Japan Journal of Industrial and Applied Mathematics (2023) 40:1053–1082
https://doi.org/10.1007/s13160-023-00574-x

ORIGINAL PAPER

Second-order accurate finite volume method for *G*-equation on polyhedral meshes

Jooyoung Hahn[1] · Karol Mikula[1] · Peter Frolkovič[1] · Peter Priesching[2] · Martin Balažovjech[1] · Branislav Basara[2]

Received: 30 June 2022 / Revised: 19 December 2022 / Accepted: 4 March 2023 /
Published online: 28 March 2023

Abstract

In this paper, we propose a cell-centered finite volume method to numerically solve the *G*-equation on polyhedral meshes in three-dimensional space, that is, a general type of the level-set equation including advective, normal, and mean curvature flow motions. The main contribution is to design a numerical algorithm for the regularized mean curvature flow equation that can be consistently combined regarding the size of the time step with previous algorithms for the advective and normal flows on polyhedral meshes. For a spatial discretization, we use a flux-balanced approximation with an orthogonal splitting of displacement vector from a center of the cell to a center of the face. For a temporal discretization, we use a nonlinear Crank–Nicolson method with a deferred correction method which gives us, firstly, second-order accuracy in space and time similarly to the algorithms for the advective and normal flow equations, and, secondly, a possibility of straightforward domain decomposition for efficient parallel

These Jooyoung Hahn, Karol Mikula, Peter Frolkovič authors contributed equally to most of work.

✉ Jooyoung Hahn
jooyoung.hahn@stuba.sk

Karol Mikula
karol.mikula@stuba.sk

Peter Frolkovič
peter.frolkovic@stuba.sk

Peter Priesching
peter.priesching@avl.com

Martin Balažovjech
balazovjech@math.sk

Branislav Basara
branislav.basara@avl.com

[1] Department of Mathematics and Descriptive Geometry, Slovak University of Technology, Radlinského 11, 810 05 Bratislava, Slovak Republic

[2] Adavanced Simulation Technologies, AVL LIST GmbH, Hans-List-Platz 1, 8020 Graz, Austria

computation. Numerical experiments quantitatively show that the size of time step proportional to an average size of computational cells is enough to obtain the second-order convergence in space and time for smooth solutions of the general level set equation. A qualitative comparison is presented for a nontrivial example to compare numerical results obtained with hexahedral and polyhedral meshes. Finally, an example of solving numerically the G-equation used in combustion literature is given.

Keywords G-equation · Level-set equations · Mean curvature flow · Polyhedral meshes · Nonlinear Crank–Nicolson method · Flux-balanced approximation · Second-order experimental order of convergence

Mathematics Subject Classification 65M08 · 65Y05 · 80A25

1 Introduction

In this paper, we propose a cell-centered finite volume method to numerically solve the G-equation with a given burning velocity on polyhedral meshes in three-dimensional space. The G-equation in the combustion literature is firstly introduced in [1] to explain a propagation of thin flame surface structure in a premixed turbulent combustion. The form of equation is based on a level set approach, that is, the flame structure is described by a zero level set of an implicit function G. Numerous and diverse formulations of G-equation has been extensively studied in order to design correct models of burning velocities in the combustion community; see more details in [2, 3] and the references therein. Here, the well-established and standard form of G-equation based on Favre mean $\tilde{G}$ of G function is used for the corrugated flamelets regime and the thin reaction zones regimes in premixed turbulent combustion:

$$\bar{\rho}\frac{\partial \tilde{G}}{\partial t} + \bar{\rho}\tilde{\mathbf{v}} \cdot \nabla \tilde{G} = \bar{\rho}_u s_T^0 |\nabla \tilde{G}| - \bar{\rho} D_t \tilde{\kappa} |\nabla \tilde{G}|, \tag{1}$$

where $\bar{\rho}$ is the Reynolds mean density, $\bar{\rho}_u$ is the Reynolds mean unburnt density, $\tilde{\mathbf{v}}$ is the Favre mean velocity, s_T^0 is the turbulent burning velocity, D_t is the turbulent diffusivity, and $\tilde{\kappa}$ is the mean curvature of the flame surface. Note that the zero level set of $\tilde{G}$ is the Favre mean turbulent flame surface; see also a practical application in smoldering front propagation [4, 5].

In order to clearly explain the numerical algorithm to solve the G-equation, we rewrite the Eq. (1) as a general type of the level-set equation with simple coefficients in a bounded Lipschitz domain $\Omega \subset \mathbb{R}^3$:

$$\frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi + \delta |\nabla \phi| = \gamma |\nabla \phi|_\epsilon \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|_\epsilon} \right), \quad t \in [0, T], \tag{2}$$

where $|\nabla \phi|_\epsilon \equiv (\epsilon^2 + |\nabla \phi|^2)^{1/2}$, a regularization parameter $\epsilon > 0$ is a small constant [6], and functions $\mathbf{v}$, δ and γ are given and continuous on Ω with $\gamma > 0$. An initial condition and Dirichlet boundary condition are given until the final time $T > 0$ in the

form

$$\begin{aligned} \phi(\mathbf{x}, 0) &= \phi_0(\mathbf{x}), \quad \mathbf{x} \in \Omega, \\ \phi(\mathbf{x}, t) &= \phi_b(\mathbf{x}, t), \quad (\mathbf{x}, t) \in \partial\Omega \times (0, T]. \end{aligned} \tag{3}$$

Three velocities in (2) generate distinctive types of the evolution of a level surface of initial function ϕ_0. The second term in the left-hand side (LHS) of (2) with the spatially variable velocity $\mathbf{v}$ describes an advective flow motion of level surfaces. The third term in LHS controls a propagation of level surfaces in surface normal directions with the speed δ. The right-hand side (RHS) with $\epsilon = 0$ and the constant $\gamma > 0$ generates a mean curvature flow (MCF) motion of the level surfaces. In numerous applications of level-set equation (2), an interface is usually represented by the zero level set of initial signed distance function ϕ_0. Several numerical algorithms and diverse applications for level set methods are presented in [7–9] and the references therein.

The first contribution of this paper is to propose a novel cell-centered finite volume method to discretize the regularized mean curvature term of the general level set equation (2) on polyhedral meshes in three-dimensional (3D) space. A polyhedral mesh means a collection of star-shaped convex polyhedral cells, which presents a spatial discretization of a given computational domain Ω in 3D. A usage of polyhedral meshes in an industrial computer-aided engineering (CAE) has been steadily increased for the last decade because of its shape flexibility to discretize highly complicated computational domains; see more details in [10]. Moreover, the faces of a polyhedron cell are distributed more isotropically than the ones in a hexahedron cell, which makes numerical errors in a polyhedral mesh less directionally dependent than in a hexahedral mesh. The second contribution is to combine the proposed discretization with the previous algorithms to discretize normal and advective terms of the general level set equation in order to numerically achieve the second-order convergence in space and time. At the end, we apply the combined algorithm to solve the *G*-equation and check whether it is reasonable in real application.

A classical approach to solve PDEs numerically on polyhedral meshes is to use finite volume methods (FVMs) [11]. Furthermore, advanced numerical methods on polyhedral meshes such as mimetic finite-difference methods [12], virtual element methods [13], and discrete duality finite volume method [14] bring promising results. We prefer the cell-centered FVMs to aforementioned methods that seem more suitable in industrial CAE applications because of less memory requirements and straightforward implementation of efficient parallel computation. We use the degrees of freedom only at the centers of cells for the memory efficiency and a 1-ring face neighborhood structure for a rather straightforward domain decomposition with parallel computations.

The numerical algorithms for advective or normal flow equations of (2) on unstructured meshes have been studied by several authors [15–17]. A fully explicit time discretization with a vertex-centered finite volume method is used in [18]. The second-order total variation diminishing explicit Runge–Kutta method with a cell-centered finite volume method and inflow based gradient is developed in [19]. To avoid a stability restriction of the size of time step, an inflow-implicit outflow-explicit (IIOE) method is suggested in [20–23]. The method is extended in [24, 25] for polyhedral

meshes with a 1-ring face neighborhood structure to enable efficient parallel computation.

The numerical algorithms for the regularized mean curvature flow equation have been studied on hexahedral or tetrahedral meshes. Finite element methods were developed by, e.g., Deckelnick, Dziuk, and Elliot, see [26] and references therein. A complementary volume (CV) method is used in [27] with a fully-implicit time discretization. A semi-implicit CV scheme is suggested in [28] to have better efficiency, while the theoretical properties from [27] are preserved. The semi-implicit CV method has been widely applied in 2D and 3D image segmentation problems [29–31]. Its theoretical properties when using the regularization of gradient norm in the sense of [6] are proved. Namely, the L^∞ stability of solution and the L^1 stability of solution gradient are given in [28, 30], and a consistency of the scheme is shown in [32]. In [33, 34], the absolute value of gradient is computed by the Green–Gauss theorem and for orthogonal meshes the convergence of a numerical solution to the weak solution of the regularized mean curvature flow equation was proved. In cases of a complicated shape of domain in 3D, the mesh orthogonality is too strong restriction to practically generate a polyhedral mesh. The method for solving the regularized mean curvature flow equation proposed in this paper extends the theoretically approved method from [33] to the case of general non-orthogonal polyhedral meshes.

To propose a spatial discretization on a non-orthogonal polyhedral mesh, the authors use a weighted least-squares minimization in [35, 36] to explicitly reconstruct a gradient at a center of polyhedron cell. In [36], the over-relaxed correction method [37] is applied to discretize the regularized mean curvature term. In this paper, inspired by [33, 38], we propose a flux-balanced spatial discretization on non-orthogonal polyhedral meshes. The proposed approach is based on a flux-balanced approximation, and it uses an orthogonal splitting of displacement vector from a center of cell to a center of face.

A temporal discretization should be properly considered to complete a numerical algorithm for solving (2). In the case of regularized mean curvature flow, the semi-implicit methods in [28–34] needs to solve a linear system of algebraic equations in each time step. However, a size of time step should be proportional to a square of the average size of cells in order to obtain the second-order experimental order of convergence. It is not practical to use such a size of time step when a semi-implicit method for the regularized mean curvature flow equation is combined with numerical methods for the advective or normal flow equations [19–23, 25], which only need a size of time step proportional to an average size of cells in order to obtain the experimental order of convergence (EOC) approximately close to 2. The discrepancy of the size of time step can be resolved by using a nonlinear Crank–Nicolson method introduced to solve a plane curve shortening flow in [39]. In the case of level-set equations, the nonlinear Crank–Nicolson method is applied to obtain $EOC \simeq 2$ when a size of time step is proportional to an average size of cells in [40]. For a practical efficiency of parallel computation, we propose to combine the nonlinear Crank–Nicolson method with a deferred correction method [41] in order to deal with a restriction of 1-ring face neighborhood structure in a decomposed computational domain.

The rest of paper is organized as follows. A spatial and temporal discretization for the regularized mean curvature flow equation is proposed in Sects. 2 and 3, respectively.

We briefly review the iterative IIOE method to solve the advective and normal flow equation in Section A and then present the final proposed algorithm to solve (2). Several numerical experiments are presented to quantitatively observe a behavior of proposed numerical algorithm and to qualitatively compare numerical solutions in Sect. 4. Finally, we conclude in Sect. 5.

2 Flux-balanced approximation

In this section, we propose a numerical scheme to discretize a regularized mean curvature flow equation on a bounded Lipschitz domain $\Omega \subset \mathbb{R}^3$:

$$\frac{\partial \phi}{\partial t} = |\nabla \phi|_\epsilon \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|_\epsilon} \right), \quad t \in [0, T], \tag{4}$$

where the regularization parameter ϵ in (4) is chosen to satisfy the property to be a small constant that is, typically, proportional to an average size of cells in numerical computations. Note that we explain the proposed algorithm for the mean curvature flow equation with $\gamma = 1$ in (2) and the algorithm is straightforwardly applied to solve the general case, for example, $\gamma = \gamma(\mathbf{x})$ in (1). The computational domain Ω is discretized by open non-overlapping polyhedral cells $\Omega_p \subset \Omega$ with non-zero volume $|\Omega_p| \neq 0$, that is, $\bar{\Omega} = \bigcup_{p \in \mathcal{I}} \bar{\Omega}_p$, where $\mathcal{I}$ is the set of cell indices. For a fixed $p \in \mathcal{I}$, we denote the set $\mathcal{N}_p$ as indices of neighbor cells where the cells Ω_q, $q \in \mathcal{N}_p$ have a non-zero two-dimensional (2D) area intersection with Ω_p. An internal face is denoted by $e_f \subset \partial\Omega_q \cap \partial\Omega_p$ and has a non-zero 2D area $|e_f| \neq 0$. The index set of all internal faces is denoted by $\mathcal{F}$. We similarly define the set $\mathcal{B}$ as the index set of all boundary faces $e_b \subset \partial\Omega_p \cap \partial\Omega$ for $p \in \mathcal{I}$ with a non-zero 2D area $|e_b| \neq 0$. The indices of faces in a cell Ω_p, $p \in \mathcal{I}$ are split into two disjoint sets; the indices of internal faces $\mathcal{F}_p$ and the indices of boundary faces $\mathcal{B}_p$. The subscript $f \in \mathcal{F}_p$ is used as the index of internal face e_f and the subscript $b \in \mathcal{B}_p$ as the index of boundary face e_b. For an internal face e_f, $f \in \mathcal{F}_p$ the outward normal vector to the face is denoted by $\mathbf{n}_{pf}$, and its length is the area of the face, that is, $|\mathbf{n}_{pf}| = |e_f|$. Clearly, if $f \in \mathcal{F}_p \cap \mathcal{F}_q$ for $q \in \mathcal{N}_p$, then we have $\mathbf{n}_{pf} = -\mathbf{n}_{qf}$. For an index of boundary face $b \in \mathcal{B}_p$, we abuse the notation to denote the outward normal vector to the boundary of computational domain as $\mathbf{n}_b = \mathbf{n}_{pb}$ with $|\mathbf{n}_b| = |e_b|$.

