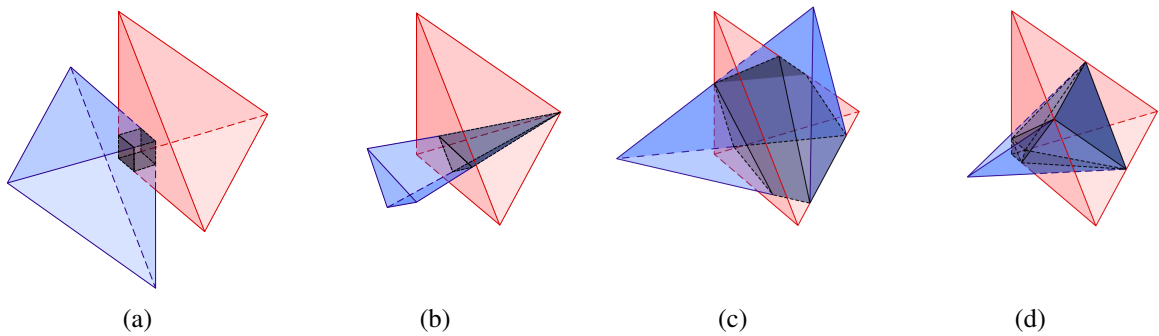


Figure 3: Examples of tetrahedron–tetrahedron intersections. (a) General intersection configuration where one vertex of a tetrahedron lies inside the other, and three of its edges intersect three corresponding faces of the second tetrahedron. Figs. (b)–(d) illustrate measure-zero degenerate cases. (b) Tetrahedra sharing a common vertex. (c) Partial overlap of two edges together with an additional edge–edge intersection point. (d) One face of a tetrahedron lying in the plane of a face of the other, with its vertices simultaneously lying on edges of the second tetrahedron.

General intersection algorithms for convex polyhedra are often formulated in terms of successive clipping by half-spaces defined by the element faces [20, 21]. This viewpoint is closely related to polygon clipping techniques known from computer graphics [22, 23], and recent tetrahedron–tetrahedron intersection approaches follow the same principle [18]. In the present work, however, all intersection points are localized explicitly before the intersection polyhedron is reconstructed. This makes it possible to treat both generic and degenerate configurations within a unified localization framework.

The localization procedure consists of the following steps. First, all vertices of $\mathcal{A}$ are tested for inclusion in tetrahedron $\mathcal{B}$. Next, all edges of $\mathcal{A}$ are tested against all faces of $\mathcal{B}$, followed by edge–edge intersection tests between $\mathcal{A}$ and $\mathcal{B}$. The same sequence is then applied in the opposite direction, by testing all edges of $\mathcal{B}$ against all faces of $\mathcal{A}$ and, finally, all vertices of $\mathcal{B}$ for inclusion in tetrahedron $\mathcal{A}$.

The vertex-in-tetrahedron test follows the same face-normal criterion used in the seed-element search. For each face of the tetrahedron, the dot product between the outward unit normal vector and the vector connecting a face vertex to the tested point is evaluated. A point is classified as interior if all such tests are non-positive up to a prescribed tolerance. In the present work, a tolerance of 10^{-13} is used, which also allows points lying on the surface of the tetrahedron to be consistently accepted as valid intersection points. This includes degenerate situations such as shared vertices or vertices lying on edges or faces of the opposite tetrahedron.

The edge–face intersections are detected using a standard segment–triangle intersection procedure. In particular, algorithms of Badouel [24] and Möller–Trumbore [25] provide efficient and robust tests for this purpose. In the present implementation, the Möller–Trumbore algorithm is employed because it avoids explicit construction of the triangle plane equation and leads to a simple and efficient evaluation of segment–triangle intersections in three dimensions.

Although many generic intersection configurations are captured by the combination of the vertex-in-tetrahedron and edge–face tests, this is not sufficient for a robust treatment of boundary and degenerate cases. In particular, configurations in which an edge intersects a triangular face exactly on one of its edges or at one of its vertices are detected more robustly by an additional edge–edge intersection test. Similar situations are sometimes handled in the literature by symbolic perturbation [26]. Here they are resolved directly by explicit edge–edge testing together with tolerance-based point classification.

