


# Finite volume method with the Sonner boundary condition for computing the signed distance function on polyhedral meshes

Jooyoung Hahn<sup>1</sup>  | Karol Mikula<sup>2</sup> | Peter Frolkovič<sup>2</sup> | Branislav Basara<sup>1</sup>

<sup>1</sup>Advanced Simulation Technologies, AVL List GmbH, Hans-List-Platz 1, Graz, Austria

<sup>2</sup>Faculty of Civil Engineering, Department of Mathematics and Descriptive Geometry, Slovak University of Technology, Radlinského 11, Bratislava, Slovak Republic

## Correspondence

Jooyoung Hahn, Advanced Simulation Technologies, AVL List GmbH, Hans-List-Platz 1, Graz 8020, Austria.  
Emails: Jooyoung.Hahn@avl.com; JooyoungHahn@gmail.com

## Funding information

APVV, Grant/Award Number: 19-0460; VEGA, Grant/Award Numbers: 1/0436/20, 1/0709/19

## Abstract

A cell-centered finite volume method with the Sonner boundary condition is proposed to compute the signed distance function from a given surface in general three-dimensional (3D) computational domains discretized by polyhedral cells. The governing equation is the bidirectional time-relaxed eikonal equation and the proposed numerical method is based on the semi-implicit inflow-implicit and outflow-explicit scheme. Numerical experiments confirm the second order accuracy in  $L^1$  and  $L^\infty$ -norms for chosen examples with smooth solutions. The inclusion of the Sonner boundary condition has proven necessary for numerical solutions to reach the viscosity solution of the eikonal equation starting from various initial conditions in general 3D domains.

## KEYWORDS

cell-centered finite volume method, eikonal equation, no-inflow boundary condition, polyhedral meshes, signed distance function, Sonner boundary condition

## 1 | INTRODUCTION

A robust numerical algorithm to compute the signed distance function from a given surface on three-dimensional (3D) polyhedral meshes is an important component to use the level set method<sup>1-3</sup> in many industrial applications. In numerical combustion, the signed distance function whose zero level set represents a thin flame surface is used to model the  $G$ -equation<sup>4</sup> and evolve the mean flame surface in a turbulent flow. It also used to accurately model the flame-wall interaction and quenching<sup>5</sup> or the end-gas autoignition for knock prediction.<sup>6</sup> In computational fluid dynamics (CFD), the level set method related to two-phase flow problems<sup>7</sup> or front capturing applications needs to keep the property of the signed distance from the interface. The wall distance function has been a key feature of turbulent modeling<sup>8-11</sup> in CFD. There is an extensive amount of literature that discusses a further usage of signed distance functions; surface reconstruction in computer graphics,<sup>12,13</sup> meshing and medial axis transformation,<sup>14</sup> boundary and spatial-partitioning representations for the navigation of robots,<sup>15</sup> heterogeneous material modeling,<sup>16</sup> automatic removal of geometrical shape feature in computer-aided design (CAD).<sup>17</sup>

In the view of numerical algorithms to compute the signed distance function, each algorithm has been developed by different purpose and perspective to use the distance properties in its main application. It diversifies the development of numerical algorithms into various methods to focus on numerical properties such as computational cost, efficiency, accuracy, robustness, and parallelization. In this article, we would like to propose the numerical algorithm to focus on its robustness for industrial applications with reasonable requirements. Due to the size and complexity of the shape of the

computational domain, polyhedral meshes and parallel computing are mostly preferred to capture the reality corresponding the simulation; see more details in Reference 18. In order to implement a new numerical algorithm straightforwardly into the prevalent CFD numerical codes, a vertex-centered or cell-centered finite volume method is a preferable choice.

Numerical methods to compute the signed distance function from a given surface is identical to numerically find the viscosity solution of the eikonal equation:<sup>19</sup>

$$\begin{aligned} |\nabla\phi(\mathbf{x})| &= 1, \quad \mathbf{x} \in \Omega \\ \text{sgn}(\phi) &= \text{sgn}(\phi_0), \end{aligned} \quad (1)$$

where  $\text{sgn}(x)$  is a sign function on  $\mathbb{R}$ ,  $\Omega$  is the computational domain, and  $\phi_0(\mathbf{x})$  is a continuous function on  $\Omega$  whose zero level set is the given surface in  $\overline{\Omega}$ . Based on the formulation to compute the viscosity solution, an extensive amount of numerical methods can be categorized by two methods; partial differential equation (PDE)-based method and optimization-based method. The PDE-based methods are divided into time-independent Equation 1 and time-dependent eikonal equation.<sup>7</sup> We briefly review the advantages and disadvantages in the view of computing the signed distance function on 3D polyhedral meshes from the industrial examples.

The viscosity solution of the time-independent eikonal Equation 1, that is, signed distance function, is numerically solved by Dijkstra-like methods: the fast marching method (FMM)<sup>20-22</sup> and the iterative fast sweeping method (FSM)<sup>23-26</sup> on regular grids or triangular meshes. Using the property that the gradient lines of the viscosity solution coincide with the characteristics of (1), FMM has shown that the causality property holds and FSM captures the causality in Gauss-Seidel iterative methods; see more details for the comparison of FMM and FSM.<sup>27,28</sup> It brings very efficient and robust numerical algorithm to compute the viscosity solution on hexahedral or tetrahedral meshes. An extension of using FMM and FSM on polyhedral meshes of 3D computational domain with complex boundaries are not straightforwardly obtained.

The time-dependent eikonal equations have been presented in various applications: the reinitialization equation,<sup>7</sup> the time-dependent form for reaction-diffusion equation in spreading of excitation in cardiac tissues,<sup>29</sup> and the transport form of eikonal equation for medial axis transformation.<sup>14</sup> The time-dependent eikonal equation<sup>7</sup> is used to keep the signed distance property close to the interface for two-phase incompressible flow. Later, there has been a lot of literature to improve the accuracy and efficiency only concerning the vicinity of the interface; see more details in References 1-3. A time-dependent eikonal equation is accompanied by the diffusion term<sup>29</sup> and the first order upwind finite difference scheme with the explicit time step is used. The cell-vertex and the cell-centered finite volume methods are implemented in general CFD codes<sup>14</sup> to solve the time-dependent eikonal equation.<sup>30</sup> From the communication with the authors,<sup>14</sup> it is confirmed that the numerical scheme can be the second order of convergence in the case of capturing linear distance function. The method<sup>14</sup> is restricted to compute the wall distance function from the boundary of computational domain.

The eikonal equation is alternatively modeled by a diffusion operator and a control parameter; a screened Poisson equation,<sup>31</sup> regularized eikonal equation,<sup>8,32</sup>  $p$ -Laplacian equation.<sup>33</sup> Due to the rich literature of numerical algorithms for solving a Poisson equation on various domains, a smooth distance function can be stably obtained as long as a reasonable control parameter is used. When the parameter goes to its limit, the equations theoretically converge to the viscosity solution of the eikonal equation. However, numerically it is a challenging task to push the control parameter to the limit because the smoothing properties from diffusion are disappeared<sup>34</sup> and the condition number of the corresponding matrix is going to be large.

The optimization-based methods are introduced [35, Chapter 8] to compute the viscosity solution of the eikonal equation as well. The geodesic-in-heat method<sup>36,37</sup> shows the robust and efficient algorithm practically applied to compute the smooth distance function on a various geometric domains in computer graphics. The alternating direction method of multipliers (ADMM) used to solve  $p$ -Laplacian equation<sup>38</sup> for obtaining the wall distance function approximation, optimal transportation, and image enhancement. The ADMM-based scheme with the variable relaxation is also applied to the distance function approximation.<sup>39</sup> Since the mentioned optimization-based methods only need to solve the elliptic type PDE on the computational domain with few interactions, they have extremely low computational cost and the parallelization is straightforward. From the communication with the authors,<sup>39</sup> it is numerically verified that the ADMM for computing the wall distance function is the first order accurate in  $L^\infty$ -norm which is the same order of convergence obtained by the geodesic-in-heat method.<sup>36,37</sup> The high-order and efficient numerical algorithm to compute the signed distance function in the optimization-based methods is presented.<sup>40,41</sup> However, the convexity restriction of the initial condition  $\phi_0$  in (1) is too strong to be used on the complex shape of the computational domain.