In this section, we focus on a spatial discretization, and then in the next section we discuss a temporal discretization. To apply a finite volume method for a spatial discretization, we integrate (4) on a cell Ω_p:

$$\int_{\Omega_p} \frac{1}{|\nabla \phi|_\epsilon} \frac{\partial \phi}{\partial t} = \int_{\Omega_p} \nabla \cdot (g \nabla \phi) = \sum_{f \in \mathcal{F}_p} \int_{e_f} g \nabla \phi \cdot \mathbf{n} + \sum_{b \in \mathcal{B}_p} \int_{e_b} g \nabla \phi \cdot \mathbf{n}, \tag{5}$$

where $g \equiv |\nabla \phi|_\epsilon^{-1}$ and $\mathbf{n}$ is an outward normal vector. An approximation of the normal flux $g \nabla \phi \cdot \mathbf{n}$ should be formulated at internal and boundary faces e_f and e_b, respectively. The approximation in a finite volume method is represented by unknowns

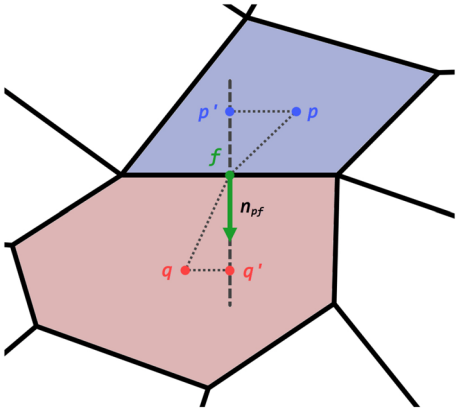

Fig. 1 An illustration of the basic notation used in the flux balanced approximation

$\phi_p = \phi(\mathbf{x}_p)$, where $\mathbf{x}_p$ is the center of cell Ω_p, $p \in \mathcal{I}$, and by Dirichlet boundary conditions $\phi_b = \phi_b(\mathbf{x}_b)$, where $\mathbf{x}_b$ is the center of boundary face e_b, $b \in \mathcal{B}$. In the case of Neumann boundary condition, the boundary flux can be directly replaced by the given condition. To describe the spatial discretization, we introduce a notation of directional vectors denoted by $\mathbf{d}$ and some proper indices. Namely, the vector $\mathbf{d}_{\alpha\beta} \equiv \mathbf{x}_\beta - \mathbf{x}_\alpha$ is defined by the directional vector from the point $\mathbf{x}_\alpha$ to the point $\mathbf{x}_\beta$.

At the left-hand side of (5), we assume that a gradient at the center of cell $\mathbf{x}_p \in \Omega_p$ is constant on each cell Ω_p:

$$\int_{\Omega_p} \frac{1}{|\nabla\phi|_\epsilon} \frac{\partial\phi}{\partial t} \approx \frac{1}{|\nabla\phi_p|_\epsilon} \int_{\Omega_p} \frac{\partial\phi}{\partial t}, \tag{6}$$

where the gradient is computed by the inverse distance weighted least-squares minimization [35, 36]:

$$\nabla\phi_p \equiv \underset{\mathbf{y}\in\mathbb{R}^3}{\arg\min} \left(\sum_{q\in\mathcal{N}_p} \frac{(\phi_p + \mathbf{y}\cdot\mathbf{d}_{pq} - \phi_q)^2}{|\mathbf{d}_{pq}|^2} + \sum_{b\in\mathcal{B}_p} \frac{(\phi_p + \mathbf{y}\cdot\mathbf{d}_{pb} - \phi_b)^2}{|\mathbf{d}_{pb}|^2} \right). \tag{7}$$

The minimizer is directly obtained by multiplying the inverse weight matrix

$$\mathcal{L}_p(\phi_p, \phi_b) \equiv \nabla\phi_p = \sum_{q\in\mathcal{N}_p} \frac{\mathbf{M}^{-1}\mathbf{d}_{pq}}{|\mathbf{d}_{pq}|^2} (\phi_q - \phi_p) + \sum_{b\in\mathcal{B}_p} \frac{\mathbf{M}^{-1}\mathbf{d}_{pb}}{|\mathbf{d}_{pb}|^2} (\phi_b - \phi_p), \tag{8}$$

where the weight matrix $\mathbf{M}$ is

$$\mathbf{M} = \sum_{q\in\mathcal{N}_p} \frac{\mathbf{d}_{pq}\otimes\mathbf{d}_{pq}}{|\mathbf{d}_{pq}|^2} + \sum_{b\in\mathcal{B}_p} \frac{\mathbf{d}_{pb}\otimes\mathbf{d}_{pb}}{|\mathbf{d}_{pb}|^2},$$

and $\mathbf{v}\otimes\mathbf{w}$ is the tensor product which gives a 3×3 matrix $\mathbf{v}\mathbf{w}^{\mathrm{T}}$ for $\mathbf{v}, \mathbf{w} \in \mathbb{R}^3$.

At the right-hand side of (5), we begin with an approximation of the internal flux at e_f, $f \in \mathcal{F}_p \cap \mathcal{F}_q$. Since the normal flux at the internal face should be balanced, we use the property of flux-balanced approximation:

$$\int_{e_f} g_p \nabla\phi \cdot \mathbf{n}_{pf} = -\int_{e_f} g_q \nabla\phi \cdot \mathbf{n}_{qf}. \tag{9}$$

In order to discretize the above property, we temporally use an auxiliary variable $\phi_f = \phi(\mathbf{x}_f)$, where $\mathbf{x}_f$ is the center of face e_f. From an orthogonal splitting of the vectors $\mathbf{d}_{pf}$ and $\mathbf{d}_{qf}$,

$$g_p \mathbf{n}_{pf} = c_p \mathbf{d}_{pf} + \mathbf{t}_{pf} \quad \text{and} \quad g_q \mathbf{n}_{qf} = c_q \mathbf{d}_{qf} + \mathbf{t}_{qf}, \tag{10}$$

where $\mathbf{n}_{pf} \perp \mathbf{t}_{pf}$ and $\mathbf{n}_{qf} \perp \mathbf{t}_{qf}$ in Fig. 1, we construct the discretization of (9):

$$c_p(\phi_f - \phi_p) + \mathbf{t}_{pf} \cdot \nabla\phi_p = -c_q(\phi_f - \phi_q) - \mathbf{t}_{qf} \cdot \nabla\phi_q, \tag{11}$$

by approximations $c_p \nabla\phi \cdot \mathbf{d}_{pf} \approx c_p(\phi_f - \phi_p)$ and $c_q \nabla\phi \cdot \mathbf{d}_{qf} \approx c_q(\phi_f - \phi_q)$. Constants c_p and c_q are directly computed from (10) using an orthogonality:

$$c_p = g_p \frac{\mathbf{n}_{pf} \cdot \mathbf{n}_{pf}}{\mathbf{n}_{pf} \cdot \mathbf{d}_{pf}} \quad \text{and} \quad c_q = g_q \frac{\mathbf{n}_{qf} \cdot \mathbf{n}_{qf}}{\mathbf{n}_{qf} \cdot \mathbf{d}_{qf}}. \tag{12}$$

Now, we have the approximation for the internal normal fluxes in (5) by using the cell and face center values:

$$\sum_{f \in \mathcal{F}_p} \int_{e_f} g \nabla\phi \cdot \mathbf{n} \approx \sum_{f \in \mathcal{F}_p} c_p(\phi_f - \phi_p) + \mathbf{t}_{pf} \cdot \nabla\phi_p. \tag{13}$$

Since the summation in (13) takes place in the cell Ω_p, we approximate the value of g in (13) by the constant g_p. Expressing the auxiliary variable ϕ_f from (11) by using the values of cell centers and their gradients,

$$\phi_f = \frac{c_p}{c_p + c_q}\phi_p + \frac{c_q}{c_p + c_q}\phi_q - \frac{1}{c_p + c_q}\left(\nabla\phi_p \cdot \mathbf{t}_{pf} + \nabla\phi_q \cdot \mathbf{t}_{qf}\right), \tag{14}$$

the discretization of internal flux can be written without ϕ_f in (13):

$$\sum_{f \in \mathcal{F}_p} \int_{e_f} g \nabla\phi \cdot \mathbf{n} \approx \sum_{f \in \mathcal{F}_p} \bar{c}_{pq}(\phi_q - \phi_p) + \bar{c}_{pq}\left(\nabla\phi_p \cdot \frac{\mathbf{t}_{pf}}{c_p} - \nabla\phi_q \cdot \frac{\mathbf{t}_{qf}}{c_q}\right), \tag{15}$$

where

$$\bar{c}_{pq} = \frac{c_p c_q}{c_p + c_q}. \tag{16}$$

The discretization of boundary flux is obtained similarly, and we obtain the final form of spatial discretization:

$$\frac{1}{|\nabla\phi_p|_\epsilon}\int_{\Omega_p}\frac{\partial\phi}{\partial t} = \sum_{f\in\mathcal{F}_p}\bar{c}_{pq}(\phi_q-\phi_p)+\bar{c}_{pq}\left(\nabla\phi_p\cdot\frac{\mathbf{t}_{pf}}{c_p}-\nabla\phi_q\cdot\frac{\mathbf{t}_{qf}}{c_q}\right) + \sum_{b\in\mathcal{B}_p}c_{pb}(\phi_b-\phi_p)+\nabla\phi_p\cdot\mathbf{t}_{pb}, \tag{17}$$

where $c_{pb} = g_p\mathbf{n}_b\cdot\mathbf{n}_b/(\mathbf{n}_b\cdot\mathbf{d}_{pb})$, $g_p\mathbf{n}_b = c_{pb}\mathbf{d}_{pb}+\mathbf{t}_{pb}$, and $\mathbf{n}_b\perp\mathbf{t}_{pb}$, for $b\in\mathcal{B}_p$.

Remark 1 A geometrical interpretation of summands in (17) brings us a connection to the alternative derivation of the spatial discretization. The tangential vector $\mathbf{t}_{pf}/c_p$ in (17) can be written as

$$-\frac{\mathbf{t}_{pf}}{c_p} = \mathbf{d}_{pf}-\frac{g_p}{c_p}\mathbf{n}_{pf} = \mathbf{d}_{pf}-\frac{\mathbf{n}_{pf}\cdot\mathbf{d}_{pf}}{\mathbf{n}_{pf}\cdot\mathbf{n}_{pf}}\mathbf{n}_{pf} = \mathbf{x}_{p'}-\mathbf{x}_p = \mathbf{d}_{pp'}, \tag{18}$$

where the relations (10) and (12) are used, and $\mathbf{x}_{p'}$ is a projection of $\mathbf{x}_p$ onto the line

$$\mathbf{r}_f(s) = \mathbf{x}_f + s\mathbf{n}_{pf},\quad s\in\mathbb{R}, \tag{19}$$

which passes through the center of face $\mathbf{x}_f$, and whose directional vector is $\mathbf{n}_{pf}$, see the Fig. 1. Similarly, we have also $-\mathbf{t}_{qf}/c_q = \mathbf{d}_{qq'}$. Then, the internal normal fluxes in (15) can be written in the following form by using $\phi_{p'}\approx\phi_p+\nabla\phi_p\cdot\mathbf{d}_{pp'}$ and $\phi_{q'}\approx\phi_q+\nabla\phi_q\cdot\mathbf{d}_{qq'}$ and relations (12):

$$\begin{aligned}\sum_{f\in\mathcal{F}_p}\int_{e_f}g\nabla\phi\cdot\mathbf{n} &\approx \sum_{f\in\mathcal{F}_p}\bar{c}_{pq}\left(\phi_{q'}-\phi_{p'}\right)\\ &= \sum_{f\in\mathcal{F}_p}\frac{1}{c_p^{-1}+c_q^{-1}}\left(\phi_{q'}-\phi_{p'}\right) = \sum_{f\in\mathcal{F}_p}\frac{1}{\frac{|\mathbf{d}_{p'f}|}{g_p|\mathbf{n}_{pf}|}+\frac{|\mathbf{d}_{q'f}|}{g_q|\mathbf{n}_{qf}|}}\left(\phi_{q'}-\phi_{p'}\right)\\ &= \sum_{f\in\mathcal{F}_p}\frac{|\mathbf{d}_{p'f}|+|\mathbf{d}_{q'f}|}{\frac{|\mathbf{d}_{p'f}|}{g_p}+\frac{|\mathbf{d}_{q'f}|}{g_q}}\frac{\phi_{q'}-\phi_{p'}}{|\mathbf{d}_{p'q'}|}|\mathbf{n}_{pf}|.\end{aligned} \tag{20}$$

In such case, the internal normal flux is approximated by the directional difference of the values ϕ at $\mathbf{x}_{p'}$ and $\mathbf{x}_{q'}$, which are the projection points of $\mathbf{x}_p$ and $\mathbf{x}_q$, respectively, onto the line $\mathbf{r}_f$ given in (19). The directional difference is multiplied by the distance weighted harmonic mean of g_p and g_q, see also [38].