Since individual localization tests may identify the same geometric point multiple times, the final set of intersection points is filtered to remove duplicates. The resulting unique point set forms the input for the convex-hull reconstruction described in the following subsection.

3.3 Intersection Polyhedron Reconstruction

After the localization step described in the previous subsection, the vertices of the intersection polyhedron $\mathcal{C} = \mathcal{A} \cap \mathcal{B}$ are known, but their connectivity remains undetermined. The objective of the present step is therefore to reconstruct the polyhedral boundary, i.e., to identify the faces and their vertex connectivity.

Unlike the two-dimensional case, where the connectivity of the intersection polygon can be recovered by a simple cyclic ordering of points, the three-dimensional setting requires an explicit reconstruction of the polyhedral surface.

Since the intersection of two tetrahedra is convex, this reconstruction can be formulated as a three-dimensional convex hull problem. Depending on the relative configuration of the tetrahedra, the resulting intersection polyhedron may vary from a tetrahedron to more complex convex shapes with different numbers of vertices and faces. In particular, the number of vertices may be as high as 12 [27]. The convex hull of the localized intersection points then provides the required face structure of the polyhedron.

A wide range of convex hull algorithms has been developed in computational geometry [28]. In the present work, the Quickhull algorithm [29] is employed because it offers a suitable balance between efficiency and robustness for the relatively small point sets arising from tetrahedron–tetrahedron intersections. Here,

the main difficulty is not the number of points itself, but the reliable treatment of degenerate or nearly degenerate configurations, such as coplanar, collinear, or nearly coincident points.

For this reason, a freely available robust implementation of Quickhull [30] is used. This implementation is designed to handle pathological geometric configurations that frequently arise in practice, especially when intersecting elements lie close to the computational boundary or nearly coincide after clipping. The resulting convex hull provides the face connectivity of the intersection polyhedron and serves as the input for the tetrahedral subdivision described in the following subsection.

3.4 Subdivision into Tetrahedral Sub-Elements

To facilitate numerical integration, the reconstructed intersection polyhedron must be subdivided into simplices, i.e., tetrahedra. A direct tetrahedralization of a general polyhedron would typically require additional topological information in order to construct a consistent volumetric decomposition. In the present work, however, this step is simplified by exploiting the output of the convex hull algorithm.

Although the geometric faces of the intersection polyhedron may in general be polygonal, the employed convex hull routine returns a triangulated boundary representation. Hence, a conforming surface triangulation of the intersection polyhedron is already available. To construct a volumetric mesh, the centroid of the polyhedron is introduced as an interior vertex of the tessellation. Since the intersection polyhedron is convex, this point lies inside the polyhedron, and each surface triangle can therefore be connected to it to form a tetrahedral sub-element. In this way, a conforming tetrahedral subdivision of the intersection polyhedron is obtained. The procedure is illustrated in Fig. 4 for the case where the intersection polyhedron is a cube.

This construction does not, in general, minimize the number of tetrahedral sub-elements, since the introduction of an additional interior point increases their total number. Consequently, numerical integration is performed over more elements than would be strictly necessary. Nevertheless, the method is simple to implement, robust with respect to varying polyhedral topologies, and fully adequate for the present purpose, where the resulting tetrahedra serve only as integration subdomains in the conservative transfer procedure.