In this article, we use the bidirectional time-relaxed (time-dependent) eikonal equation, called the bidirectional flow equation,<sup>42,43</sup> to numerically obtain the signed distance function from an evolving surface on a whole computational domain discretized by polyhedral meshes. A practical usage of level set method in industrial applications like combustion, multiphase flow, and the other front capturing problems requires computational domains of very general shapes. The robustness and accuracy of the numerical algorithm is more important than the computational cost as long as a parallel computing is reasonably possible. Additionally, we would like to cover the case that the given surface is a part of the boundary of the domain. A typical application of the mentioned case is path planning, visibility detection, optimal control, and shape-from-shading; see more details in References 1-3.

In the case of general shape of the computational domain, we show that the Soner boundary condition, called state-constraint condition<sup>44-47</sup> in the shape-from-shading, should be used to numerically approach the viscosity solution of the eikonal equation; see more details in Section 2. The incompleteness of measured data<sup>44</sup> is the main cause to consider the Soner boundary condition in shape-from-shading. Similarly, the lack of the visibility from the given surface on a nonconvex domain is the main reason to apply the same condition in the case of computing the signed distance function on a nonconvex domain. The Soner boundary condition is called as the no-inflow boundary condition.<sup>34</sup> The similar concept of the boundary condition is also considered in the reinitialization close to contact lines in CFD.<sup>48</sup>

The rest of article is presented as follows. In Section 2, we review the bidirectional flow equation and explain the necessity of using the Soner boundary condition. In Section 3, the modification of the previous cell-centered finite volume method<sup>42,49</sup> is proposed to use both Soner and Dirichlet boundary condition. In Section 4, various numerical experiments and numerical convergence order are illustrated. Finally, we conclude in Section 5.

## 2 | BIDIRECTIONAL TIME-RELAXED EIKONAL EQUATION

The bidirectional time-relaxed eikonal equation is used to obtain a signed distance function from a closed, bounded, and connected surface  $\Lambda \subset \mathbb{R}^3$ . The given surface  $\Lambda$  divides the computational domain  $\Omega \subset \mathbb{R}^3$  into two open sets  $\Omega^+$  and  $\Omega^-$ :

$$\Lambda = \partial\Omega^+ \cap \partial\Omega^-, \quad \overline{\Omega^+} \cup \overline{\Omega^-} = \overline{\Omega}, \quad \Omega^+ \cap \Omega^- = \emptyset. \quad (2)$$

Then, the equations called the bidirectional flow<sup>42,43</sup> propagate the given values on the surface  $\Lambda$  along the normal to  $\Lambda$  into two separated domains  $\Omega^+$  and  $\Omega^-$ :

$$\begin{aligned} \frac{\partial}{\partial t} \phi(\mathbf{x}, t) + |\nabla \phi(\mathbf{x}, t)| &= 1, & (\mathbf{x}, t) \in \Omega^+ \times [0, T], \\ \frac{\partial}{\partial t} \phi(\mathbf{x}, t) - |\nabla \phi(\mathbf{x}, t)| &= -1, & (\mathbf{x}, t) \in \Omega^- \times [0, T], \end{aligned} \quad (3)$$

where  $\phi(\mathbf{x}, t) = \phi_0(\mathbf{x})$  is given for  $\mathbf{x} \in \Lambda$ . When  $\phi_0 = 0$  on  $\Lambda$ , the steady state solution is the signed distance function from the surface  $\Lambda$  whose positive and negative distance values are in  $\Omega^+$  and  $\Omega^-$ , respectively. Note that the equations in (3) are formally same as the reinitialization equation:<sup>7</sup>

$$\frac{\partial}{\partial t} \phi(\mathbf{x}, t) + \text{sgn}(\phi_0(\mathbf{x}))(|\nabla \phi(\mathbf{x}, t)| - 1) = 0, \quad (\mathbf{x}, t) \in (\Omega \setminus \Lambda) \times [0, T], \quad (4)$$

where  $\phi_0(\mathbf{x})$  is a continuous function whose zero level set is the surface  $\Lambda$  and its value on  $\Omega^+$  and  $\Omega^-$  is positive and negative, respectively. The steady state viscosity solution<sup>50</sup> of (3) and (4) is also same as the viscosity solution of the time-independent eikonal Equation 1.<sup>19</sup>

From the PDE point of view, the time-relaxed and time-independent eikonal equations are well defined using only the boundary values on the surface  $\Lambda$ . However, from the numerical point of view, another crucial piece of information is required, namely, the boundary condition defined on the boundary of the computational domain. A possible choice is the no-flux boundary condition:

$$\mathbf{v}(\mathbf{x}) \cdot \nabla \phi(\mathbf{x}, t) = 0, \quad \mathbf{x} \in \partial\Omega, \quad (5)$$

where  $\mathbf{v}$  is the outward unit normal vector to the boundary  $\partial\Omega$ . Whenever there is a diffusion term added to the eikonal equation, for example, the equations used in References 8,29,31-34, the no-flux boundary condition (5) can be used.

Although a distortion of iso-surfaces of  $\phi$  on  $\partial\Omega$  is then unavoidable, it can be used for the cases when the region of the interest is the zero level set of  $\phi$  that is away from  $\partial\Omega$ .

In the case of computing the signed distance function on the whole domain, the no-flux boundary condition obviously cannot be satisfied on  $\partial\Omega$ . The appropriate condition in this case is the Soner boundary condition or the state-constraint condition:<sup>44-47</sup>

$$\mathbf{v}(\mathbf{x}) \cdot \nabla\phi(\mathbf{x}, t) \geq 0, \quad \mathbf{x} \in \partial\Omega. \quad (6)$$

The same condition is called the no-inflow boundary condition<sup>34</sup> which decides the admissible direction of the propagation on the boundary  $\partial\Omega$ . The Soner boundary condition is a natural consequence of the following relation:<sup>50</sup>

$$|\nabla\phi(\mathbf{x})| = 1 \Leftrightarrow \sup_{|\mathbf{a}| \leq 1} \{\nabla\phi(\mathbf{x}) \cdot \mathbf{a} - 1\}, \quad \mathbf{x} \in \overline{\Omega}. \quad (7)$$

The supremum in (7) is obtained by  $\mathbf{a} = \frac{\nabla\phi(\mathbf{x})}{|\nabla\phi(\mathbf{x})|}$  which is the characteristic direction of the eikonal equation.

The Soner boundary condition limits the search for a distance from points on the boundary to inside the computational domain. If it is violated, there must be an object out of the computational domain for which we want to measure the distance. However, such an object cannot be counted simply because it is not presented in the domain of interest.