In the same manner of deriving (20), the boundary normal flux $\nabla\phi \cdot \mathbf{n}_b$ in (17) is geometrically interpreted:

$$\sum_{b\in\mathcal{B}_p}\int_{e_b} g\nabla\phi\cdot\mathbf{n} \approx \sum_{b\in\mathcal{B}_p} c_{pb}(\phi_b-\phi_p-\nabla\phi_p\cdot\mathbf{d}_{pp'}) = \sum_{b\in\mathcal{B}_p} c_{pb}(\phi_b-\phi_{p'}). \tag{21}$$

Combining (20) and (21), our discretization scheme (17) can be also written in the form inspired by [38]:

$$\frac{1}{|\nabla\phi_p|_\epsilon}\int_{\Omega_p}\frac{\partial\phi}{\partial t} = \sum_{f\in\mathcal{F}_p}\bar{c}_{pq}(\phi_{q'}-\phi_{p'}) + \sum_{b\in\mathcal{B}_p} c_{pb}(\phi_b-\phi_{p'}), \tag{22}$$

from which we clearly recover (17).

3 Nonlinear Crank–Nicolson method

Let us denote a size of time step as Δt and $\phi_p^n = \phi(\mathbf{x}_p, n\Delta t)$, $p \in \mathcal{I}$, and $n \in \mathbb{N}$. A given initial condition is denoted by $\phi^0 \equiv (\phi_1^0, \phi_2^0, \ldots, \phi_{|\mathcal{I}|}^0)^{\mathrm{T}}$. Since the spatial discretization (17) contains nonlinear terms, one can linearize it using a semi-implicit method similar to [28, 31]. The numerical solution ϕ^n is then obtained by solving a system of linear algebraic equations:

$$\begin{aligned}\frac{|\Omega_p|}{\Delta t}\left(\phi_p^n-\phi_p^{n-1}\right) = &\sum_{f\in\mathcal{F}_p}\alpha_{pf}^{n-1}\left(\phi_q^n-\phi_p^n+\nabla\phi_q^n\cdot\mathbf{d}_{qq'}-\nabla\phi_p^n\cdot\mathbf{d}_{pp'}\right)\\ &+\sum_{b\in\mathcal{B}_p}\alpha_{pb}^{n-1}\left(\phi_b^n-\phi_p^n-\nabla\phi_p^n\cdot\mathbf{d}_{pp'}\right),\end{aligned} \tag{23}$$

where the coefficients are

$$\alpha_{pf}^{n-1} = \bar{c}_{pq}^{n-1}|\nabla\phi_p^{n-1}|_\epsilon \quad\text{and}\quad \alpha_{pb}^{n-1} = c_{pb}^{n-1}|\nabla\phi_p^{n-1}|_\epsilon, \tag{24}$$

and the gradients are computed by the formula (8):

$$\nabla\phi_p^{n-1} = \mathcal{L}_p(\phi_p^{n-1}, \phi_b^{n-1}). \tag{25}$$

For a regular hexahedral mesh, it is rather straightforward to solve the linear system of equations (23). In the case of polyhedral mesh in three-dimensional (3D) domain, some practical issues should be considered. Comparing to the case of a regular hexahedral mesh, non-zero coefficients of the matrix in (23) for a polyhedral mesh are much widely distributed because a polyhedron cell has in general a larger number of faces than a hexahedron cell. It can cause a heavy computational cost to solve the linear algebraic equation. Moreover, the presence of $\nabla\phi_q$ at the nth time step requires to use at least two

overlapping layers of cells between decomposed domains for a parallel computation. It causes a larger communication cost than when using only one overlapping layers of cells.

In order to avoid the mentioned issues, we apply a deferred correction method in (23). For $n \geq 1$ and $k \geq 1$, a numerical solution $\phi^{n,k}$ is updated from $\phi^{n,k-1}$ and ϕ^{n-1} by solving the following system of linear algebraic equations:

$$\begin{aligned}\frac{|\Omega_p|}{\Delta t}\left(\phi_p^{n,k}-\phi_p^{n-1}\right) = &\sum_{f\in\mathcal{F}_p}\alpha_{pf}^{n-1}\left(\phi_q^{n,k}-\phi_p^{n,k}+\nabla\phi_q^{n,k-1}\cdot\mathbf{d}_{qq'}-\nabla\phi_p^{n,k-1}\cdot\mathbf{d}_{pp'}\right)\\ &+\sum_{b\in\mathcal{B}_p}\alpha_{pb}^{n-1}\left(\phi_b^{n}-\phi_p^{n,k}-\nabla\phi_p^{n,k-1}\cdot\mathbf{d}_{pp'}\right),\end{aligned} \tag{26}$$

where $\phi_p^{n,0} \equiv \phi_p^{n-1}$, $p \in \mathcal{I}$ and

$$\nabla\phi_p^{n,k-1} = \mathcal{L}_p(\phi_p^{n,k-1}, \phi_b^n), \quad k \geq 1. \tag{27}$$

Rewriting (26) formally as the matrix equation:

$$\mathbf{A}^{n-1}\phi^{n,k} = \mathbf{f}(\phi^{n,k-1}), \tag{28}$$

the k^{th} iteration is stopped at the smallest K_n such that a residual error is smaller than a chosen error bound η:

$$\frac{1}{|\mathcal{I}|}\sum_{p\in\mathcal{I}}\left|\left(\mathbf{A}^{n-1}\phi^{n,K_n}-\mathbf{f}(\phi^{n,K_n})\right)_p\right| < \eta, \tag{29}$$

where an above parenthesis with a subscript $()_p$ denotes the p^{th} component of a vector in the parenthesis. Then, we define $\phi^n \equiv \phi^{n,K_n}$.

An experimental order of convergence is presented in Sect. 4 and it depends on a choice of size of time step Δt in (26). In the case of a regular hexahedral mesh, it is expected to be a second-order convergence by using the semi-implicit method (23) if a size of the time step is proportional to a square of an average size of cells. Such a time step is unnecessary to obtain the second-order convergence for the normal or advective flow terms in (2). In [25], a size of time step proportional to an average size of cells is enough to obtain the second-order convergence in the iterative inflow-implicit and outflow-explicit (IIOE) method. Therefore, in order to consistently combine the time discretization for the mean curvature flow with the iterative IIOE method, we propose here a nonlinear Crank–Nicolson method with a deferred correction method. It is derived by combining a fully-implicit and a fully-explicit method, for $n \geq 1$ and $k \geq 1$,

$$\frac{|\Omega_p|}{\Delta t}\left(\phi_p^{n,k}-\phi_p^{n-1}\right) = \frac{1}{2}\sum_{f\in\mathcal{F}_p}\alpha_{pf}^{n,k-1}\left(\phi_q^{n,k}-\phi_p^{n,k}+\nabla\phi_q^{n,k-1}\cdot\mathbf{d}_{qq'}-\nabla\phi_p^{n,k-1}\cdot\mathbf{d}_{pp'}\right)$$

$$
\begin{aligned}
&+\frac{1}{2}\sum_{b\in\mathcal{B}_p}\alpha_{pb}^{n,k-1}\left(\phi_b^n-\phi_p^{n,k}-\nabla\phi_p^{n,k-1}\cdot\mathbf{d}_{pp'}\right)\\
&+\frac{1}{2}\sum_{f\in\mathcal{F}_p}\alpha_{pf}^{n-1}\left(\phi_q^{n-1}-\phi_p^{n-1}+\nabla\phi_q^{n-1}\cdot\mathbf{d}_{qq'}-\nabla\phi_p^{n-1}\cdot\mathbf{d}_{pp'}\right)\\
&+\frac{1}{2}\sum_{b\in\mathcal{B}_p}\alpha_{pb}^{n-1}\left(\phi_b^{n-1}-\phi_p^{n-1}-\nabla\phi_p^{n-1}\cdot\mathbf{d}_{pp'}\right),
\end{aligned}
\tag{30}
$$

where the coefficients are now given by

$$
\alpha_{pf}^{n,k-1}=\bar{c}_{pq}^{n,k-1}|\nabla\phi_p^{n,k-1}|_\epsilon\quad\text{and}\quad\alpha_{pb}^{n,k-1}=c_{pb}^{n,k-1}|\nabla\phi_p^{n,k-1}|_\epsilon.
\tag{31}
$$

Again, rewritting (30) as a matrix equation

$$
\mathbf{A}^{n,k-1}\phi^{n,k}=\mathbf{f}(\phi^{n,k-1}),
\tag{32}
$$

the k^{th} iteration is stopped at the smallest K_n such that the residual error is smaller than the error bound η:

$$
\frac{1}{|\mathcal{I}|}\sum_{p\in\mathcal{I}}|\left(\mathbf{A}^{n,K_n}\phi^{n,K_n}-\mathbf{f}(\phi^{n,K_n})\right)_p|<\eta.
\tag{33}
$$

Note that, opposite to (32), the coefficient matrix $\mathbf{A}$ of (28) from the semi-implicit method does not change in a middle of the k-iteration (26). In Sect. 4, numerical experiments using the semi-implicit and the nonlinear Crank–Nicolson methods are presented to discuss about their advantages and disadvantages.

Combining the RHS of (30) and (A9), we finally obtain the proposed finite volume method to solve a general type of level-set equation (2) on polyhedral meshes:

$$
\begin{aligned}
\frac{|\Omega_p|}{\Delta t}\left(\phi_p^{n,k}-\phi_p^{n-1}\right)=&-\sum_{i\in\bar{\mathcal{F}}_p^-}\left(\phi_q^{n,k}+\mathcal{D}_q\phi^{n,k-1}\cdot\mathbf{d}_{qi}-\phi_p^{n,k}\right)a_{pi}^{n-1}\\
&-\sum_{i\in\bar{\mathcal{B}}_p^-}\left(\phi_{bi}^n-\phi_p^{n,k}\right)a_{pi}^{n-1}-\sum_{i\in\bar{\mathcal{B}}_p^+\cup\bar{\mathcal{F}}_p^+}\left(\mathcal{D}_p\phi^{n-1}\cdot\mathbf{d}_{pi}\right)a_{pi}^{n-1}\\
&+\frac{1}{2}\sum_{f\in\mathcal{F}_p}\alpha_{pf}^{n,k-1}\left(\phi_q^{n,k}-\phi_p^{n,k}+\nabla\phi_q^{n,k-1}\cdot\mathbf{d}_{qq'}-\nabla\phi_p^{n,k-1}\cdot\mathbf{d}_{pp'}\right)\\
&+\frac{1}{2}\sum_{b\in\mathcal{B}_p}\alpha_{pb}^{n,k-1}\left(\phi_b^n-\phi_p^{n,k}-\nabla\phi_p^{n,k-1}\cdot\mathbf{d}_{pp'}\right)\\
&+\frac{1}{2}\sum_{f\in\mathcal{F}_p}\alpha_{pf}^{n-1}\left(\phi_q^{n-1}-\phi_p^{n-1}+\nabla\phi_q^{n-1}\cdot\mathbf{d}_{qq'}-\nabla\phi_p^{n-1}\cdot\mathbf{d}_{pp'}\right)\\
&+\frac{1}{2}\sum_{b\in\mathcal{B}_p}\alpha_{pb}^{n-1}\left(\phi_b^{n-1}-\phi_p^{n-1}-\nabla\phi_p^{n-1}\cdot\mathbf{d}_{pp'}\right).
\end{aligned}
\tag{34}
$$

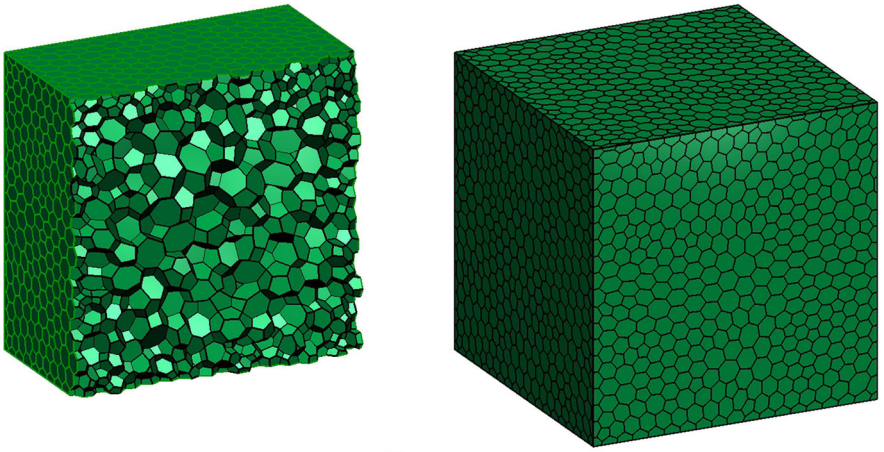

Fig. 2 A polyhedral mesh in the cube domain $\Omega_1 = [-1.25, 1.25]^3 \subset \mathbb{R}^3$ is illustrated with the average size of cells (35) $h_1 = 1.90 \cdot 10^{-1}$ corresponding to the level $N = 1$ in Table 1. The shape of boundary of Ω_1 is shown on the right side and its inside is shown on the left side

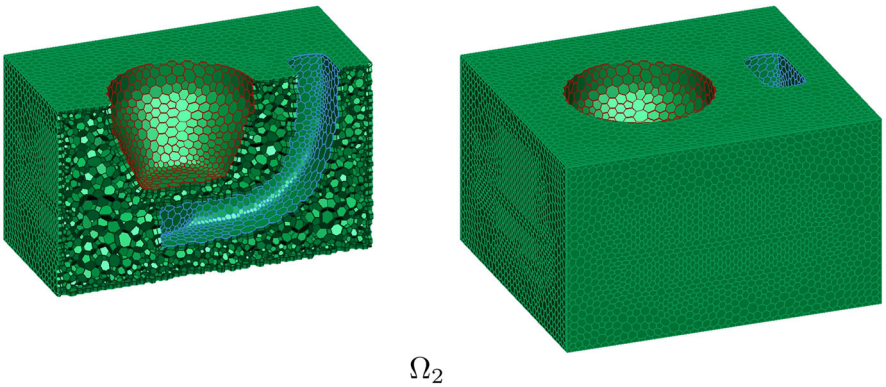

Fig. 3 A polyhedral mesh for the domain Ω_2 of complex shape is illustrated with the average size of cells (35) $h_1 = 4.17 \cdot 10^{-2}$ corresponding to the level $N = 1$ in Table 1. The shape of boundary of Ω_2 is shown on the right side and its inside is shown on the left side

Analogously to the previous methods, the k^{th} iteration is stopped at the smallest K_n such that a residual error is smaller than an error bound η in (33). In this case, a formal matrix $\mathbf{A}$ and a vector $\mathbf{f}$ in (33) are obtained by (30) and (A9).