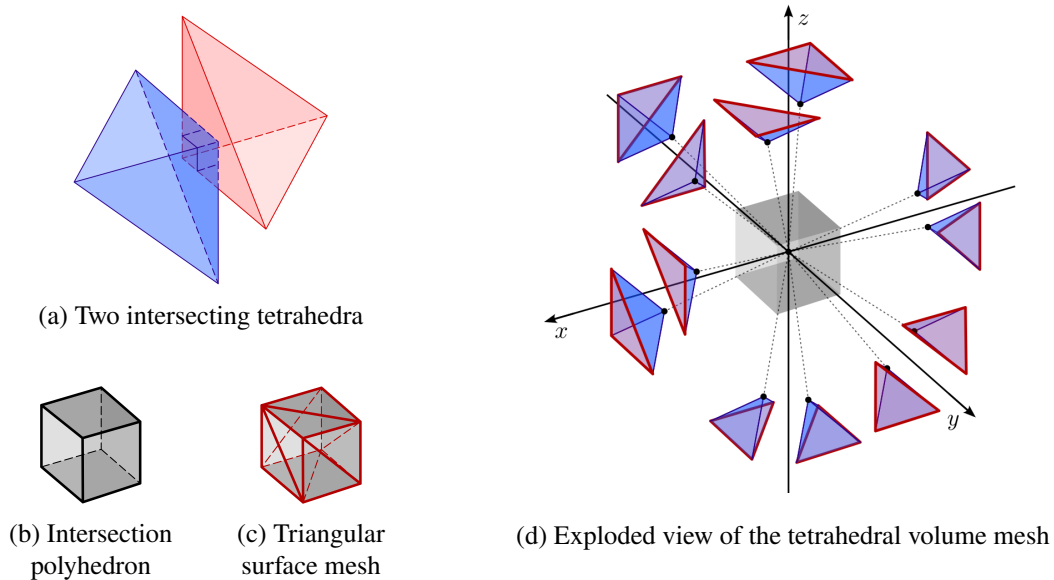


Figure 4: Tessellation of the intersection polyhedron. (a) Two tetrahedra whose intersection forms a cube. (b) The resulting intersection polyhedron. (c) Triangulated surface representation obtained from the convex hull construction. (d) Tetrahedral subdivision obtained by connecting each surface triangle to the centroid of the polyhedron.

4. Numerical Results

The proposed three-dimensional solution transfer is assessed in two complementary studies. First, the transfer is verified independently of any governing equation by repeated projection of analytical functions between non-matching tetrahedral meshes over the domain $\Omega = [-1, 1]^3$. Second, the method is embedded into a time-accurate hybridized discontinuous Galerkin (HDG) framework with anisotropic mesh adaptation [16] and applied to a model convection-diffusion problem in order to evaluate its behavior under successive mesh adaptations.

4.1 Verification of the Transfer Procedure

The transfer procedure is first verified independently of any governing equation by repeated projection of analytical functions between non-matching tetrahedral meshes. Two representative functions are considered,

$$w_1 = \sin(\pi x) \sin(\pi y) \sin(\pi z), \quad (16)$$

$$w_2 = \frac{10x}{\sqrt{1 + (10x)^2}} + \frac{10y}{\sqrt{1 + (10y)^2}} + \frac{10z}{\sqrt{1 + (10z)^2}}. \quad (17)$$

Here, w_1 is a smooth trigonometric function, whereas w_2 exhibits strong localized gradients corresponding to a smooth step-like profile.

Each function is projected onto a structured source mesh and subsequently transferred to a non-matching unstructured target mesh. This two-step process is repeated ten times in order to amplify the accumulated projection error. The considered mesh pairs differ in topology but share the same characteristic element size, with $h = \{0.5, 0.25, 0.125, 0.0625\}$.

Figs. 5(a) and 5(b) show the L^2 -error after the final transfer. Optimal convergence rates of $\mathcal{O}(h^{p+1})$ are obtained for $p = 1, \dots, 4$, confirming the high-order accuracy of the three-dimensional supermesh-based Galerkin projection. A slightly slower convergence rate is observed for w_2 , since reaching the full asymptotic regime for this smooth approximation of a discontinuous profile would require further mesh refinement.

In addition to accuracy, conservation is examined by monitoring the integral of the transferred quantity over the computational domain. Since the exact integral of both test functions is zero, the numerical results directly reflect the conservative character of the transfer. The integral is preserved up to numerical precision across all refinement levels and polynomial orders.