Now, we present the bidirectional time-relaxed eikonal equation in a complete form including the boundary condition with the aim to compute the signed distance function from the surface  $\Lambda$ . Additionally, we also consider the Dirichlet boundary condition assigned on  $\sigma_D \subset \partial\Omega$  (only if it is available):

$$\frac{\partial}{\partial t}\phi(\mathbf{x}, t) + s(\mathbf{x})|\nabla\phi(\mathbf{x}, t)| = s(\mathbf{x}), \quad (\mathbf{x}, t) \in (\Omega \setminus \Lambda) \times [0, T], \quad (8)$$

where  $s(\mathbf{x}) = 1$  on  $\Omega^+$ ,  $s(\mathbf{x}) = -1$  on  $\Omega^-$ , and the boundary conditions are given on  $\Lambda$ ,  $\sigma_D$ , and  $\sigma_N = \partial\Omega \setminus \sigma_D$ , namely,

$$\phi(\mathbf{x}, t) = \phi_0(\mathbf{x}), \quad (\mathbf{x}, t) \in \Lambda \times [0, T], \quad (9)$$

$$\phi(\mathbf{x}, t) = \phi_D(\mathbf{x}), \quad (\mathbf{x}, t) \in \sigma_D \times [0, T], \quad (10)$$

$$\mathbf{v}(\mathbf{x}) \cdot \nabla\phi(\mathbf{x}, t) \geq 0, \quad (\mathbf{x}, t) \in \sigma_N \times [0, T]. \quad (11)$$

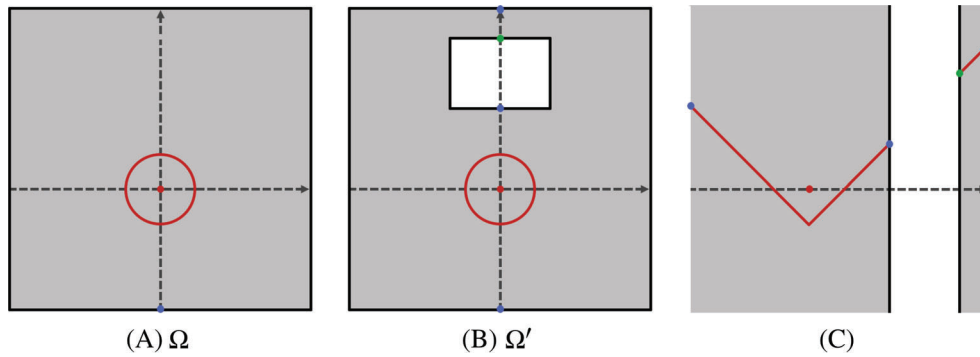
The initial condition is defined by

$$\phi(\mathbf{x}, 0) = \phi_I^\pm(\mathbf{x}), \quad \mathbf{x} \in \Omega^\pm, \quad (12)$$

where the given functions  $\phi_I^+$  and  $\phi_I^-$  shall fulfil

$$\phi_I^+(\mathbf{x}) > \max_{\mathbf{x} \in \Lambda} \phi_0(\mathbf{x}), \quad \mathbf{x} \in \Omega^+ \quad \text{and} \quad \phi_I^-(\mathbf{x}) < \min_{\mathbf{x} \in \Lambda} \phi_0(\mathbf{x}), \quad \mathbf{x} \in \Omega^-. \quad (13)$$

We introduce a simple two-dimensional (2D) example to illustrate that the Soner boundary condition guides a numerical solution of the time-relaxed eikonal equation to approach the viscosity solution of the eikonal Equation 1 and prohibits approaching a nonviscosity solution. It is obvious that the cone  $\psi(\mathbf{x}) = |\mathbf{x}| - r$  with the radius  $r > 0$  whose zero level set is the red circle in Figure 1A is the signed distance function to the circle with the radius  $r$  on a computational domain  $\Omega$  of the rectangular shape. In other words, the cone is the viscosity solution of the eikonal Equation 1 in  $\Omega$ . However, if the domain  $\Omega$  is modified by a hole as the domain  $\Omega'$  in Figure 1B, the cone is not the viscosity solution anymore in the domain  $\Omega'$ . That is, it is not the distance function from the red circle because the value  $\psi(\mathbf{x}_0)$  where  $\mathbf{x}_0$  is the green point in Figure 1B or C is not a distance from the red circle. Note that the cone is formally a steady state solution of (3) because  $|\nabla\psi(\mathbf{x})| = 1$ , so it is a nonviscosity solution of the eikonal equation in the domain  $\Omega'$ . Now, it is easy to check  $\mathbf{v}(\mathbf{x}_0) \cdot \nabla\psi(\mathbf{x}_0) < 0$  where  $\mathbf{x}_0$  is the green point in Figure 1B or C, which violates the Soner boundary condition. If a time-dependent numerical solution of (8) is computed with the Soner boundary condition, then the incorrect information,  $\mathbf{v}(\mathbf{x}) \cdot \nabla\phi(\mathbf{x}, t) < 0$ , on the boundary is avoided and the numerical solution will use the correct value from inside of the computational domain and the correct distance value will eventually reach to all boundary points at the steady state. The next section explains the main idea of enforcing the Soner boundary condition in the cell-centered finite volume method.



**FIGURE 1** (A) is a computational domain of the rectangular shape  $\Omega$ . (B) is a computational domain  $\Omega'$  (gray region) with a hole and a given surface  $\Lambda$  (red circle). (C) shows the graph of the cone along the y-axis in  $\Omega'$ , which is a nonviscosity solution of (3). It violates the Soner boundary condition at the green point and it is not the distance function from the red circle anymore on the domain  $\Omega'$

### 3 | FINITE VOLUME METHOD WITH SONER BOUNDARY CONDITION

This section firstly introduces the notations to understand the finite volume method on 3D polyhedron meshes. Secondly, applying the semi-implicit inflow-implicit and outflow-explicit (IIOE) method<sup>43,49</sup> to discretize the bidirectional flow (3), a cell-centered finite volume method is presented to explain the limitations of previous methods. In the end, we propose the algorithm to solve the bidirectional time-relaxed eikonal Equation 8 and to enforce the Soner boundary condition to overcome the mentioned limitations.

Let us discretize the computational domain,

$$\bar{\Omega} = \bigcup_{p \in \mathcal{I}} \bar{\Omega}_p,$$

where  $\Omega_p$  are nonoverlapped polyhedral cells with nonzero volume  $|\Omega_p| \neq 0$  and  $\mathcal{I}$  is the set of the indices of cells. For a cell  $\Omega_p$ , the set  $\mathcal{N}_p \subset \mathcal{I}$  is the collection of indices to indicate the neighbor cells whose boundary intersection with  $\partial\Omega_p$  has a nonzero area. That is, if  $q \in \mathcal{N}_p$ , there is a common face  $e_f \subset \partial\Omega_p \cap \partial\Omega_q$  whose area is nonzero,  $|e_f| \neq 0$ . Such a face is called the internal face and the index set  $\mathcal{F}$  contains indices of all internal faces. Similarly, the index set  $\mathcal{B}$  of all boundary faces is defined by indicating a boundary face  $e_b \subset \partial\Omega_p \cap \partial\Omega$  for  $p \in \mathcal{I}$  with a nonzero area  $|e_b| \neq 0$ . Note that an internal or boundary face of the polyhedron cell can be a polygonal shape. In 3D, such a face can be easily distorted and nonplanar, therefore the polygonal face is always tessellated by triangles; see more details in Reference 49. In Figure 2, such a nonplanar face in 3D can be seen as a nonconvex cell in 2D. For a cell  $\Omega_p, p \in \mathcal{I}$ , the indices of its faces are split into two disjoint sets; the indices of internal faces  $\mathcal{F}_p$  and the indices of boundary faces  $\mathcal{B}_p$ . For example, if the computational domain in Figure 2 consists of blue and red cells only, then  $|\mathcal{F}_p| = |\mathcal{F}_q| = 2$ ,  $|\mathcal{B}_p| = 3$ , and  $|\mathcal{B}_q| = 5$ . For a convenience, we use the subscripts  $f$  and  $b$  to indicate the internal and boundary faces,  $e_f$  and  $e_b$ , respectively, unless otherwise noted. For an internal face  $e_f, f \in \mathcal{F}_p$ , the vector  $\mathbf{n}_{pf}$  denotes the outward normal vector to the face and its length is the area of the face,  $|\mathbf{n}_{pf}| = |e_f|$ . Obviously, if  $f \in \mathcal{F}_p \cap \mathcal{F}_q$  for  $q \in \mathcal{N}_p$ , then  $\mathbf{n}_{pf} = -\mathbf{n}_{qf}$ . For  $b \in \mathcal{B}_p$ , the outward normal vector to the boundary of computational domain is denoted by  $\mathbf{n}_b = \mathbf{n}_{pb}$  with  $|\mathbf{n}_b| = |e_b|$  without mentioning the index  $p$  of the cell. A directional vector is denoted by  $\mathbf{d}_{ab} = \mathbf{x}_b - \mathbf{x}_a$ , where  $\mathbf{x}_a$  and  $\mathbf{x}_b$  are position vectors.