4 Numerical Experiments

Firstly, we use some exact solutions of (2) in order to quantitatively observe a behavior of the proposed numerical algorithm (30). Since the advective or normal flow equations and their combination are tested in [25], we mostly focus here on numerical tests

Table 1 The average sizes of cells (35) for each level N are listed for the cube and the domain of complex shape in Figs. 2 and 3, respectively

	N	1	2	3	4
h_N	Ω_1	$1.90 \cdot 10^{-1}$	$9.52 \cdot 10^{-2}$	$4.76 \cdot 10^{-2}$	$2.48 \cdot 10^{-2}$
	Ω_2	$6.64 \cdot 10^{-2}$	$4.17 \cdot 10^{-2}$	$2.27 \cdot 10^{-2}$	$1.29 \cdot 10^{-2}$

of the mean curvature flow equation and combinations with advective and normal flow equations. Secondly, a qualitative comparison of numerical solutions between hexahedral and polyhedral meshes are presented. Finally, as a typical example of the engineering application of (2), the G-equation in a premixed turbulent combustion is shown in a gasoline directed injection (GDI) engine at the end of this section. Note that polyhedral meshes in Figs. 2, 3, and 6 are generated by AVL FIRE™.

The experimental order of convergence (EOC) is checked by using appropriate norms of errors. The average size of cells is defined by

$$h_N = \frac{1}{|\mathcal{I}_N|} \sum_{p \in \mathcal{I}_N} |\Omega_p|_B^{1/3}, \quad N \in \{1, 2, 3, 4\}, \tag{35}$$

where $\mathcal{I}_N$ is the set of cell indices of the level N, $|\mathcal{I}_N|$ is the number of cells, and $|\Omega_p|_B$ is the volume of the smallest box aligned with the coordinate axes to enclose the cell Ω_p. The mesh of higher level is generated using smaller volumes of cells. The four levels of meshes are used to compute the EOC with the average sizes of cells listed in the Table 1. The following norms of the errors are used. The norms of errors E^2 and E^∞ are the $L^2((0, T) \times \Omega)$ and $L^\infty(0, T; L^2(\Omega))$ norms of the difference between an exact and a numerical solution, respectively. The norms of errors G^2 and G^∞ are the $L^2((0, T) \times \Omega)^3$ and $L^\infty(0, T; L^2(\Omega)^3)$ norms of the difference between the gradient of exact and numerical solution, respectively. If E^2 is used to compute a corresponding EOC, it is computed by

$$EOC_N = \frac{\log(E_N^2 / E_{N-1}^2)}{\log(h_N / h_{N-1})}, \quad N \in \{2, 3, 4\}, \tag{36}$$

where E_N^2 is the norm of error E^2 on the mesh of the level N.

We use the following default values for all examples in this section, if there are no other comments: the threshold to stop the k-iteration as $\eta = 10^{-10}$ in (33) and the regularization parameter (4) as $\epsilon = h_N^2$ for each level N.

4.1 Experimental order of convergence (*EOC*)

We use two exact solutions of the mean curvature flow equation, that is, (2) with $\epsilon = 0$, $\mathbf{v} = 0$, $\delta = 0$, and $\gamma = 1$, computed on Ω_1 in Fig. 2 and Ω_2 in Fig. 3 in order

Table 2 The EOC of numerical solution of (37) with $l = 1$ on Ω_1 (top) and Ω_2 (bottom) is presented by using the proposed method (30) (top) and the method in [36] (bottom)

N	E^2	EOC	E^∞	EOC	G^2	EOC	G^∞	EOC
Ω_1								
1	$3.156 \cdot 10^{-3}$		$8.337 \cdot 10^{-3}$		$2.128 \cdot 10^{-2}$		$6.089 \cdot 10^{-2}$	
2	$6.892 \cdot 10^{-4}$	2.20	$2.108 \cdot 10^{-3}$	1.99	$8.620 \cdot 10^{-3}$	1.31	$2.731 \cdot 10^{-2}$	1.16
3	$2.000 \cdot 10^{-4}$	1.78	$6.391 \cdot 10^{-4}$	1.72	$3.909 \cdot 10^{-3}$	1.14	$1.477 \cdot 10^{-2}$	0.89
4	$5.485 \cdot 10^{-5}$	1.99	$2.286 \cdot 10^{-4}$	1.58	$1.728 \cdot 10^{-3}$	1.26	$7.917 \cdot 10^{-3}$	0.96
Ω_2								
1	$5.868 \cdot 10^{-4}$		$1.848 \cdot 10^{-3}$		$9.684 \cdot 10^{-3}$		$3.467 \cdot 10^{-2}$	
2	$2.052 \cdot 10^{-4}$	2.25	$8.243 \cdot 10^{-4}$	1.73	$4.671 \cdot 10^{-3}$	1.56	$2.172 \cdot 10^{-2}$	1.00
3	$5.330 \cdot 10^{-5}$	2.23	$3.430 \cdot 10^{-4}$	1.45	$2.002 \cdot 10^{-3}$	1.40	$1.256 \cdot 10^{-2}$	0.90
4	$1.814 \cdot 10^{-5}$	1.91	$1.438 \cdot 10^{-4}$	1.54	$9.433 \cdot 10^{-4}$	1.33	$7.517 \cdot 10^{-3}$	0.91
Ω_1								
1	$3.541 \cdot 10^{-3}$		$9.412 \cdot 10^{-3}$		$2.357 \cdot 10^{-2}$		$6.598 \cdot 10^{-2}$	
2	$7.388 \cdot 10^{-4}$	2.27	$2.145 \cdot 10^{-3}$	2.14	$9.580 \cdot 10^{-3}$	1.30	$3.002 \cdot 10^{-2}$	1.14
3	$2.082 \cdot 10^{-4}$	1.83	$7.586 \cdot 10^{-4}$	1.50	$4.459 \cdot 10^{-3}$	1.10	$1.721 \cdot 10^{-2}$	0.80
4	$4.698 \cdot 10^{-5}$	2.29	$2.345 \cdot 10^{-4}$	1.81	$1.859 \cdot 10^{-3}$	1.35	$8.245 \cdot 10^{-3}$	1.13
Ω_2								
1	$7.034 \cdot 10^{-4}$		$2.007 \cdot 10^{-3}$		$1.034 \cdot 10^{-2}$		$3.611 \cdot 10^{-2}$	
2	$2.300 \cdot 10^{-4}$	2.40	$8.579 \cdot 10^{-4}$	1.82	$4.919 \cdot 10^{-3}$	1.59	$2.209 \cdot 10^{-2}$	1.05
3	$5.434 \cdot 10^{-5}$	2.38	$3.486 \cdot 10^{-4}$	1.49	$2.047 \cdot 10^{-3}$	1.45	$1.218 \cdot 10^{-2}$	0.98
4	$1.727 \cdot 10^{-5}$	2.03	$1.444 \cdot 10^{-4}$	1.56	$9.668 \cdot 10^{-4}$	1.33	$7.192 \cdot 10^{-3}$	0.93

Table 3 The EOC of numerical solution of (37) with $l = 2$ is presented by using the proposed method (30) (top) and the method in [36] (bottom)

N	E^2	EOC	E^∞	EOC	G^2	EOC	G^∞	EOC
Ω_1								
1	$4.734 \cdot 10^{-3}$		$1.522 \cdot 10^{-2}$		$4.335 \cdot 10^{-2}$		$1.130 \cdot 10^{-1}$	
2	$9.588 \cdot 10^{-4}$	2.31	$2.972 \cdot 10^{-3}$	2.36	$1.604 \cdot 10^{-2}$	1.44	$4.086 \cdot 10^{-2}$	1.47
3	$1.709 \cdot 10^{-4}$	2.49	$4.802 \cdot 10^{-4}$	2.63	$6.212 \cdot 10^{-3}$	1.37	$1.714 \cdot 10^{-2}$	1.25
4	$3.367 \cdot 10^{-5}$	2.50	$9.274 \cdot 10^{-5}$	2.53	$2.501 \cdot 10^{-3}$	1.40	$7.249 \cdot 10^{-3}$	1.32
Ω_2								
1	$7.029 \cdot 10^{-4}$		$1.955 \cdot 10^{-3}$		$1.128 \cdot 10^{-2}$		$2.950 \cdot 10^{-2}$	
2	$2.019 \cdot 10^{-4}$	2.68	$5.589 \cdot 10^{-4}$	2.69	$5.364 \cdot 10^{-3}$	1.59	$1.484 \cdot 10^{-2}$	1.47
3	$3.727 \cdot 10^{-5}$	2.79	$1.035 \cdot 10^{-4}$	2.78	$2.022 \cdot 10^{-3}$	1.61	$5.991 \cdot 10^{-3}$	1.50
4	$7.613 \cdot 10^{-6}$	2.81	$2.123 \cdot 10^{-5}$	2.80	$9.192 \cdot 10^{-4}$	1.40	$2.792 \cdot 10^{-3}$	1.35
Ω_1								
1	$5.020 \cdot 10^{-3}$		$1.489 \cdot 10^{-2}$		$4.788 \cdot 10^{-2}$		$1.236 \cdot 10^{-1}$	
2	$1.064 \cdot 10^{-3}$	2.24	$2.813 \cdot 10^{-3}$	2.41	$1.878 \cdot 10^{-2}$	1.35	$4.994 \cdot 10^{-2}$	1.31
3	$3.012 \cdot 10^{-4}$	1.82	$8.937 \cdot 10^{-4}$	1.65	$7.733 \cdot 10^{-3}$	1.28	$2.262 \cdot 10^{-2}$	1.14
4	$8.598 \cdot 10^{-5}$	1.93	$2.742 \cdot 10^{-4}$	1.82	$3.148 \cdot 10^{-3}$	1.38	$9.847 \cdot 10^{-3}$	1.28
Ω_2								
1	$9.718 \cdot 10^{-4}$		$2.581 \cdot 10^{-3}$		$1.406 \cdot 10^{-2}$		$3.917 \cdot 10^{-2}$	
2	$2.914 \cdot 10^{-4}$	2.58	$8.027 \cdot 10^{-4}$	2.51	$6.683 \cdot 10^{-3}$	1.59	$2.019 \cdot 10^{-2}$	1.42
3	$6.076 \cdot 10^{-5}$	2.59	$1.792 \cdot 10^{-4}$	2.48	$2.457 \cdot 10^{-3}$	1.65	$7.993 \cdot 10^{-3}$	1.53
4	$2.097 \cdot 10^{-5}$	1.88	$6.639 \cdot 10^{-5}$	1.76	$1.147 \cdot 10^{-3}$	1.35	$3.834 \cdot 10^{-3}$	1.30

Table 4 The EOC of numerical solution of (37) with $l = 1$ on a cube domain Ω_1 is presented by using the semi-implicit method

N	E^2	EOC	E^∞	EOC	G^2	EOC	G^∞	EOC
1	$2.847 \cdot 10^{-3}$		$7.423 \cdot 10^{-3}$		$1.980 \cdot 10^{-2}$		$5.125 \cdot 10^{-2}$	
2	$5.616 \cdot 10^{-4}$	2.35	$1.735 \cdot 10^{-3}$	2.10	$8.547 \cdot 10^{-3}$	1.22	$2.351 \cdot 10^{-2}$	1.13
3	$3.794 \cdot 10^{-4}$	0.57	$1.057 \cdot 10^{-3}$	0.71	$3.889 \cdot 10^{-3}$	1.14	$1.115 \cdot 10^{-2}$	1.08
4	$2.280 \cdot 10^{-4}$	0.78	$6.194 \cdot 10^{-4}$	0.82	$1.751 \cdot 10^{-3}$	1.23	$5.292 \cdot 10^{-3}$	1.15
N	E^2	EOC	E^∞	EOC	G^2	EOC	G^∞	EOC
1	$2.847 \cdot 10^{-3}$		$7.423 \cdot 10^{-3}$		$1.980 \cdot 10^{-2}$		$5.125 \cdot 10^{-2}$	
2	$4.076 \cdot 10^{-4}$	2.81	$1.162 \cdot 10^{-3}$	2.68	$8.422 \cdot 10^{-3}$	1.24	$2.309 \cdot 10^{-2}$	1.15
3	$1.312 \cdot 10^{-4}$	1.64	$4.175 \cdot 10^{-4}$	1.48	$3.772 \cdot 10^{-3}$	1.16	$1.069 \cdot 10^{-2}$	1.11
4	$4.045 \cdot 10^{-5}$	1.81	$1.260 \cdot 10^{-4}$	1.84	$1.657 \cdot 10^{-3}$	1.27	$4.868 \cdot 10^{-3}$	1.21

The top and bottom tables use a size of time step as Δt_1 and Δt_2 (38), respectively

to numerically check the EOC:

$$\phi^l(\mathbf{x}, t) = \left(\frac{|\mathbf{x}|^2}{4} + t\right)^{l/2}, \quad (\mathbf{x}, t) \in \Omega \times [0, T], \tag{37}$$

where $l = 1$ or 2, $\Omega = \Omega_1$ or Ω_2, and $T = 0.16$. The size of time step $\Delta t_N = T/2^{N-1}$ is used for each level $N \in \{1, 2, 3, 4\}$ in Table 1, which is proportional to the average size of cells. In Tables 2 and 3, the EOCs of numerical solutions of (37) with $l = 1$ and $l = 2$ are presented, respectively, and the results of the proposed method and the method in [36] are compared. The proposed numerical algorithm (30) shows that the EOCs are close to 2 in the norms of errors E^2 and E^∞ on Ω_1 and the EOCs are larger than 1 in the norms of errors G^2 and G^∞ on Ω_1 and Ω_2. By comparing errors in Table 2, we see that the proposed method and the method in [36] behaves similarly in case of (37) with $l = 1$. On the other hand, in Table 3 for (37) with $l = 2$, the proposed method shows better convergence order and it has about two or three times lower error on fine meshes.