4.2 Application in a Time-Accurate Adaptive HDG Framework

To assess the behavior of the proposed solution transfer procedure within a time-dependent adaptive framework based on the HDG spatial discretization, we consider a three-dimensional linear convection-diffusion problem with a translating Gaussian profile,

$$\frac{\partial w}{\partial t} + \mathbf{c} \cdot \nabla w = \varepsilon \nabla^2 w, \quad \mathbf{x} \in \Omega, \quad t \geq 0, \quad (18)$$

where $\mathbf{c} \in \mathbb{R}^3$ and $w \equiv w(x, y, z, t)$. The computational domain is $\Omega = [-1, 1]^3$. The Gaussian pulse is initially centered at $(x_c, y_c, z_c) = (-0.3, -0.3, -0.3)$ and translates diagonally across the domain up to the final time $T = \pi/4$.

The exact solution is given by

$$w(x, y, z, t) = \left(\frac{2\sigma^2}{2\sigma^2 + 4\varepsilon t} \right)^{3/2} \exp \left(-\frac{(\hat{x} - x_c)^2 + (\hat{y} - y_c)^2 + (\hat{z} - z_c)^2}{2\sigma^2 + 4\varepsilon t} \right), \quad (19)$$

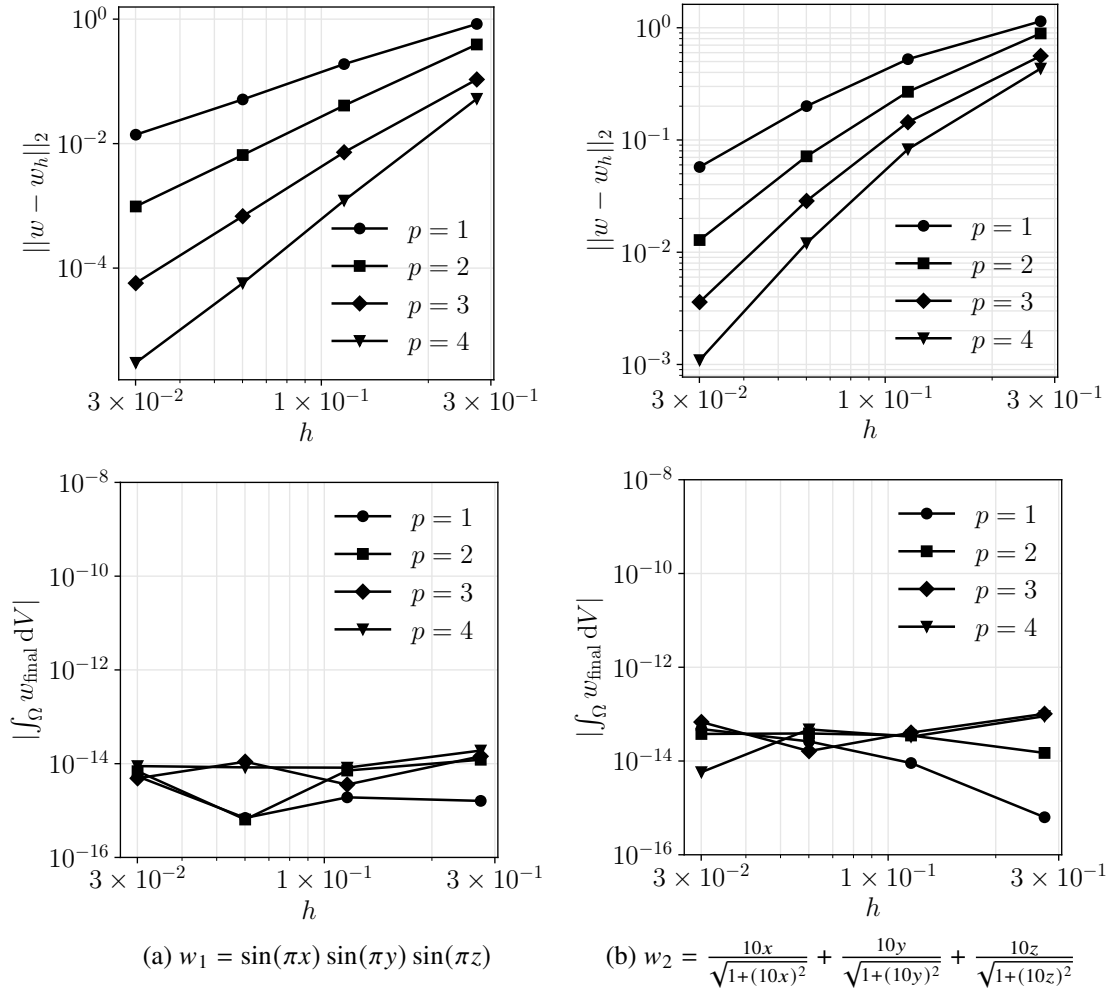


Figure 5: 3D convergence results in terms of the L^2 -norm error (top) and the integral of the function representation over the domain Ω (bottom) for polynomial orders $p = 1, \dots, 4$

where

$$\hat{x} = x - c_x t, \quad \hat{y} = y - c_y t, \quad \hat{z} = z - c_z t, \quad (20)$$

with $\sigma = 0.1$, $\varepsilon = 0.001$, and advection velocity vector $\mathbf{c} = (c_x, c_y, c_z) = \frac{4}{\pi} (0.6, 0.6, 0.6)$. With this choice, the pulse center reaches $(0.3, 0.3, 0.3)$ at the final time T . Dirichlet boundary conditions are prescribed from the exact solution.

Convergence studies are performed on both static and adaptively refined meshes for polynomial degrees $p = 2$ and $p = 3$. In the static case, uniform unstructured meshes with $N_e = \{500, 1\,000, 2\,500, 5\,000, 11\,000\}$ elements are used throughout the simulation. The same values are used as maximum element counts in the adaptive runs. The anisotropic mesh adaptation framework follows [3], and its three-dimensional extension is described in [31]. The adaptation itself is driven by the metric-based approach using the Refine mesh generator [32, 33].