To compute the gradient at the center  $\mathbf{x}_p$  of the cell  $\Omega_p$ , we use the weighted least-squares method:

$$\nabla \phi_p \equiv \nabla \phi(\mathbf{x}_p) = \arg \min_{\mathbf{y} \in \mathbb{R}^3} \left( \sum_{q \in \mathcal{N}_p} \frac{(\phi_p + \mathbf{y} \cdot \mathbf{d}_{pq} - \phi_q)^2}{|\mathbf{d}_{pq}|^2} \right). \tag{14}$$

The explicit form of  $\nabla \phi_p$  can be obtained from the matrix equation:

$$\left( \sum_{q \in \mathcal{N}_p} \frac{\mathbf{d}_{pq} \otimes \mathbf{d}_{pq}}{|\mathbf{d}_{pq}|^2} \right) \nabla \phi_p = \sum_{q \in \mathcal{N}_p} \frac{\mathbf{d}_{pq}}{|\mathbf{d}_{pq}|^2} (\phi_q - \phi_p). \tag{15}$$

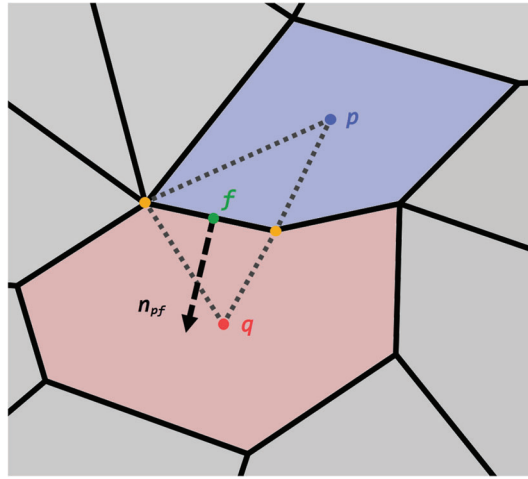


FIGURE 2 The diagram of polyhedral cells with tessellated faces

Note that the coefficient matrix in (15) is symmetric and invertible if  $|\mathcal{N}_p| \geq 3$ .

We briefly follow the derivation of the semi-implicit IIOE method<sup>43,49</sup> to obtain a cell-centered finite volume discretization for the bidirectional flow Equation 8. Expressing the norm of the gradient (if  $|\nabla\phi| \neq 0$ )

$$|\nabla\phi| = \frac{\nabla\phi}{|\nabla\phi|} \cdot \nabla\phi = \nabla \cdot \left( \phi \frac{\nabla\phi}{|\nabla\phi|} \right) - \phi \nabla \cdot \left( \frac{\nabla\phi}{|\nabla\phi|} \right), \quad (16)$$

Equation (8) can be evaluated at  $(\mathbf{x}_p, t) \in \Omega_p \times [0, T]$  as follows

$$\frac{\partial\phi}{\partial t}(\mathbf{x}_p, t) + s_p \nabla \cdot \left( \phi \frac{\nabla\phi}{|\nabla\phi|} \right)(\mathbf{x}_p, t) - s_p \phi(\mathbf{x}_p, t) \nabla \cdot \left( \frac{\nabla\phi}{|\nabla\phi|} \right)(\mathbf{x}_p, t) = s_p, \quad (17)$$

where  $s_p = s(\mathbf{x}_p)$  with the center of the cell,  $\mathbf{x}_p \in \Omega_p$ . Approximating the divergence by the averaging over the volume of the cell  $|\Omega_p|$  and applying Gauss's theorem to the integral, we obtain the approximation

$$\frac{\partial\phi}{\partial t}(\mathbf{x}_p, t) + \frac{s_p}{|\Omega_p|} \int_{\partial\Omega_p} \phi \frac{\nabla\phi}{|\nabla\phi|} \cdot \mathbf{v} dS - \frac{s_p}{|\Omega_p|} \phi(\mathbf{x}_p, t) \int_{\partial\Omega_p} \frac{\nabla\phi}{|\nabla\phi|} \cdot \mathbf{v} dS = s_p, \quad (18)$$

The second term of (18) is approximated:

$$s_p \int_{\partial\Omega_p} \phi \frac{\nabla\phi}{|\nabla\phi|} \cdot \mathbf{v} dS \approx \sum_{f \in \mathcal{F}_p \cup \mathcal{B}_p} \phi_{pf} \mu_{pf}, \quad (19)$$

where a normal flux  $\mu_{pf}$  is computed by

$$\mu_{pf} = s_p \int_{e_f} \frac{\nabla\phi}{|\nabla\phi|} \cdot \mathbf{v} dS = s_p \int_{e_f} \frac{\nabla\phi}{|\nabla\phi|} \cdot \frac{\mathbf{n}_{pf}}{|\mathbf{n}_{pf}|} dS \approx s_p \frac{\beta_f}{\sqrt{|\beta_f|^2 + \epsilon^2}} \cdot \mathbf{n}_{pf}, \quad f \in \mathcal{F}_p \cup \mathcal{B}_p, \quad (20)$$

the value  $\epsilon = 10^{-12}$  is a constant, and the face value  $\phi_{pf}$  is evaluated depending on the sign of the flux  $\mu_{pf}$ ; see the details in (26). The approximation  $\beta_f \approx \nabla\phi(\mathbf{x}_f)$  is obtained by a generalization of the diamond-cell strategy.<sup>51</sup> More specifically, for a triangle face  $e_f, f \in \mathcal{F}_p, p \in \mathcal{I}$ , the point  $\mathbf{x}_f = \frac{1}{3} \sum_{i=1}^3 \mathbf{x}_f^i$  is the center of the triangle whose vertices are  $\mathbf{x}_f^i$ . Defining a tetrahedron whose apex is  $\mathbf{x}_p$  and the base is  $e_f$ ,

$$\mathcal{T}_{pf} = \left\{ \sum_{i=1}^3 \lambda_i \mathbf{d}_{pf}^i : 0 \leq \sum_{i=1}^3 \lambda_i \leq 1, \lambda_i \geq 0, \mathbf{d}_{pf}^i = \mathbf{x}_f^i - \mathbf{x}_p \right\}, \quad (21)$$



the gradient at the center of the internal face  $e_f \subset \partial\Omega_p \cap \partial\Omega_q$ ,  $q \in \mathcal{N}_p$ , is computed by the minimization:

$$(\alpha_f, \beta_f) = \arg \min_{(\alpha_f, \beta_f) \in \mathbb{R}^4} \mathcal{E}(\alpha_f, \beta_f), \quad \mathcal{E}(\alpha_f, \beta_f) = \sum_{\mathbf{x} \in \mathcal{P}_f} \frac{|\alpha_f + \beta_f \cdot (\mathbf{x} - \mathbf{x}_f) - \phi(\mathbf{x})|^2}{|\mathbf{x} - \mathbf{x}_f|^2}, \quad (22)$$

where  $\mathcal{P}_f$  is the set of all vertices of two tetrahedrons  $\mathcal{T}_{pf}$  and  $\mathcal{T}_{qf}$ . In 2D case,  $\beta_f$  is the gradient at the center of the edge obtained by the weighted least squares method on the dotted region in Figure 2. Similarly,  $\beta_b$ ,  $b \in \mathcal{B}_p$  can be computed by using the tetrahedron  $\mathcal{T}_{pb}$ . The value of  $\phi$  at the vertex  $\mathbf{x}_v$  on the cell  $\Omega_p$  is obtained by the inverse distance weighted average of the first order Taylor's expansion for the surrounding cells which also have the same vertex  $\mathbf{x}_v$ . In the Taylor's approximation, we use the gradient at the center of cell from (14). The approximation of the third term of (18) is already presented by (20). To sum up, the spatial discretization of (18) is finally obtained:

$$|\Omega_p| \frac{\partial \phi}{\partial t}(\mathbf{x}_p, t) + \sum_{f \in \mathcal{F}_p \cup \mathcal{B}_p} (\phi_{pf} - \phi_p) \mu_{pf} = s_p |\Omega_p|. \quad (23)$$