4.2 Time discretization and regularization parameter

The two time discretizations in Sect. 3 are compared by using the exact solution (37) with $l = 1$ on Ω_1 in Fig. 2. In Table 4, we purposely use the following two sizes of time step for each level N in Table 1:

$$\Delta t_N^j = \frac{C}{(2^{N-1})^j}, \quad j = 1,\ 2, \quad N \in \{1, 2, 3, 4\}, \tag{38}$$

where $C = 0.04$. Note that Δt_N^1 is chosen to be proportional to an average size of cells and Δt_N^2 is to be proportional to a square of an average size of cells. In the case of the

Table 5 The total number of iterations to find a numerical solution at $T = 0.16$ of (37) with $l = 1$ on Ω_1 are compared between semi-implicit method (23) and nonlinear Crank–Nicolson method (30)

	N	1	2	3	4
(23)	K^1	24	48	80	128
	K^2	24	80	293	1024
(30)	K^1	52	120	288	580

The sizes of time step Δt_1 and Δt_2 (38) are used in K^1 and K^2 (39), respectively

semi-implicit method (23), the $EOC \simeq 2$ is obtained by the time step $\Delta t = \Delta t_N^2$, but the EOC is reduced to be 1 by the time step $\Delta t = \Delta t_N^1$, with respect to the norms of errors E^2 and E^∞. The EOCs with the norms of errors G^2 and G^∞ are approximately 1 regardless of the sizes of time step. In the case of nonlinear Crank–Nicolson method (30), even if the time step $\Delta t = \Delta t_N^1$ which is, of course, larger than Δt_N^2 is used, the $EOC \simeq 2$ is obtained in cases of the norms of errors E^2 and E^∞ by using; see Table 2.

When comparing computational costs, a method that uses a larger size of time step is not necessarily more efficient than a method that uses a smaller size of time step, because of, e.g., the number of iterations in iterative solvers. Therefore, for the semi-implicit method (23) and the nonlinear Crank–Nicolson method (30), we count also the total number of k-iteration to reach the final time $T = 0.16$ under the same stopping threshold $\eta = 10^{-10}$ in (29) and (33) in order to roughly estimate the computational cost:

$$K^j = \sum_{n=1}^{(2^{N-1})^j} K_n, \quad j = 1,\ 2. \tag{39}$$

When K^1 is compared between the semi-implicit method (23) and the nonlinear Crank–Nicolson method (30) in Table 5, the semi-implicit method is more efficient, but it is only first order accurate. To obtain second-order accuracy with the semi-implicit method, a size of time step proportional to a square of average size of cells should be used, that can, eventually, result in more iterations. Therefore, comparing between K^2 for the semi-implicit method (23) and K^1 for the nonlinear Crank–Nicolson method (30), the nonlinear Crank–Nicolson method has a lower computational cost with respect to the semi-implicit method when the highest level is used. It is caused by the fact that Δt_4^1 is 8 times larger than Δt_4^2. However, when the number of cells is smaller, the semi-implicit method shows less computational cost. In practice, the nonlinear Crank–Nicolson method for the regularized mean curvature flow equation is a reasonable choice to achieve second-order accuracy when we consider to solve a general type of level set equation, since a size of time step proportional to an average size of cells is used for the advective or normal flow equations.

In the rest of subsection, the effect of regularization parameter (4) is briefly tested for the exact solution of (37) with $l = 2$ on Ω_2 in Fig. 3. From a numerical point of view, the regularization parameter (4) is introduced to avoid a division by zero when $|\nabla\phi| = 0$ in (4), and it is chosen to have smaller values when the average size of

Table 6 An effect of the regularization parameter ϵ on numerical solutions of (37) with $l = 2$ for the domain Ω_2 of complex shape. The top and bottom tables use $\epsilon = 10^{-8}$ and $\epsilon = h_N$, respectively, for each level N in Table 1

N	E^2	EOC	E^∞	EOC	G^2	EOC	G^∞	EOC
1	$7.023 \cdot 10^{-4}$		$1.952 \cdot 10^{-3}$		$1.127 \cdot 10^{-2}$		$2.950 \cdot 10^{-2}$	
2	$2.019 \cdot 10^{-4}$	2.67	$5.587 \cdot 10^{-4}$	2.68	$5.364 \cdot 10^{-3}$	1.59	$1.484 \cdot 10^{-2}$	1.47
3	$3.726 \cdot 10^{-5}$	2.79	$1.034 \cdot 10^{-4}$	2.78	$2.022 \cdot 10^{-3}$	1.61	$5.991 \cdot 10^{-3}$	1.50
4	$7.612 \cdot 10^{-6}$	2.81	$2.123 \cdot 10^{-5}$	2.80	$9.192 \cdot 10^{-4}$	1.40	$2.792 \cdot 10^{-3}$	1.35

N	E^2	EOC	E^∞	EOC	G^2	EOC	G^∞	EOC
1	$1.013 \cdot 10^{-3}$		$3.056 \cdot 10^{-3}$		$1.217 \cdot 10^{-2}$		$3.159 \cdot 10^{-2}$	
2	$3.253 \cdot 10^{-4}$	2.44	$1.025 \cdot 10^{-3}$	2.34	$5.670 \cdot 10^{-3}$	1.64	$1.525 \cdot 10^{-2}$	1.56
3	$8.189 \cdot 10^{-5}$	2.28	$2.697 \cdot 10^{-4}$	2.20	$2.103 \cdot 10^{-3}$	1.64	$6.063 \cdot 10^{-3}$	1.52
4	$2.283 \cdot 10^{-5}$	2.26	$7.618 \cdot 10^{-5}$	2.24	$9.370 \cdot 10^{-4}$	1.43	$2.803 \cdot 10^{-3}$	1.37

Table 7 The EOC for the normal and mean curvature flow equation on Ω_1 in Fig. 2 using numerical solutions of (2) computed for $\mathbf{v} = 0$, $\delta = 1$, and $\gamma = 1$ with the initial condition $\phi_0(\mathbf{x}) = |\mathbf{x}|$ until $T = 1$

N	E^2	EOC	E^∞	EOC	G^2	EOC	G^∞	EOC
1	$1.824 \cdot 10^{-2}$		$3.058 \cdot 10^{-2}$		$9.262 \cdot 10^{-2}$		$1.211 \cdot 10^{-1}$	
2	$3.255 \cdot 10^{-3}$	2.49	$4.610 \cdot 10^{-3}$	2.74	$3.339 \cdot 10^{-2}$	1.48	$5.397 \cdot 10^{-2}$	1.17
3	$1.144 \cdot 10^{-3}$	1.51	$1.519 \cdot 10^{-3}$	1.60	$1.423 \cdot 10^{-2}$	1.23	$2.942 \cdot 10^{-2}$	0.88
4	$3.186 \cdot 10^{-4}$	1.97	$4.471 \cdot 10^{-4}$	1.88	$5.992 \cdot 10^{-3}$	1.33	$1.576 \cdot 10^{-2}$	0.96

cells is getting smaller. In Table 6, the comparison of using the values $\epsilon = 10^{-8}$ and $\epsilon = h_N$ for each level N in Table 1 is presented. Additionally, the case with $\epsilon = h_N^2$ is presented at the bottom of Table 3. Note that the choice of $\epsilon = 10^{-8}$ is smaller of order 10^{-4} than any values of $\epsilon = h_N^2$, so it is practically very small constant regardless of the average size of cells for the tested levels with Ω_1. The EOCs for the norms of errors E^2 and E^∞ are larger than 2 and the EOCs for the norms of errors G^2 and G^∞ are larger than 1 for all cases. The error values obtained by $\epsilon = 10^{-8}$ and $\epsilon = h_N^2$ are nearly same. However, the errors with $\epsilon = h_N$ from E^2 and E^∞ are nearly 2 times larger than with $\epsilon = 10^{-8}$ or $\epsilon = h_N^2$. In order to deal with diverse sizes of cells in a polyhedral mesh, a fixed regularization parameter is preferred to be practically used.

4.3 General level set equation

In this section, an exact solution of (2) described in [40] is used to numerically check the EOC on Ω_1 in Fig. 2 for general level-set equations. We briefly explain the solution $\phi(\mathbf{x}, t)$ for the initial condition $\phi_0(\mathbf{x}) = |\mathbf{x}|$ and some constants δ and $\gamma > 0$ in (2). Since an evolved profile of each level surface for $\phi(x, t)$ should preserve a spherical shape. the governing equation with $\mathbf{v} = 0$ and $\epsilon = 0$ in (2) of the change of radius

Table 8 The EOC for the advective, normal, and mean curvature flow equation on Ω_1 in Fig. 2 using numerical solutions of (2) computed for $\delta = 0.1$, $\gamma = 1$, and the rotational velocity (45) with the initial condition $\phi_0(\mathbf{x}) = |\mathbf{x} - \mathbf{x}_0|$ until $T = 1$

N	E^2	EOC	E^∞	EOC	G^2	EOC	G^∞	EOC
1	$9.288 \cdot 10^{-3}$		$1.343 \cdot 10^{-2}$		$8.315 \cdot 10^{-2}$		$1.124 \cdot 10^{-1}$	
2	$2.801 \cdot 10^{-3}$	1.73	$3.674 \cdot 10^{-3}$	1.88	$3.241 \cdot 10^{-2}$	1.36	$5.317 \cdot 10^{-2}$	1.08
3	$9.483 \cdot 10^{-4}$	1.56	$1.191 \cdot 10^{-3}$	1.62	$1.413 \cdot 10^{-2}$	1.20	$2.796 \cdot 10^{-2}$	0.93
4	$2.550 \cdot 10^{-4}$	2.02	$4.858 \cdot 10^{-4}$	1.38	$6.085 \cdot 10^{-3}$	1.30	$1.646 \cdot 10^{-2}$	0.82

$r(t) \geq 0$ can be written:

$$\frac{dr}{dt} = \delta + \gamma \frac{2}{r}, \quad r(0) = r_0 > 0. \tag{40}$$

Note that $\frac{2}{r}$ is the mean curvature of a sphere with a radius r. The exact solution of (40) is obtained by using Lambert function $\mathcal{W}$, that is, $z = \mathcal{W}(ze^z)$:

$$r(t) = \xi^{-1} \left(1 + \mathcal{W}\left((-1 + \xi r_0) e^{-1+\xi r_0 + \delta t}\right)\right), \tag{41}$$

where $\xi = \frac{\delta}{2\gamma}$. Consequently, for the initial condition $\phi_0(\mathbf{x}) = |\mathbf{x}|$, the exact solution of (2) with $\epsilon = 0$, $\mathbf{v} = 0$, $\delta > 0$, and $\gamma > 0$ is

$$\phi(\mathbf{x}, t) = \xi^{-1} \left(1 + \psi(\mathbf{x}, t)\right), \quad \psi(\mathbf{x}, t) \equiv \mathcal{W}\left((-1 + \xi |\mathbf{x}|) e^{-1+\xi|\mathbf{x}|+\delta t}\right), \tag{42}$$

and its gradient is computed analytically by

$$\nabla \phi(\mathbf{x}, t) = \left(\frac{\xi \psi(\mathbf{x}, t)}{(-1 + \xi |\mathbf{x}|)(1 + \psi(\mathbf{x}, t))}\right) \mathbf{x}. \tag{43}$$

Note that the part, $-1 + \xi|\mathbf{x}|$, of denominator in the gradient is eliminated by ψ defined in (42) because of the Taylor series expansion of Lambert function:

$$\mathcal{W}(x) = \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{n!} x^n. \tag{44}$$

In Table 7, we present the EOC of normal and regularized mean curvature flow equation by choosing $\mathbf{v} = 0$ and $\delta = \gamma = 1$ in (2). The numerical solution for the propose algorithm (34) is computed until $T = 1$ by using the size of time step $\Delta t = \Delta t_N^1$ in (38). The EOCs for the norms of errors E^2 and E^∞ are close to 2 and the EOCs for the norms of errors G^2 and G^∞ are larger than 1. In Table 8, the EOCs of the advective, normal, and regularized mean curvature flow equation are presented.