The adaptive simulation starts from an unstructured mesh with 2 000 elements, onto which the initial condition is projected. Two initial mesh adaptations are performed before time integration begins. In the adaptive setting, the mesh is modified repeatedly during the simulation in order to follow the translating and diffusing solution profile. Each such mesh update requires transferring the numerical solution from the previous mesh to the new one, making this test case a suitable setting for assessing the repeated use of the proposed conservative transfer procedure within a time-dependent computation.

Time integration is carried out by a third-order three-stage diagonally implicit Runge–Kutta method. Time-accurate mesh adaptation is performed using the Backward Euler mesh predictor strategy introduced in our previous work [16], with the mesh updated at every second time step. The time step size is fixed for all simulations, and the total number of time steps is chosen as $N_e/2$. This setting therefore provides a direct test of the coupling between time integration, anisotropic mesh adaptation, and conservative solution transfer in three dimensions.

Fig. 6 shows the convergence behavior in terms of the L^2 -error at the final time. The adaptive simulations exhibit convergence rates ranging from $O(h^{p+1})$ to $O(h^{p+2})$, confirming that successive transfers do not degrade the formal accuracy of the high-order discretization. In contrast, the static meshes display delayed asymptotic convergence due to insufficient resolution of the localized Gaussian peak.

Snapshots of the numerical solution and the corresponding anisotropically adapted meshes are shown in Fig. 7. The mesh evolution follows the translation and spreading of the Gaussian profile, illustrating the stable coupling of time integration, anisotropic mesh adaptation, and conservative three-dimensional solution transfer.

The adaptive simulations show that the proposed transfer can be repeatedly applied within the time-dependent adaptive loop without compromising the accurate tracking of the translating and diffusing Gaussian profile.

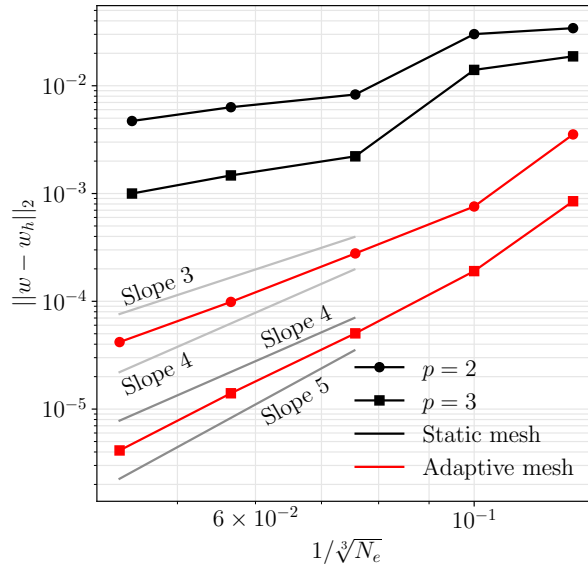


Figure 6: Convergence study for the 3D translating Gaussian test case with static and adaptively refined meshes in terms of the L^2 -error of the numerical solution for $p = 2$ and $p = 3$.