We define the sets of indices to indicate the faces to have negative flux or non-negative flux:

$$\mathcal{F}_p^- \equiv \{f \in \mathcal{F}_p : \mu_{pf} < 0\}, \quad \mathcal{F}_p^+ \equiv \{f \in \mathcal{F}_p : \mu_{pf} \geq 0\}, \quad (24)$$

$$\mathcal{B}_p^- \equiv \{f \in \mathcal{B}_p : \mu_{pf} < 0\}, \quad \mathcal{B}_p^+ \equiv \{f \in \mathcal{B}_p : \mu_{pf} \geq 0\}. \quad (25)$$

Then, using  $\Delta t = T/K$  and  $t^n = n\Delta t$ ,  $n = 0, \dots, K$ , we can introduce the semi-implicit IIOE method<sup>43</sup> with the only one deferred iteration to obtain the finite volume discretization of the bidirectional flow (3):

$$\begin{aligned} & \frac{|\Omega_p|}{\Delta t} (\phi_p^n - \phi_p^{n-1}) + \sum_{f \in \mathcal{F}_p^-} (\phi_q^n + D_q^- \phi^{n-1} \cdot \mathbf{d}_{qf} - \phi_p^n) \mu_{pf}^{n-1} \\ & + \sum_{b \in \mathcal{B}_p^-} (\alpha_b^{n-1} - \phi_p^n) \mu_{pb}^{n-1} + \sum_{f \in \mathcal{B}_p^+ \cup \mathcal{F}_p^+} (D_p^- \phi^{n-1} \cdot \mathbf{d}_{pf}) \mu_{pf}^{n-1} = s_p |\Omega_p|, \end{aligned} \quad (26)$$

where the index  $q \in \mathcal{I}$  for  $f \in \mathcal{F}_p^-$  is such that  $f \in \mathcal{F}_p^- \cap \mathcal{F}_q$ . Furthermore,  $\phi_p^n = \phi(\mathbf{x}_p, t^n)$ ,  $\alpha_b^{n-1}$  is a linearly extended value computed by the minimization (22) for  $b \in \mathcal{B}_p$ , and the inflow-based gradient  $D_p^- \phi$  is defined by:

$$D_p^- \phi = \frac{\sum_{f \in \mathcal{F}_p^- \cup \mathcal{B}_p^-} \frac{1}{|\mathbf{d}_{pf}|} \beta_f}{\sum_{f \in \mathcal{F}_p^- \cup \mathcal{B}_p^-} \frac{1}{|\mathbf{d}_{pf}|}}. \quad (27)$$

We comment on the choice of boundary conditions for the inflow boundary faces  $e_b$ ,  $b \in \mathcal{B}_p^-$ . The linearly extended boundary value<sup>49</sup>  $\alpha_b^{n-1}$  is changed to a predefined value  $\phi_b^n = \phi_D(\mathbf{x}_b, t^n)$  from the Dirichlet boundary condition. To compute the signed distance function for a given surface inside of the computational domain, the Dirichlet boundary condition cannot be expected to be known except for academic test examples. The linearly extended value is a too simple approximation to guess the boundary value and it satisfies the steady state of (8), that is,  $|\nabla \phi| = 1$ , only if the iso-surface of the signed distance function is a plane. More crucially, the linearly extended value is calculated with no consideration of whether the Soner boundary condition is satisfied or not. Therefore, using the linearly extended value  $\alpha_b^{n-1}$  in (26) may cause that a steady state numerical solution of (26) ends up in a nonviscosity solution; see the examples in Section 4.2.

We derive here a numerical scheme that enforces the Soner boundary condition canceling effectively the negative boundary fluxes on the boundary in (26). In other words, from the decomposition of the index set of the boundary face  $\mathcal{B}_p$ ,

$$\mathcal{B}_p = \mathcal{B}_p^- \cup \mathcal{B}_p^+ = (\mathcal{B}_p^- \cap \mathcal{B}_D) \cup (\mathcal{B}_p^- \setminus \mathcal{B}_D) \cup \mathcal{B}_p^+, \quad (28)$$

where  $\mathcal{B}_D = \{b \in \mathcal{B} : e_b \subset \sigma_D\}$ , the numerical scheme uses only the fluxes from  $\mathcal{B}_p^+$  which do not violate the Soner boundary condition or from  $\mathcal{B}_p^- \cap \mathcal{B}_D$  where the Dirichlet boundary condition is assigned. Since the boundary flux from  $\mathcal{B}_p^- \setminus \mathcal{B}_D$  violates the Soner boundary condition, all values of the numerical solution occurring in the negative flux on the boundary should be ignored.

Now, combining the previous ideas in the finite volume formulation we derive our proposed scheme:

$$\begin{aligned} & \frac{|\Omega_p|}{\Delta t} (\phi_p^n - \phi_p^{n-1}) + \sum_{f \in \mathcal{F}_p^-} (\phi_q^n + D_q^* \phi^{n-1} \cdot \mathbf{d}_{qf} - \phi_p^n) \mu_{pf}^{n-1} \\ & + \sum_{b \in \mathcal{B}_p^- \cap \mathcal{B}_D} (\phi_D(\mathbf{x}_b) - \phi_p^n) \mu_{pb}^{n-1} + \sum_{f \in \mathcal{B}_p^+ \cup \mathcal{F}_p^+} (D_p^* \phi^{n-1} \cdot \mathbf{d}_{pf}) \mu_{pf}^{n-1} = s_p |\Omega_p|, \end{aligned} \tag{29}$$

where the modified inflow-based gradient  $D_p^* \phi$  is defined by

$$D_p^* \phi = \frac{\sum_{f \in \mathcal{F}_p^- \cup (\mathcal{B}_p^- \cap \mathcal{B}_D)} \frac{1}{|\mathbf{d}_{pf}|} \beta_f}{\sum_{f \in \mathcal{F}_p^- \cup (\mathcal{B}_p^- \cap \mathcal{B}_D)} \frac{1}{|\mathbf{d}_{pf}|}}. \tag{30}$$

In order to finalize the description of the proposed method, it is necessary to present how to treat the boundary condition on  $\Lambda$  in (9) and the initial condition (12) in the discretized computational domain by polyhedral cells. To do so, we divide the index set  $\mathcal{I}$  into three parts:

$$\mathcal{I}^\pm = \{p \in \mathcal{I} : \text{all vertices of } \Omega_p \in \Omega^\pm\} \quad \text{and} \quad \mathcal{I}^0 = \mathcal{I} \setminus (\mathcal{I}^+ \cup \mathcal{I}^-). \tag{31}$$

Then, we fix the values  $\phi(\mathbf{x}_p, t) = \phi_0(\mathbf{x}_p)$  for all  $t \geq 0$  and  $p \in \mathcal{I}^0$  where  $\phi_0(\mathbf{x}_p)$  is the exact (or approximated) signed distance value at  $\mathbf{x}_p$  from  $\Lambda$ .<sup>52</sup> The initial condition is written by

$$\phi(\mathbf{x}_p, 0) = \begin{cases} \phi_I^\pm(\mathbf{x}_p), & p \in \mathcal{I}^\pm \\ \phi_0(\mathbf{x}_p), & p \in \mathcal{I}^0 \end{cases} \tag{32}$$

where  $\phi_I^+$  and  $\phi_I^-$  are defined in (13). An overall solution procedure per time step is presented by Algorithm 1. Note that a final time  $T$  in Algorithm 1 is large enough to numerically obtain a steady state solution of (8).

---

**Algorithm 1.** A solution procedure per time step

---

**procedure** FVM WITH SONER BOUNDARY CONDITION

Initialization of  $\phi^0(\mathbf{x}) = \phi(\mathbf{x}, 0)$  by (32).