The numerical solution is computed by choosing $\delta = 0.1$, $\gamma = 1$, and the rotational velocity

$$\mathbf{v}(\mathbf{x}) = (\pi x_2, -\pi x_1, 0) \tag{45}$$

in (2) with the initial condition, $\phi_0(\mathbf{x}) = |\mathbf{x} - \mathbf{x}_0|$, where $\mathbf{x}_0 = (-0.625, 0, 0)$. The size of time step $\Delta t = \Delta t_N^1$ in (38) is used until $T = 1$. The exact solution and its gradient are obtained using (42) and (43) with a rotational transformation. The EOCs for the norm of error E^2 are close to 2 and the EOC for the norm of error E^∞ is close to 1.3. The EOC for the norms of errors G^2 and G^∞ are close to 1.

4.4 Qualitative comparisons

Two examples are presented to qualitatively compare numerical solutions of general level-set equations in (2). The first example is chosen in order to show a behavior of the solution as an evolution of the zero level set of a signed distance function. We choose the initial surface,

$$\mathcal{S} = \bigcup_{i=1}^{3} \mathcal{S}_i, \quad \mathcal{S}_i \equiv \left\{ \mathbf{x} \in \Sigma_i : \max_{j \in \{1,2,3\}} \{|\chi_j^i x_j|\} = 1 \right\}, \tag{46}$$

where $\Sigma_i = \left\{ \mathbf{x} \in \mathbb{R}^3 : |x_i| \geq 1 \right\}$, $\boldsymbol{\chi}^1 = (0.5, 1, 1)$, $\boldsymbol{\chi}^2 = (1, 0.5, 1)$, and $\boldsymbol{\chi}^3 = (1, 1, 0.5)$. The initial function ϕ_0 is analytically obtained by a signed distance function whose zero level set is the initial surface $\mathcal{S}$. We use a negative and positive distance inside and outside of $\mathcal{S}$, respectively. This particular choice of the sign of the initial condition in (2) brings an expansion or shrinkage of the zero level surface along the surface normal when δ is positive or negative, respectively. In Fig. 4, the rotational velocity $5\mathbf{v}$ where $\mathbf{v}$ is given in (45) and $\gamma = 1$ are fixed for each example, and we change δ in order to see an effect on the evolution of zero level set. The numerical solutions are computed by using the proposed algorithm (34) with the size of time step $\Delta t = 10^{-3}$ on the computational domain Ω_1 of level 2 in Table 1. Note that a zero Neumann boundary condition is applied because of unknown exact solution. From the top left to the bottom right in Fig. 4, we present three results by using $\delta = 0.1$(left), $\delta = 1$(middle), and $\delta = 5$(right) on each picture. Since the mean curvature flow forces the evolving surface to converge to a point in a finite time, the zero level surface (46) is going to vanish at a point. However, if we use a large expansion speed δ in (2), the initial surface can expand and does not vanish at a point, which can be observed in the rightmost case of each picture in Fig. 4.

In the second example, we use the Dragon surface from the Stanford 3D scanning repository[1] in order to qualitatively compare numerical results of proposed algorithm (30) to solve the regularized mean curvature flow equation (4) on hexahedral and polyhedral meshes. For the hexahedral mesh, the box domain $\Omega_3 = [-0.101, 0.111] \times [-0.061, 0.091] \times [-0.049, 0.053]$ is discretized by $207 \times 150 \times 100$ cells, which

[1] http://graphics.stanford.edu/data/3Dscanrep.

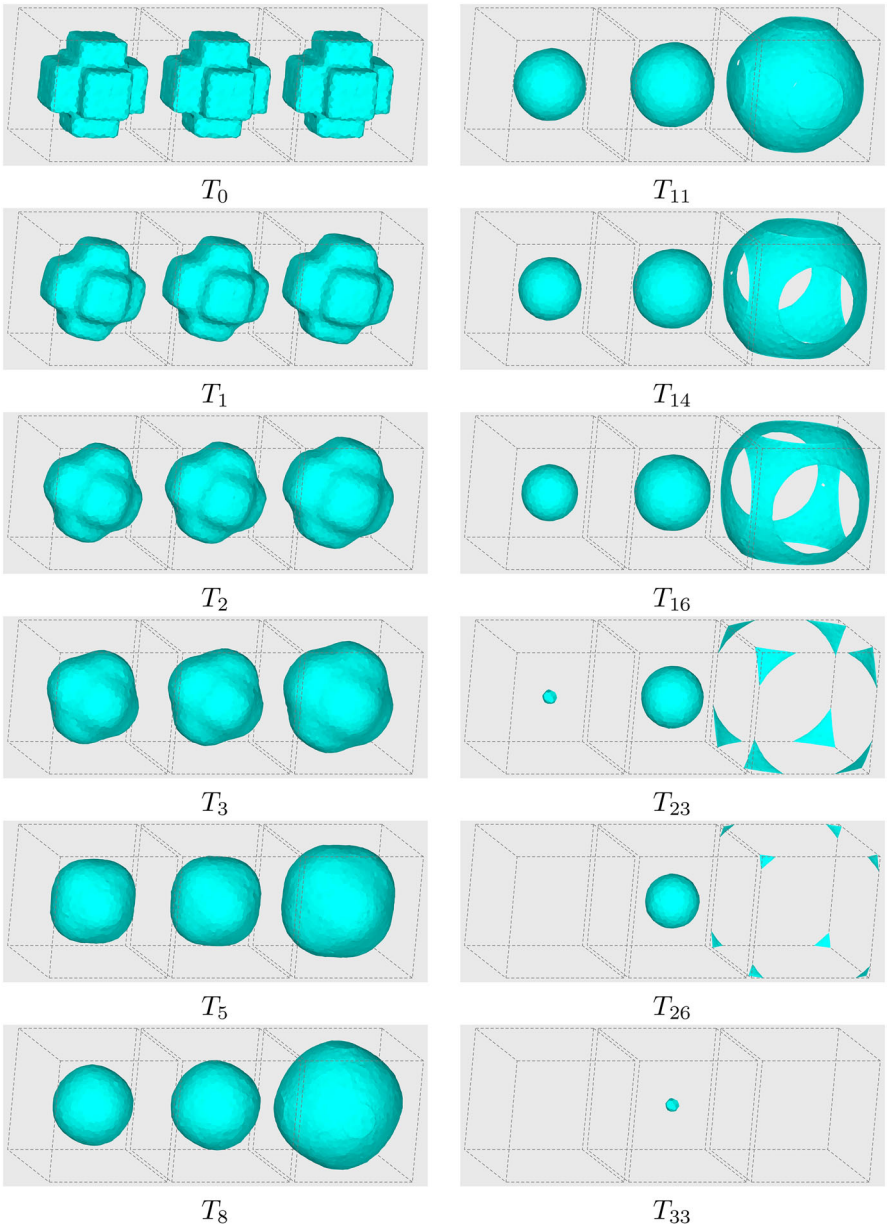

Fig. 4 Two columns present sets of triplet figures for three different δ evolving in time; $\delta = 0.1$ (left), $\delta = 1$ (middle), and $\delta = 5$ (right) and with the fixed rotational velocity $5\mathbf{v}$ in (50) and $\gamma = 1$. The size of time step is $\Delta t = 10^{-3}$. From top left to bottom right, numerical results at $T_i \equiv \Delta t + i \cdot 10^{-2}$ are presented. The evolution of zero level set shows an effect of different normal speeds in the general level set equation

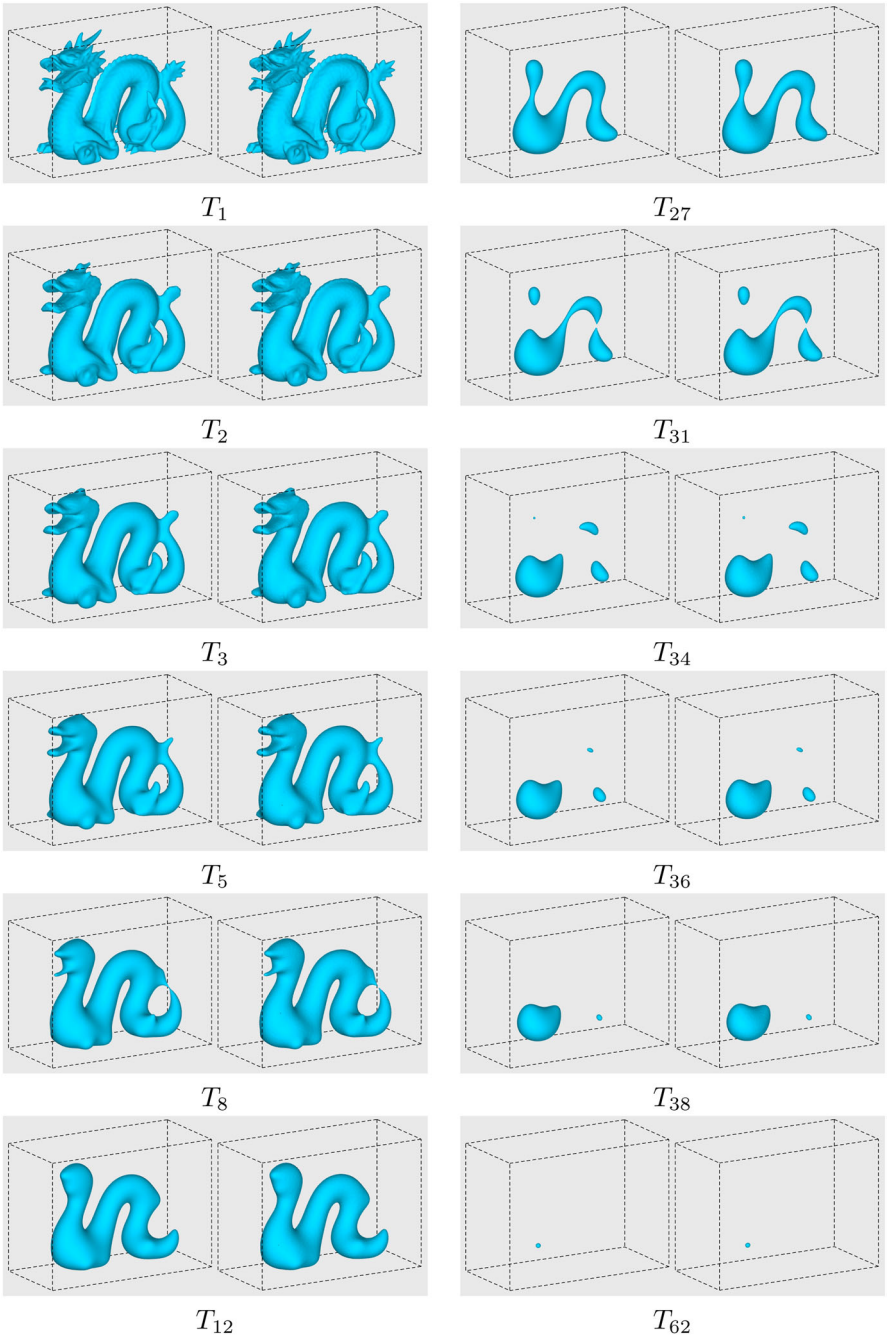

Fig. 5 In each picture, the left and right sides illustrate an evolution of the numerical solution of regularized mean curvature flow on the hexahedral and the polyhedral mesh of box domain Ω_3, respectively. From the top left to the bottom right, the numerical results at $T_i \equiv (i-1)\Delta t$ with $\Delta t = 5 \cdot 10^{-6}$ are presented

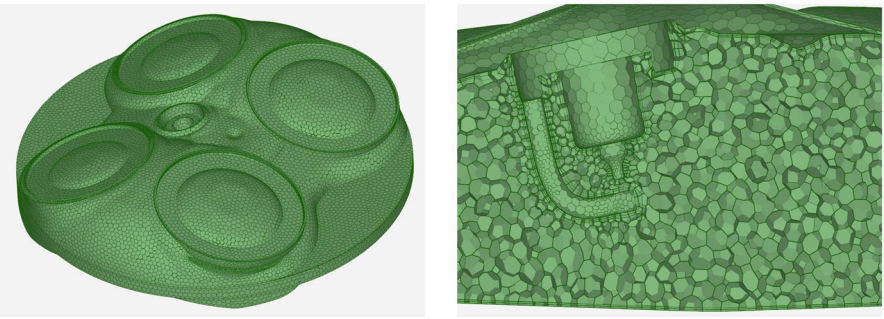

Fig. 6 A top view of a production GDI engine cylinder (left) and polyhedral cells in the mesh close to a part of spark ignition plug (right) are illustrated

results in 3,105,000 hexahedral cells and the average size of cells $h = 1.02 \cdot 10^{-3}$. For the polyhedral mesh, 2,451,033 polyhedral cells are generated resulting in $h = 1.49 \cdot 10^{-3}$. The size of time step $\Delta t = 5 \cdot 10^{-6}$ is used in the nonlinear Crank–Nicolson method (30). The reason to choose a relatively small size of the time step is to illustrate the details of topological changes in evolved surfaces. The initial condition is a signed distance function to the Dragon surface. The evolution of zero level set is illustrated in Fig. 5. Since an exact solution is unknown, we apply a zero Neumann boundary condition. The topological change looks indistinguishable for both types of the mesh, and the location and the time of vanishing point in the regularized mean curvature flow for two meshes are very similar.