5. Discussion and Conclusions

This work presents a conservative high-order solution transfer procedure for three-dimensional tetrahedral meshes. The method is formulated as a local Galerkin projection between non-matching meshes and is evaluated over a supermesh constructed from tetrahedron–tetrahedron intersections. In comparison with the two-dimensional setting, the main new difficulty is not the algebraic form of the transfer itself, but the geometric treatment of the intersection regions, which form convex polyhedra of varying topology. For this reason, particular emphasis is placed on the robustness of the underlying geometric procedure, namely the localization of intersection points, reconstruction of the intersection polyhedron, and its subdivision into tetrahedral integration sub-elements, which together define the construction of the local supermesh.

The proposed methodology is assessed in two complementary numerical studies. First, independent transfer tests confirm optimal convergence rates of order $O(h^{p+1})$ and preservation of integral quantities

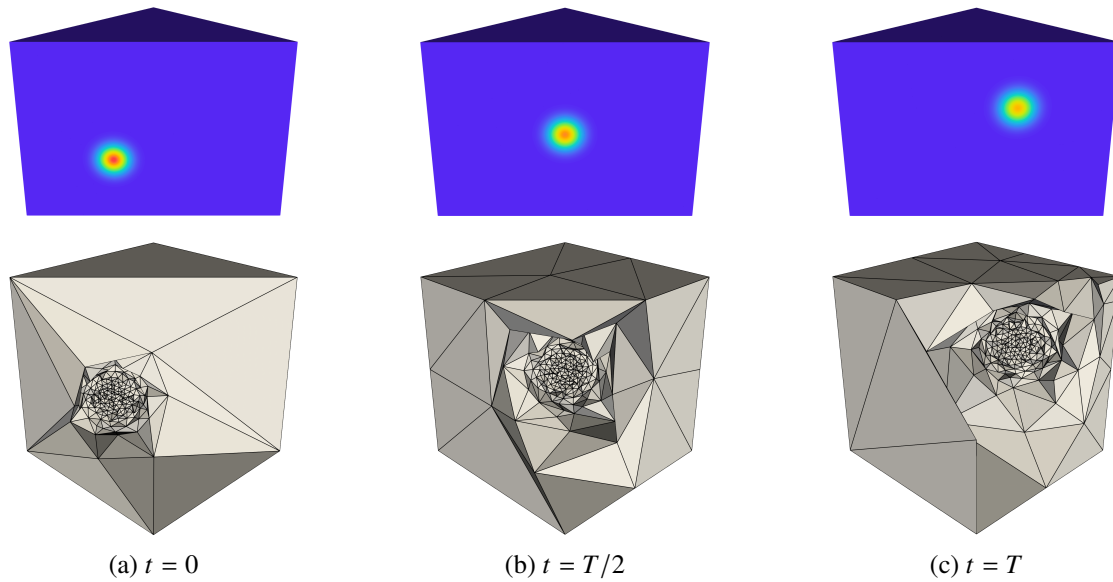


Figure 7: Numerical solution and the corresponding anisotropically adapted meshes for the 3D translating Gaussian test case with $N_e = 5\,000$ and $p = 2$ at times $t = 0$ (after the second initial adaptation), $t = T/2$, and $t = T$.

up to numerical precision, demonstrating that the transfer retains both high-order accuracy and conservation. Second, when embedded into a time-accurate adaptive framework based on the HDG spatial discretization, the transfer remains stable under repeated application during successive mesh adaptations. In the presented three-dimensional convection–diffusion test case, the adaptive computations preserve the formal accuracy of the underlying high-order discretization and successfully track a translating and diffusing localized solution profile. These results indicate that the proposed transfer provides a reliable mechanism for repeated solution transfer in fully three-dimensional adaptive simulations.