Set  $n = 1$ .

**while**  $n\Delta t \leq T$  **do**

    Compute  $\mu_{pf}^{n-1}$  by (20) and  $D_p^* \phi^{n-1}$  by (30).

    Solve the matrix Equation (29) to find  $\phi^n$ .

$n \leftarrow n + 1$ .

**end while**

**end procedure**

---

*Remark 1.* It is observed numerically that a restriction of the norm of the gradient approximated by  $\beta_f$  produces better results.<sup>14,42</sup> More specifically, in the case of using (29), we apply the restriction when approximating the gradient on a face:

$$(\bar{\alpha}_f, \bar{\beta}_f) = \underset{\substack{(\alpha_f, \mathbf{b}_f) \in \mathbb{R}^4 \\ |\mathbf{b}_f| \leq 1}}{\operatorname{argmin}} \mathcal{E}(\alpha_f, \mathbf{b}_f), \tag{33}$$

where  $\mathcal{E}$  is defined by (22). Then,  $|\bar{\beta}_f| \leq 1$ ,  $f \in \mathcal{F} \cup \mathcal{B}$  and the modified values of  $\beta_f$  can be used in the definition of the inflow gradient (30) and the normal flux (20). We summarize some observations of using  $\beta_f$  or  $\bar{\beta}_f$  for all numerical examples in Section 4.



### 4 | NUMERICAL EXPERIMENTS

In order to test the numerical qualities and characteristics of the proposed algorithm (29), in the following subsections, we use different computational domains of polyhedral meshes generated by AVL FIRE™ listed in Table 1.

Let  $L = 1.25$  denote a representative size of domains. The domain  $\mathcal{M}_N^1$  is a simple box of the size  $Q = [-L, L]^3 \subset \mathbb{R}^3$  in Figure 3A. The nonconvex domain  $\mathcal{M}_N^2$  is of the same size as  $\mathcal{M}_N^1$ , but it has a hole inside, see Figure 4A. Another nonconvex domain  $\mathcal{M}_N^3$  is of the size  $[-L, L]^2 \times [-\frac{L}{3}, \frac{L}{3}]$  in Figure 3B, where the blue and green surfaces are squares of the size  $[-\frac{L}{3}, \frac{L}{3}]$ . The domain  $\mathcal{M}_N^4$  is of a general shape of the size  $[0, 2L] \times [0, 1.2L] \times [0, 1.76L]$  in Figure 12A and it is typically used to simulate a motion of the electric arc generated by the ignition process in a combustion engine. The average size  $h_N$  of cells and its extremal values are given in Table 1:

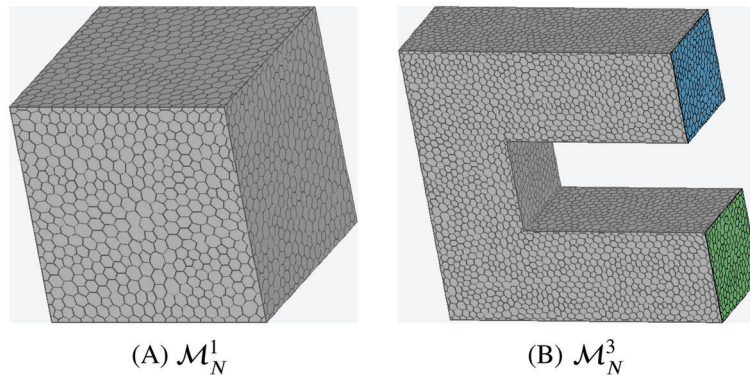


FIGURE 3 The shape of computational domains  $\mathcal{M}_N^1$  and  $\mathcal{M}_N^3$ ,  $N = 1$ , in Table 1

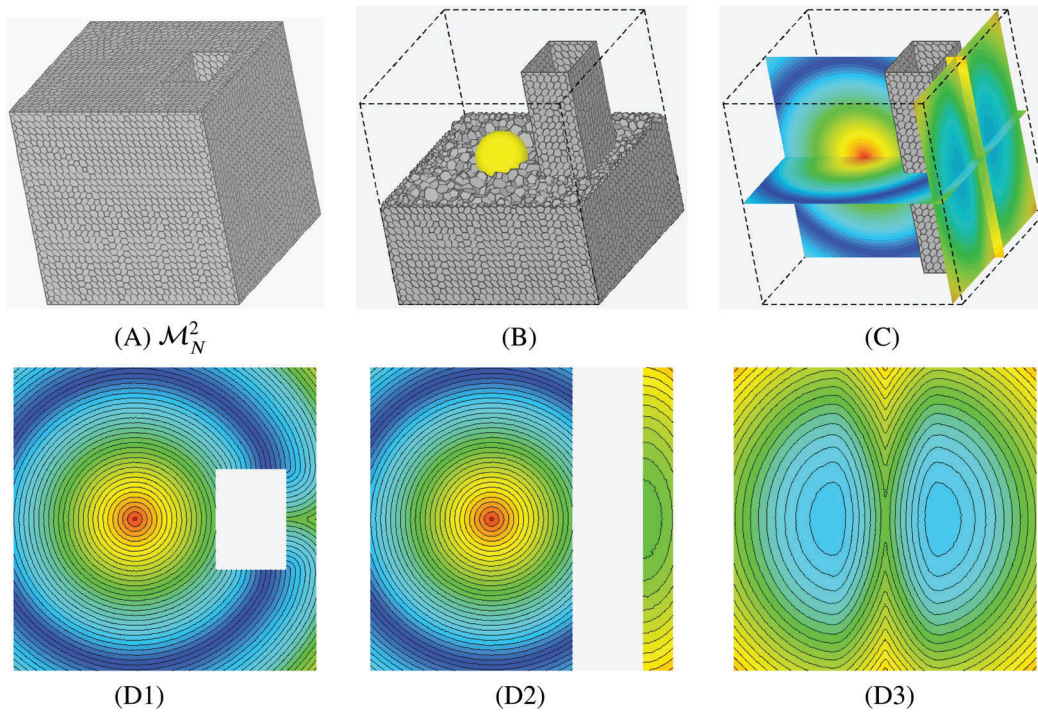


FIGURE 4 (A) is the shape of computational domain  $\mathcal{M}_N^2$ ,  $N = 1$ , in Table 1. (B) shows the inner part of (A) and the yellow sphere is the given surface  $\Lambda$  in (8). (C) is the numerical result of (29) at  $T = 5$  on the coordinate planes. (D1), (D2), and (D3) are the results in (C) from the orthogonal view with the iso-surfaces

**TABLE 1** The list of computational domains:  $\mathcal{M}_N^1$  in Figure 3A,  $\mathcal{M}_N^2$  in Figure 4A,  $\mathcal{M}_N^3$  in Figure 3B, and  $\mathcal{M}_N^4$  in Figure 12A