4.5 *G*-equation

In real application, we add a constant parameter $C > 0$ to control overall speed of flame surface from the G-equation (1):

$$\bar{\rho}\frac{\partial \tilde{G}}{\partial t} + \bar{\rho}\tilde{\mathbf{v}} \cdot \nabla \tilde{G} = C\bar{\rho}_u s_T^0 |\nabla \tilde{G}| - \bar{\rho} D_t \tilde{\kappa} |\nabla \tilde{G}|. \tag{47}$$

Note that the zero level set of $\tilde{G}$ is the Favre mean turbulent flame surface. When the constant $C > 1$ (or $C < 1$), the burning speed is faster (or slower) than a given turbulent burning velocity s_T^0.

The governing Eq. (2) with $\epsilon = 0$ contains all terms in (47) to propagate the thin flame surface. In Fig. 6, a production GDI engine is discretized by a polyhedral mesh and a detailed view of the inside mesh close to a spark ignition plug is illustrated. A size of a cell is hugely changed because of complicated shape of boundary and two boundary layers. In Fig. 7, the parameter values, $\eta = 10^{-8}$ and $\epsilon = h$, are used and a propagation of the Favre mean turbulent flame surface is presented by red surfaces which are the zero level set of $\tilde{G}$ in (47). From the top left to the bottom right, the bottom of computational domain is moved downwards in order to properly simulate a movement of piston in a combustion engine. An ignition procedure is started from the crank angle (CA) 705°.

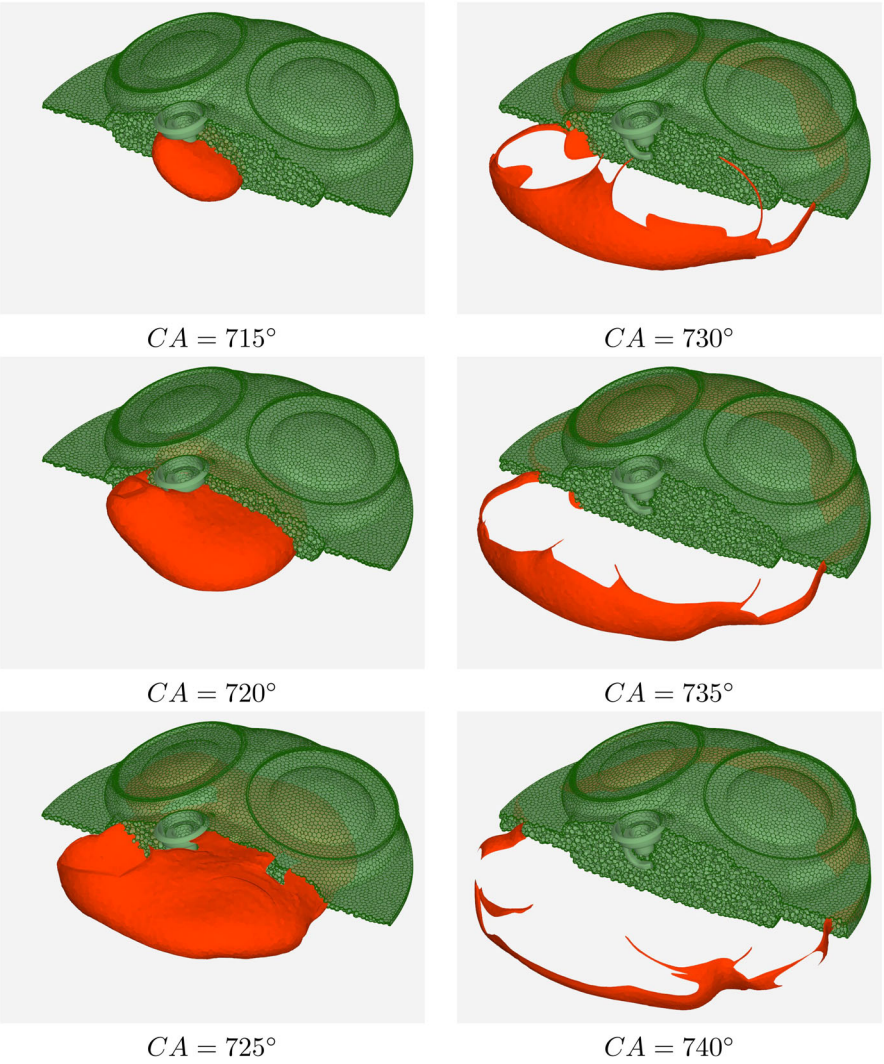

Fig. 7 Red surfaces present a propagation of the Favre mean turbulent flame surface, that is, the zero level set of $\tilde{G}$ in (47). The flame surface is ignited at $CA = 705°$

One can observe that some topological changes of flame structures are computed without any instabilities. A detail explanation of obtaining all burning velocities to evolve the flame surface is unfortunately beyond a scope of this paper. To understand a whole premixed combustion simulation from G-equation approach, we refer the interested readers to complete references [3, 42]. In the Fig. 8, a mean pressure graph (Pa) versus CA (deg) is presented. The dotted curve represents a measurement data and the solid red curve is obtained from the simulation result of G-equation combustion model in AVL FIRE™ with the parameter adjustment, that is, $C = 0.685$. The peaks of highest mean pressure and the maximizers of mean pressure are very close enough

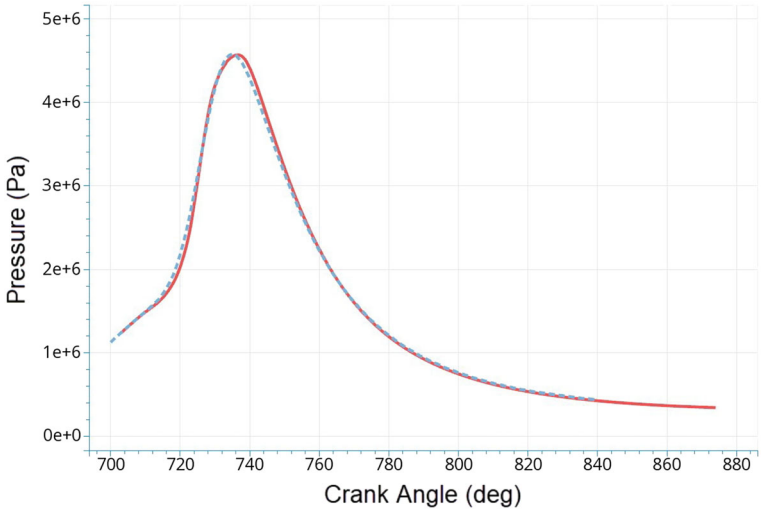

Fig. 8 A mean pressure graph (Pa) versus CA (deg) in the GDI engine cylinder in Fig. 6 is presented. The dotted blue curve is a measurement data and the solid red curve is the simulation result with $C = 0.685$ in AVL FIRE™

that indicates that the *G*-equation combustion model properly simulates a measurement of GDI engine.

5 Conclusion

In this paper, we propose the second-order accurate finite volume method to numerically solve the general level-set equation on polyhedral meshes. The choice of degrees of freedom is restricted to the centers of cells in order to achieve a low computational cost. The main focus is to design a numerical algorithm for the regularized mean curvature flow equation which can be consistently combined with the previous numerical algorithm for the advective or normal flow equations. In the case of regularized mean curvature flow equation, concerning a spatial discretization, the flux-balanced approximation and the orthogonal splitting are used at the faces between two polyhedral cells with the explicit reconstruction of gradients using the weighted least-squares minimization. For a temporal discretization, we propose the nonlinear Crank–Nicolson method with the deferred correction method in order to obtain an efficient algorithm for parallel computations using 1-ring neighborhood structure in decomposed domains. The size of time step is proportional to an average size of cells, which brings a consistent combination with the advective and normal flow equations in order to obtain the second-order convergence for level-set equations. The proposed method is implemented in AVL FIRE™ and tested on several examples to check the experimental order of convergence or to show some qualitative properties.

Acknowledgements The work was supported by grants VEGA 1/0314/23, VEGA 1/0436/20, and APVV-19-0460. This project No. 2140/01/01 has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 945478.

Data Availability The data that support the findings of this study are available on request from the corresponding author.

Appendix A: Iterative IIOE method

In order to complete the finite volume method to solve (2), we briefly review the iterative inflow-implicit outflow-explicit method (IIOE) [24, 25] of discretizing the advective and normal flow equations which appear in the second and third term of (2):

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0. \tag{A1}$$

The following two forms of the velocity function $\mathbf{u}$ are considered:

$$\mathbf{u} = \mathbf{v}(\mathbf{x}, t) \quad \text{or} \quad \mathbf{u} = \delta \frac{\nabla \phi}{|\nabla \phi|}, \quad (\mathbf{x}, t) \in \Omega \times [0, T], \tag{A2}$$

that describes the advective and normal flow, respectively, for an evolution of surface described by a zero level set of ϕ. Without a loss of generality, we choose the spatial variable $\delta = \delta(\mathbf{x})$ as the constant $\delta = 1$ in (4) for the rest of derivation of the method.

Firstly, we introduce a spatial discretization, and, secondly, we discuss a temporal discretization. Let us denote the set of indices $\bar{\mathcal{F}}_p$ and $\bar{\mathcal{B}}_p$ of triangles obtained by a tessellation of internal and boundary faces e_f, $f \in \mathcal{F}_p$ and e_b, $b \in \mathcal{B}_p$, respectively; see more details how to construct the tessellation in [25]. For a triangle index $i \in \bar{\mathcal{F}}_p$, $\bar{e}_i$ is the triangle as a subset of some face e_f, $f \in \mathcal{F}_p$. Furthermore, $\bar{\mathbf{n}}_{pi}$ is the outward normal vector to the triangle whose length is the area of $\bar{e}_i$, and $\bar{\mathbf{x}}_i$ is the center of the triangle. Using the Gauss divergence theorem on (A1), the spatial discretization is obtained:

$$\int_{\Omega_p} \frac{\partial \phi}{\partial t} + \sum_{i \in \bar{\mathcal{F}}_p \cup \bar{\mathcal{B}}_p} a_{pi} \left(\phi_{pi} - \phi_p \right) = 0, \quad a_{pi} \equiv \int_{\bar{e}_i} \mathbf{u} \cdot \bar{\mathbf{n}}_{pi} \approx \mathbf{u} \left(\bar{\mathbf{x}}_i, t \right) \cdot \bar{\mathbf{n}}_{pi}, \tag{A3}$$

where ϕ_{pi} is an approximated value defined at the center of the triangle $\bar{\mathbf{x}}_i$. Note that the flux a_{pi} is computed using Gaussian quadrature of degree 1 which insures the exactness of integration as polynomials of degree 1.

A crucial point in (A3) is how to approximate the value ϕ_{pi}. In order to obtain a second-order accurate upwind scheme, we prepare some values at vertices and gradients at centers of tessellated triangles in the following procedure. A value $\phi(\mathbf{x}_v)$ at an internal vertex is computed by an inverse distance weighted average from adjacent cells. For the fixed vertex $\mathbf{x}_v$, let us denote a set of indices of cells containing $\mathbf{x}_v$ as a vertex, $\mathcal{N}_v \equiv \left\{ p \in \mathcal{I} : \mathbf{x}_v \in \partial \Omega_p \right\}$. Then the value at internal vertex is obtained by

the inverse distance weighted average of the first order Taylor polynomial:

$$\phi(\mathbf{x}_v) = \frac{\sum_{p\in\mathcal{N}_v} |\mathbf{d}_{pv}|^{-1}(\phi(\mathbf{x}_p) + \nabla\phi(\mathbf{x}_p)\cdot\mathbf{d}_{pv})}{\sum_{p\in\mathcal{N}_v}|\mathbf{d}_{pv}|^{-1}}.$$

Note that a value at boundary vertex is directly assigned by Dirichlet boundary condition. From the obtained vertex values, a gradient at a center of tessellated triangle is computed. For an internal triangle $\bar{e}_i \subset e_f$, $f \in \mathcal{F}_p$, there exists $q \in \mathcal{I}$ such that $e_f \subset \partial\Omega_p \cap \partial\Omega_q$. Then, we consider two tetrahedrons whose apices are $\mathbf{x}_p$ and $\mathbf{x}_q$ and they have the common base $\bar{e}_i$. We denote the set of all vertices of the tetrahedrons as $\mathcal{P}_i$. Then, the gradient $\bar{\boldsymbol{\beta}}_i$ at the center of $\bar{e}_i$ is computed by the weighted least-squares minimization:

$$(\bar{\alpha}_i, \bar{\boldsymbol{\beta}}_i) = \underset{(a_i,\mathbf{b}_i)\in\mathbb{R}^4}{\arg\min} \sum_{\mathbf{y}\in\mathcal{P}_i} w_i(\mathbf{y})\left|a_i + \mathbf{b}_i\cdot(\mathbf{x}-\bar{\mathbf{x}}_i) - \phi(\mathbf{x})\right|^2, \tag{A4}$$

where $\bar{\mathbf{x}}_i$ is the center of $\bar{e}_i$ the weight function is defined by $w_i(\mathbf{y}) = |\mathbf{y}-\bar{\mathbf{x}}_i|^{-2}$. The formula (A4) is a generalization of the diamond-cell method described for a regular structured hexahedron cell in [20]. Now, we define a so-called average-based gradient [19] as the inverse distance weighted average of gradients:

$$\mathcal{D}_p\phi = \frac{\sum_{i\in\bar{\mathcal{F}}_p}|\mathbf{d}_{pi}|^{-1}\bar{\boldsymbol{\beta}}_i}{\sum_{i\in\bar{\mathcal{F}}_p}|\mathbf{d}_{pi}|^{-1}}, \quad \mathbf{d}_{pi} = \mathbf{x}_p - \bar{\mathbf{x}}_i. \tag{A5}$$