Although a detailed complexity analysis is beyond the scope of the present work, the extension from two to three dimensions introduces a noticeable increase in computational cost. This is mainly due to the greater geometric complexity of tetrahedron–tetrahedron intersections, the construction of the local supermesh, and the subsequent integration over tetrahedral supermesh elements. In practice, the number of local supermesh elements associated with a target element is significantly higher than in the two-dimensional setting and depends strongly on the topology of the intersecting meshes. Nevertheless, the overall transfer procedure retains the same conceptual structure as in two dimensions and remains applicable in the three-dimensional setting.

The emphasis of the present work is on the methodological aspects of conservative solution transfer in three dimensions. Within the time-accurate adaptive framework, the three-dimensional assessment is restricted to a scalar linear convection–diffusion problem. Its purpose is to verify the coupling between time integration, anisotropic mesh adaptation, and conservative solution transfer in a fully three-dimensional setting. The convection–diffusion test case is therefore not intended as a demonstration of full three-dimensional CFD capability, but rather as a controlled setting for assessing repeated conservative transfer in an adaptive time-dependent loop. While the proposed transfer itself is formulated for general piecewise polynomial solution representations, more complex three-dimensional systems of partial differential equations are not considered here. In particular, applications to the Euler or Navier–Stokes equations, together with a broader study of computational efficiency and parallel performance, are left for future work.

Acknowledgements

This study was supported by Institutional support for the long-term conceptual development of the University of West Bohemia in Pilsen.

References

- [1] Ngoc Cuong Nguyen, Jaime Peraire, and Bernardo Cockburn. An implicit high-order hybridizable discontinuous Galerkin method for nonlinear convection-diffusion equations. *Journal of Computational Physics*, 228(23):8841–8855, 2009.
- [2] Tomáš Levý and Georg May. Comparison of implicit time-discretization schemes for hybridized discontinuous Galerkin methods. *Applied and Computational Mechanics*, 16(2), 12 2022.
- [3] Ajay Mandyam Rangarajan, Aravind Balan, and Georg May. Mesh Adaptation and Optimization for Discontinuous Galerkin Methods Using a Continuous Mesh Model. In *AIAA Modeling and Simulation Technologies Conference*, 2016.
- [4] John K. Dukowicz and John W. Kodis. Accurate conservative remapping (rezoning) for arbitrary Lagrangian-Eulerian computations. *SIAM Journal on Scientific & Statistical Computing*, 8:3, 05 1987.
- [5] Jeffrey Grandy. Conservative remapping and region overlays by intersecting arbitrary polyhedra. *Journal of Computational Physics*, 148(2):433–466, 1999.
- [6] Rao Garimella, Milan Kucharik, and Mikhail Shashkov. An efficient linearity and bound preserving conservative interpolation (remapping) on polyhedral meshes. *Computers & Fluids*, 36(2):224–237, 2007.
- [7] Frédéric Alauzet and Michel Mehrenberger. P1-conservative solution interpolation on unstructured triangular meshes. *International Journal for Numerical Methods in Engineering*, 84(13):1552–1588, 2010.
- [8] Jean-François Remacle, Xiangrong Li, Mark S. Shephard, and Joseph E. Flaherty. Anisotropic adaptive simulation of transient flows using discontinuous Galerkin methods. *International Journal for Numerical Methods in Engineering*, 62(7):899–923, 2005.
- [9] Frédéric Alauzet, Adrien Loseille, and Géraldine Olivier. Time-accurate multi-scale anisotropic mesh adaptation for unsteady flows in CFD. *Journal of Computational Physics*, 373:28–63, 2018.
- [10] Frédéric Alauzet, Pascal J. Frey, Paul L. George, and Bijan Mohammadi. 3D transient fixed point mesh adaptation for time-dependent problems: Application to CFD simulations. *Journal of Computational Physics*, 222(2):592–623, 2007.
- [11] Frédéric Alauzet. A parallel matrix-free conservative solution interpolation on unstructured tetrahedral meshes. *Computer Methods in Applied Mechanics and Engineering*, 299:116–142, 2016.
- [12] Patrick E. Farrell, Matthew D. Piggott, Christopher C. Pain, Gerard J. Gorman, and Clark R. Wilson. Conservative interpolation between unstructured meshes via supermesh construction. *Computer Methods in Applied Mechanics and Engineering*, 198(33):2632–2642, 2009.
- [13] Patrick E. Farrell and James R. Maddison. Conservative interpolation between volume meshes by local Galerkin projection. *Computer Methods in Applied Mechanics and Engineering*, 200:89–100, 2011.
- [14] Tzanio Kolev and Will Pazner. Conservative and Accurate Solution Transfer Between High-Order and Low-Order Refined Finite Element Spaces. *SIAM Journal on Scientific Computing*, 44(1):A1–A27, 2022.
- [15] Tomáš Levý and Georg May. Conservative solution transfer between anisotropic meshes for time-accurate hybridized discontinuous Galerkin methods. *International Journal for Numerical Methods in Fluids*, 96, 02 2024.
- [16] Tomáš Levý and Georg May. Towards a Robust Time-Accurate Anisotropically Adaptive Hybridized Discontinuous Galerkin Method. *Computers & Fluids*, 301:106792, 10 2025.
- [17] Robert Chiodi and Olivier Desjardins. General, Robust, and Efficient Polyhedron Intersection in