	$N$	$ \mathcal{I}_N $	$h_N$	$h_N^{\min}$	$h_N^{\max}$
$\mathcal{M}_N^1$	1	4079	$1.90 \times 10^{-1}$	$6.56 \times 10^{-2}$	$4.21 \times 10^{-1}$
	2	32004	$9.52 \times 10^{-2}$	$2.29 \times 10^{-2}$	$2.00 \times 10^{-1}$
	3	252433	$4.76 \times 10^{-2}$	$1.36 \times 10^{-2}$	$9.46 \times 10^{-2}$
	4	2024478	$2.48 \times 10^{-2}$	$5.51 \times 10^{-3}$	$4.48 \times 10^{-2}$
$\mathcal{M}_N^2$	1	14821	$1.17 \times 10^{-1}$	$2.72 \times 10^{-2}$	$3.13 \times 10^{-1}$
	2	70859	$6.89 \times 10^{-2}$	$1.50 \times 10^{-2}$	$1.65 \times 10^{-1}$
	3	363418	$4.08 \times 10^{-2}$	$5.88 \times 10^{-3}$	$8.35 \times 10^{-2}$
	4	2153388	$2.34 \times 10^{-2}$	$2.96 \times 10^{-3}$	$4.31 \times 10^{-2}$
$\mathcal{M}_N^3$	1	18118	$7.02 \times 10^{-2}$	$1.59 \times 10^{-2}$	$1.90 \times 10^{-1}$
	2	74301	$4.22 \times 10^{-2}$	$9.25 \times 10^{-3}$	$1.32 \times 10^{-1}$
	3	362679	$2.44 \times 10^{-2}$	$5.38 \times 10^{-3}$	$6.60 \times 10^{-2}$
	4	1868820	$1.45 \times 10^{-2}$	$2.42 \times 10^{-3}$	$3.60 \times 10^{-2}$
$\mathcal{M}_N^4$	1	37303	$6.64 \times 10^{-2}$	$1.04 \times 10^{-2}$	$1.88 \times 10^{-1}$
	2	149351	$4.17 \times 10^{-2}$	$5.36 \times 10^{-3}$	$1.04 \times 10^{-1}$
	3	1068890	$2.27 \times 10^{-2}$	$3.29 \times 10^{-3}$	$4.71 \times 10^{-2}$
	4	6451022	$1.29 \times 10^{-2}$	$1.41 \times 10^{-3}$	$2.39 \times 10^{-2}$

Note:  $\mathcal{I}_N$  is the number of cells and the average, minimal and maximal size of cells is defined in (34).

$$h_N = \frac{1}{|\mathcal{I}_N|} \sum_{p \in \mathcal{I}_N} |\Omega_p|_B^{\frac{1}{3}}, \quad h_N^{\min} = \min_{p \in \mathcal{I}_N} |\Omega_p|_B^{\frac{1}{3}}, \quad h_N^{\max} = \max_{p \in \mathcal{I}_N} |\Omega_p|_B^{\frac{1}{3}}, \quad (34)$$

where  $N \in \{1, 2, 3, 4\}$  and  $|\Omega_p|_B$  is the volume of the smallest box to enclose the polyhedron cell  $\Omega_p$ .

The time steps for the level  $N$  are given by

$$\Delta t = \frac{0.04}{N}, \quad N \in \{1, 2, 3, 4\} \quad (35)$$

for most examples, unless otherwise noted. For all examples in this article, we use the end time  $T$  large enough in (8) to guarantee that a numerical solution of the proposed algorithm (29) is in a steady state. The initial conditions, except the Section 4.3, are selected by one of three functions,  $\psi_0$ ,  $\psi_1$ , and  $\psi_{-1}$ :

$$\psi_0(\mathbf{x}_p) = \begin{cases} \pm 0.1, & p \in \mathcal{I}^\pm, \\ \phi_0(\mathbf{x}_p), & p \in \mathcal{I}^0, \end{cases} \quad \text{and} \quad \psi_i(\mathbf{x}_p) = \begin{cases} 3^i \phi_0(\mathbf{x}_p), & p \in \mathcal{I}^\pm, \\ \phi_0(\mathbf{x}_p), & p \in \mathcal{I}^0, \end{cases} \quad (36)$$

where  $i \in \{-1, 1\}$  and the choice of  $\phi_0$  is explained in each example.

Note that the color scheme “red-blue-red” is used to present all numerical results obtained by (29). The minimum and maximum value in the red-blue-red scheme are red and the intermediate values from small to large are presented from red to blue and then from blue to red. Moreover, we use a single color between two adjacent iso-surfaces to present the shape of iso-surfaces without explicitly drawing them.

#### 4.1 | Experimental order of convergence

The experimental order of convergence (EOC) is checked using appropriate norms of errors  $E^1$  and  $E^\infty$ , which are  $L^1(\Omega)$  and  $L^\infty(\Omega)$  norms of the difference between the exact solution and the numerical solution, respectively, and

**TABLE 2** The *EOCs* are presented to compute the signed distance from the sphere (38) on the domain  $\mathcal{M}_N^1$  for the example in Section 4.1

$N$	$E^1$	<i>EOC</i>	$E^\infty$	<i>EOC</i>	$E_\sigma^\infty$	<i>EOC</i>
1	$5.03 \times 10^{-3}$		$2.87 \times 10^{-2}$		$2.87 \times 10^{-2}$	
2	$1.54 \times 10^{-3}$	1.71	$1.32 \times 10^{-2}$	1.12	$1.19 \times 10^{-2}$	1.27
3	$3.60 \times 10^{-4}$	2.10	$7.41 \times 10^{-3}$	0.83	$2.60 \times 10^{-3}$	2.20
4	$7.45 \times 10^{-5}$	2.42	$3.67 \times 10^{-3}$	1.08	$6.98 \times 10^{-4}$	2.02
$N$	$E^1$	<i>EOC</i>	$E^\infty$	<i>EOC</i>	$E_\sigma^\infty$	<i>EOC</i>
1	$5.76 \times 10^{-3}$		$2.15 \times 10^{-2}$		$2.15 \times 10^{-2}$	
2	$1.77 \times 10^{-3}$	1.71	$1.65 \times 10^{-2}$	0.38	$1.09 \times 10^{-2}$	0.98
3	$4.13 \times 10^{-4}$	2.10	$6.23 \times 10^{-3}$	1.41	$2.64 \times 10^{-3}$	2.05
4	$8.37 \times 10^{-5}$	2.45	$2.51 \times 10^{-3}$	1.40	$6.33 \times 10^{-4}$	2.20

Note: The upper and lower table shows the results of (29) using  $\beta_f$  in (22) and  $\bar{\beta}_f$  in (33), respectively.

where  $\Omega$  is one of the computational domain in Table 1. For each error norm, the corresponding *EOC* is computed by

$$EOC_N = \frac{\log(E_N/E_{N-1})}{\log(h_N/h_{N-1})}, \quad N \in \{2, 3, 4\}, \quad (37)$$

where  $E_N$  is either  $E^1$  or  $E^\infty$  at the  $N^{\text{th}}$  level.

To show the expected *EOC* for the scheme (29) we compute the signed distance function to a sphere on the computational domain  $\Omega = \mathcal{M}^1$  in Figure 3A. The given surface  $\Lambda$  in (8) is the sphere:

$$\Lambda = \{\mathbf{x} \in \Omega : S_{r,\mathbf{c}}(\mathbf{x}) = 0\}, \quad S_{r,\mathbf{c}}(\mathbf{x}) = |\mathbf{x} - \mathbf{c}| - r, \quad (38)$$

where  $r = 0.6$  and  $\mathbf{c} = \mathbf{0}$ . We choose  $\phi_0 = S_{0.6,\mathbf{0}}$  and  $\psi_{-1}$  as the initial condition in (36).

Since the exact solution is differentiable except at the origin, it is a good standard example to check the highest *EOC* of the proposed algorithm (29) to compute the signed distance function. In Table 2, we present the numerical results at  $T = 5$  and compare the results for  $\beta_f$  in (22) and  $\bar{\beta}_f$  in (33). The restriction of the norm of the gradient does not show any significant difference with respect to the unrestricted form. The *EOCs* show the second order convergence in the  $L^1$  norm and the first order convergence in the  $L^\infty$  norm. Additionally, we check the error  $E_\sigma^\infty$  defined by  $L^\infty(\Omega_\sigma)$  norm of the difference between the exact solution and the numerical solution, where  $\Omega_\sigma = \{\mathbf{x} \in \Omega : |\mathbf{x}| > \sigma\}$  and  $\sigma = 0.2$ . Since the singularity is removed, the corresponding *EOC* also shows the second order convergence.