Finally, we compute the value ϕ_{pi} at an internal triangle $\bar{e}_i \subset e_f$, $f \in \mathcal{F}_p$, in (A3) using the average-based gradient and the upwind principle:

$$\phi_{pi} = \begin{cases} \phi_p + \mathcal{D}_p\phi\cdot(\bar{\mathbf{x}}_i - \mathbf{x}_p) & \text{if } a_{pi} \geq 0, \\ \phi_q + \mathcal{D}_q\phi\cdot(\bar{\mathbf{x}}_i - \mathbf{x}_q) & \text{if } a_{pi} < 0, \end{cases} \tag{A6}$$

where the neighbor cell index $q \in \mathcal{I}$ is such that $e_f \subset \partial\Omega_p \cap \partial\Omega_q$. The value ϕ_{pi} at a boundary triangle $\bar{e}_i \subset e_b$, $b \in \mathcal{B}_p$, in (A3) is computed by

$$\phi_{pi} = \begin{cases} \phi_p + \mathcal{D}_p\phi\cdot(\bar{\mathbf{x}}_i - \mathbf{x}_p) & \text{if } a_{pi} \geq 0, \\ \phi_{bi} & \text{if } a_{pi} < 0, \end{cases} \tag{A7}$$

where $\phi_{bi} \equiv \phi_b(\bar{\mathbf{x}}_i)$ from Dirichlet boundary condition. We finally obtain the spatial discretization from (A6) and (A7):

$$\begin{aligned} \int_{\Omega_p}\frac{\partial\phi}{\partial t} = &-\sum_{i\in\bar{\mathcal{F}}_p^-}\left(\phi_q + \mathcal{D}_q\phi\cdot\mathbf{d}_{qi} - \phi_p\right)a_{pi} - \sum_{i\in\bar{\mathcal{F}}_p^+}\left(\mathcal{D}_p\phi\cdot\mathbf{d}_{pi}\right)a_{pi} \\ &-\sum_{i\in\bar{\mathcal{B}}_p^-}\left(\phi_{bi} - \phi_p\right)a_{pi} - \sum_{i\in\bar{\mathcal{B}}_p^+}\left(\mathcal{D}_p\phi\cdot\mathbf{d}_{pi}\right)a_{pi}, \end{aligned} \tag{A8}$$

where $\bar{\mathcal{F}}_p^-$ and $\bar{\mathcal{B}}_p^-$ are the subsets of $\bar{\mathcal{F}}_p$ and $\bar{\mathcal{B}}_p$ with $a_{pi} < 0$, respectively, and $\bar{\mathcal{F}}_p^+ \equiv \bar{\mathcal{F}}_p \backslash \bar{\mathcal{F}}_p^-$, $\bar{\mathcal{B}}_p^+ \equiv \bar{\mathcal{B}}_p \backslash \bar{\mathcal{B}}_p^-$.

Concerning the time discretization, we apply the deferred correction method because of the same reasons as discussed in Sect. 3. Following the analogous notations as in previous sections, the IIOE method inspired by [20–23] is defined by using an implicit and explicit time discretization of terms in (A8) on an inflow and outflow triangle, respectively:

$$\begin{aligned}\frac{|\Omega_p|}{\Delta t}\left(\phi_p^{n,k} - \phi_p^{n-1}\right) = &- \sum_{i \in \bar{\mathcal{F}}_p^-} \left(\phi_q^{n,k} + \mathcal{D}_q \phi^{n,k-1} \cdot \mathbf{d}_{qi} - \phi_p^{n,k}\right) a_{pi}^{n-1} \\ &- \sum_{i \in \bar{\mathcal{B}}_p^-} \left(\phi_{bi}^{n} - \phi_p^{n,k}\right) a_{pi}^{n-1} - \sum_{i \in \bar{\mathcal{B}}_p^+ \cup \bar{\mathcal{F}}_p^+} \left(\mathcal{D}_p \phi^{n-1} \cdot \mathbf{d}_{pi}\right) a_{pi}^{n-1},\end{aligned} \tag{A9}$$

where $\phi^{n,0} = \phi^{n-1}$. Note that the average-based gradient $\mathcal{D}_p \phi^{n,k-1}$ is computed by values at centers of cells from $\phi^{n,k-1}$ and values at centers of boundary faces from the nth time level.

References

1. Williams, F.A.: Turbulent combustion. In: Buckmaster, J.D. (ed.) The Mathematics in Combustion, pp. 97–131. SIAM, Philadelphia (1985)
2. Peters, N.: The turbulent burning velocity for large-scale and small-scale turbulence. J. Fluid Mech. **384**, 107–132 (1999)
3. Peters, N.: Turbulent Combustion. Cambridge Monographs on Mechanics. Cambridge University Press, Cambridge (2000)
4. Kolář, M., Kobayashi, S., Uegata, Y., Yazaki, S., Beneš, M.: Analysis of Kuramoto-Sivashinsky model of flame/smoldering front by means of curvature driven flow. In: Vermolen, F.J., Vuik, C. (eds.) Numerical Mathematics and Advanced Applications ENUMATH 2019. Lecture Notes in Computational Science and Engineering, vol. 139, pp. 615–624. Springer, Cham (2021)
5. Goto, M., Kuwana, K., Kushida, G., Yazaki, S.: Experimental and theoretical study on near-floor flame spread along a thin solid. Proc. Combust. Inst. **37**, 3783–3791 (2019)
6. Evans, L.C., Spruck, J.: Motion of level sets by mean curvature. I. J. Differ. Geom. **33**, 635–681 (1991)
7. Osher, S., Fedkiw, R.: Level Set Methods and Dynamic Implicit Surfaces. Springer, Berlin (2000)
8. Sethian, J.A.: Level Set Methods and Fast Marching Methods, Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materical Science. Cambridge University Press, New York (1999)
9. Gibou, F., Fedkiw, R., Osher, S.: A review of level-set methods and some recent applications. J. Comput. Phys. **353**, 82–109 (2018)
10. Perič, M.: Flow simulation using control volumes of arbitrary polyhedral shape. In: ERCOFTAC Bulletin, vol. 62, pp. 25–29 (2004)
11. Eymard, R., Gallouët, T., Herbin, R.: Finite volume methods. Handb. Numer. Anal. **7**, 713–1018 (2000)
12. Lipnikov, K., Manzini, G., Shashkov, M.: Mimetic finite difference method. J. Comput. Phys. **257**, 1163–1227 (2014)
13. da Veiga, L.B., Breszzi, F., Cangiani, A., Manzini, G., Marini, L.D., Russo, A.: Basic principles of virtual element methods. Math. Models Methods Appl. Sci. **23**, 199–214 (2013)
14. Coudière, Y., Manzini, G.: The discrete duality finite volume method for convection-diffusion problems. SIAM J. Numer. Anal. **47**, 4163–4192 (2010)

15. Barth, T.J., Sethian, J.: Numerical schemes for the Hamilton–Jacobi and level set equations on triangulated domains. J. Comput. Phys. **145**(1), 1–40 (1998)
16. Kees, C.E., Akkerman, I., Farthing, M.W., Bazilevs, Y.: A conservative level set method suitable for variable-order approximations and unstructured meshes. J. Comput. Phys. **230**(12), 4536–4558 (2011)
17. Lipnikov, K., Morgan, N.: A high-order discontinuous Galerkin method for level set problems on polygonal meshes. J. Comput. Phys. **397**, 108834 (2019)
18. Frolkovič, P., Mikula, K.: High-resolution flux-based level set method. SIAM J. Sci. Comput. **29**, 579–597 (2007)
19. Hahn, J., Mikula, K., Frolkovič, P., Basara, B.: Inflow-based gradient finite volume method for a propagation in a normal direction in a polyhedron mesh. J. Sci. Comput. **72**, 442–465 (2017)
20. Mikula, K., Ohlberger, M.: A new level set method for motion in normal direction based on a semi-implicit forward–backward diffusion approach. SIAM J. Sci. Comput. **32**, 1527–1544 (2010)
21. Mikula, K., Ohlberger, M.: Inflow-implicit/outflow-explicit scheme for solving advection equations. In: Fort, J. (ed.) Finite Volumes for Complex Applications VI—Problems and Perspectives. Springer Proceedings in Mathematics 4, vol. 1, pp. 683–691. Springer, Berlin (2011)
22. Mikula, K., Ohlberger, M., Urbán, J.: Inflow-implicit/outflow-explicit finite volume methods for solving advection equations. Appl. Numer. Math. **85**, 16–37 (2014)
23. Frolkovič, P., Mikula, K., Urbán, J.: Semi-implicit finite volume level set method for advective motion of interfaces in normal direction. Appl. Numer. Math. **95**, 214–228 (2015)
24. Hahn, J., Mikula, K., Frolkovič, P., Basara, B.: Semi-implicit level set method with inflow-based gradient in a polyhedron mesh. In: Cancès, C., Omnes, P. (eds.) Finite Volumes for Complex Applications VIII - Hyperbolic, Elliptic and Parabolic Problems, pp. 81–89 (2017). Springer International Publishing
25. Hahn, J., Mikula, K., Frolkovič, P., Medl'a, M., Basara, B.: Iterative inflow-implicit outflow-explicit finite volume scheme for level-set equations on polyhedron meshes. Comput. Math. Appl. **77**, 1639–1654 (2019)
26. Deckelnick, K., Dziuk, G., Elliott, C.M.: Computation of geometric partial differential equations and mean curvature flow. Acta Numer. **14**, 139–232 (2005)
27. Walkington, N.J.: Algorithms for computing motion by mean curvature. SIAM J. Numer. Anal. **33**, 2215–2238 (1996)
28. Handlovičová, A., Mikula, K., Sgallari, F.: Semi-implicit complementary volume scheme for solving level set like equations in image processing and curve evolution. Numer. Math. **93**, 675–695 (2003)
29. Mikula, K., Sarti, A., Sgallari, F.: Co-volume level set method in subjective surface based medical image segmentation, Chap. 11. In: Suri, J.S., Wilson, D.L., Laxminarayan, S. (eds.) Handbook of Biomedical Image Analysis. International Topics in Biomedical Engineering. Springer, Boston, pp. 583–626 (2005). https://doi.org/10.1007/0-306-48551-6_11
30. Mikula, K., Sarti, A., Sgallari, F.: Co-volume method for Riemannian mean curvature flow in subjective surfaces multiscale segmentation. Comput. Vis. Sci. **9**, 23–31 (2006)
31. Corsaro, S., Mikula, K., Sarti, A., Sgallari, F.: Semi-implicit covolume method in 3d image segmentation. SIAM J. Sci. Comput. **28**, 2248–2265 (2006)
32. Handlovičová, A., Mikula, K.: Stability and consistency of the semi-implicit co-volume scheme for regularized mean curvature flow equation in level set formulation. Appl. Math. **53**, 105–129 (2008)
33. Eymard, R., Handlovičová, A., Mikula, K.: Study of a finite volume scheme for the regularized mean curvature flow level set equation. IMA J. Numer. Anal. **31**, 813–846 (2011)
34. Eymard, R., Handlovičová, A., Mikula, K.: Approximation of nonlinear parabolic equations using a family of conformal and non-conformal schemes. Commun. Pure Appl. Anal. **11**, 147–172 (2012)
35. Frolkovič, P., Mikula, K., Hahn, J., Martin, D., Basara, B.: Flux balanced approximation with least-squares gradient for diffusion equation on polyhedral mesh. Discrete Contin. Dyn. Syst. S (2020). https://doi.org/10.3934/dcdss.2020350
36. Hahn, J., Mikula, K., Frolkovič, P., Balažovjech, M., Basara, B.: Cell-centered finite volume method for regularized mean curvature flow on polyhedral meshes. In: Klöfkorn, R., Keilegavlen, E., Radu, F.A., Fuhrmann, J. (eds.) Finite Volumes for Complex Applications IX—Methods, Theoretical Aspects, Examples. Springer International Publishing, pp. 755–763 (2020). https://doi.org/10.1007/978-3-030-43651-3_72
37. Demirdžić, I.: On the discretization of the diffusion term in finite-volume continuum mechanics. Numer. Heat Transfer B Fundam. **68**, 1–10 (2015)

38. Niceno, B.: A three dimensional finite volume method for incompressible Navier–Stokes equations on unstructured staggered grids. In: European Conference on Computational Fluid Dynamics ECCOMAS CFD 2006, Egmond Aan Zee, The Netherlands, Paper 196 (2006)
39. Balažovjech, M., Mikula, K.: A higher order scheme for a tangentially stabilized plane curve shortening flow with a driving force. SIAM J. Sci. Comput. **33**(5), 2277–2294 (2011)
40. Balažovjech, M., Frolkovič, P., Frolkovič, R., Mikula, K.: Semi-implicit second order accurate finite volume method for advection-diffusion level set equation. In: Finite Volumes for Complex Applications VII-Elliptic, Parabolic and Hyperbolic Problems. Springer, Berlin, pp. 479–487 (2014)
41. Böhmer, K., Hemker, P.W., Stetter, H.J.: The defect correction approach. In: Defect Correction Methods. Springer, pp. 1–32 (1984)
42. Pitsch, H.: A consistent level set formulation for large-eddy simulation of premixed turbulent combustion. Combust. Flame **143**, 587–598 (2005)