- the Interface Reconstruction Library. *Journal of Computational Physics*, 449:110787, 2022.
- [18] Conor McCoid and Martin J. Gander. Intersection of Tetrahedra. *ACM Transactions on Mathematical Software*, 51(2), 2025.
 - [19] Moshe Dubiner. Spectral methods on triangles and other domains. *Journal of Scientific Computing*, 6:345–390, 1991.
 - [20] Herbert Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *Monographs in Theoretical Computer Science. An EATCS Series*. Springer-Verlag, Berlin, Heidelberg, 1987.
 - [21] Mike Cyrus and Jay Beck. Generalized two- and three-dimensional clipping. *Computers & Graphics*, 3(1):23–28, 1978.
 - [22] Kevin Weiler and Peter Atherton. Hidden surface removal using polygon area sorting. *ACM SIGGRAPH Computer Graphics*, 11(2):214–222, 1977.
 - [23] Günther Greiner and Kai Hormann. Efficient clipping of arbitrary polygons. *ACM Transactions on Graphics (TOG)*, 17(2):71–83, 1998.
 - [24] Didier Badouel. *An efficient ray-polygon intersection*, pages 390–393. 1990.
 - [25] Tomas Möller and Ben Trumbore. Fast, minimum storage ray/triangle intersection. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, page 7–es, 2005.
 - [26] Herbert Edelsbrunner and Ernst P. Mücke. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Transactions on Graphics*, 9(1):66–104, 1990.
 - [27] Franco P. Preparata and Michael Ian Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, NY, 1985.
 - [28] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition, 2008.
 - [29] C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa. The Quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483, 1996.
 - [30] Leo McCormack. *convhull_3d: A header-only C implementation of the Quickhull algorithm for building N-dimensional Convex Hulls and Delaunay meshes*, 2021.
 - [31] Ajay Mandyam Rangarajan, Ankit Chakraborty, Georg May, and Vít Dolejší. A Continuous–Mesh Optimization Technique for Piecewise Polynomial Approximation on Tetrahedral Grids. In *AIAA Scitech 2018 Forum*. American Institute of Aeronautics and Astronautics, 2018.
 - [32] Michael A. Park, Nicolas Barral, Daniel Ibanez, Dmitry S. Kamenetskiy, Joshua A. Krakos, Todd R. Michal, and Adrien Loseille. Unstructured Grid Adaptation and Solver Technology for Turbulent Flows. In *2018 AIAA Aerospace Sciences Meeting*, 2018.
 - [33] NASA. Refine: Mesh Generator and Adaptation Tool, 2023.