## 4.2 | Nonconvex domain

A typical example to compute a signed distance function on a nonconvex domain is the evolution of level set function close to a nozzle structure in complicated shape of computational domain or an electric spark plug in a combustion engine chamber; see Figure 12. Since a signed distance function on a nonconvex domain is mostly a nonanalytic form, it is necessary to choose rather simple shape of the nonconvex domain in order to check the *EOC*. We consider the surface  $\Lambda$  in Figure 4B defined by  $S_{r,\mathbf{c}} = 0$  in (38) with  $r = 0.3$  and  $\mathbf{c} = \mathbf{0}$  on the computational domain  $\mathcal{M}^2$  in Figure 4A. Unlike the simple example in Section 4.1, the function  $S_{0.3,\mathbf{0}}$  is not the exact solution anymore, because of the hole in the computational domain. In order to find the exact solution, let us denote three colored regions on the  $xy$ -plane of the domain in Figure 5:

$$\mathcal{M}^2 = \Omega = (\Omega_{\text{gray}} \cup \Omega_{\text{red}} \cup \Omega_{\text{blue}}) \times [-L, L], \quad (39)$$

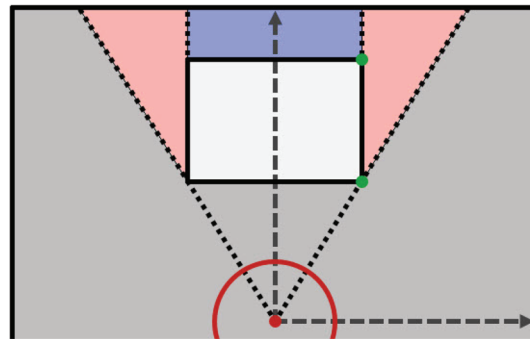
where  $L = 1.25$ . Then, the exact solution can be analytically computed by multivariable calculus:

$$\begin{aligned} \phi_e(\mathbf{x}) &= \bar{\phi}_e(|x_1|, x_2, x_3), \quad \mathbf{x} = (x_1, x_2, x_3) \in \Omega, \\ \bar{\phi}_e(\mathbf{x}) &= \begin{cases} |\mathbf{x}| - r, & \mathbf{x} \in \Omega_{\text{gray}} \times [-L, L], \\ ((|P(\mathbf{p}_r)| + |P(\mathbf{p}_r - \mathbf{x})|)^2 + x_3^2)^{\frac{1}{2}} - r, & \mathbf{x} \in \Omega_{\text{red}} \times [-L, L], \\ ((|P(\mathbf{p}_b)| + |P(\mathbf{p}_b - \mathbf{x})|)^2 + x_3^2)^{\frac{1}{2}} - r, & \mathbf{x} \in \Omega_{\text{blue}} \times [-L, L], \end{cases} \end{aligned} \quad (40)$$

where  $\mathbf{p}_r = (p_1, p_2, p_3)$  and  $\mathbf{p}_b = (p_1, p_2 + \ell, p_3)$ ,  $\ell > 0$ , are the lower and upper green points in Figure 5 and  $P(x_1, x_2, x_3) = (x_1, x_2, 0)$  is the projection operator onto the  $xy$ -plane. In Figure 4, the numerical results of (29) at  $T = 5$  are plotted for the initial condition  $\psi_{-1}$  and  $\phi_0 = S_{0.3,0}$  in (36). Looking at the iso-curves in the coordinate planes, we can see that the Soner boundary condition (11) is clearly enforced to the numerical results.

In Table 3, the  $EOC$  is computed by the exact solution (40) and the numerical solution of (8). The results of (29) using  $\beta_f$  in (22) and  $\bar{\beta}_f$  in (33) are presented on the left and right in Table 3, respectively. With  $\phi_0 = S_{0.3,0}$  in (36) we use the initial conditions  $\psi_{-1}$  on the left and  $\psi_0$  on the right. Nevertheless, the  $EOCs$  for these two choices do not differ significantly. The values of  $E^1$  and  $E^\infty$  are mostly smaller in the case of no restriction of the norm of the gradient.

In order to see numerical behavior for different initial conditions, we choose all three initial conditions (36) with  $\phi_0 = S_{0.3,0}$  on the computational domain  $\mathcal{M}_2^2$  and compare the numerical results as the log-scaled plot of the  $E^1$  over the time in Figure 6. The curves on the left and right are the results of (29) using  $\beta_f$  in (22) and  $\bar{\beta}_f$  in (33), respectively, and the label  $f_{i+2}$  means the result for the initial condition  $\psi_i$  for  $i \in \{-1, 0, 1\}$ . The red and blue colored curves are the results of the proposed method (29) and the previous method (26), respectively. Since the previous method does not consider the Soner boundary condition, it is difficult to converge to the viscosity solution on a nonconvex domain unless a manipulated initial condition is selected.



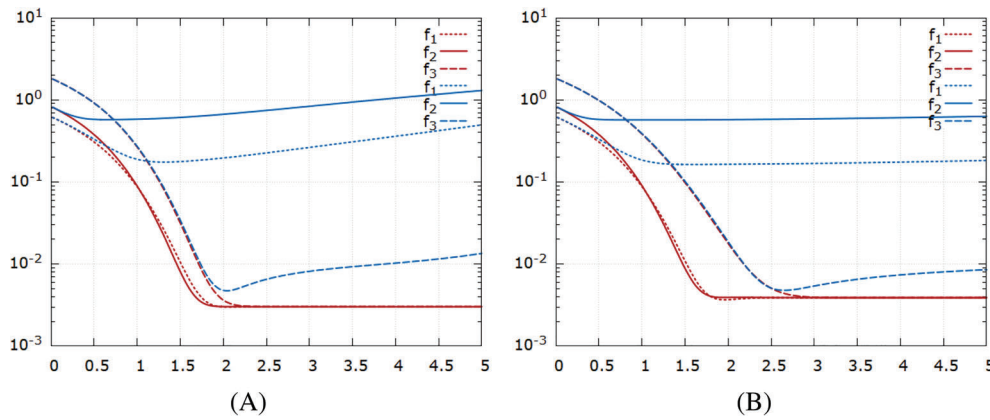
**FIGURE 5** The  $xy$ -plane of the computational domain  $\mathcal{M}_N^2$  in Figure 4 is presented. The red circle is the given surface  $\Lambda$  in (8). The lower and upper green points,  $\mathbf{p}_r = (p_1, p_2, p_3)$  and  $\mathbf{p}_b = (p_1, p_2 + \ell, p_3)$  with the height of the hole  $\ell > 0$ , are used to compute the exact solution (40). The gray region is the part of the domain visible to the center of the circle (red point) and the colored regions (red and blue) are not directly visible to the red point

**TABLE 3** The  $EOCs$  for the example in Section 4.2 are presented to compute the signed distance from the sphere in Figure 4B on the domain  $\mathcal{M}_N^2$  in Table 1

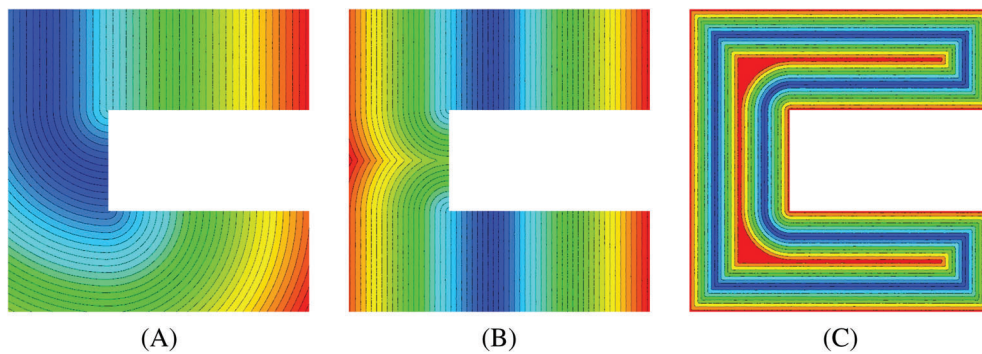
$N$	$E^1$	$EOC$	$E^\infty$	$EOC$	$N$	$E^1$	$EOC$	$E^\infty$	$EOC$
1	$8.03 \times 10^{-3}$		$4.89 \times 10^{-2}$		1	$1.19 \times 10^{-2}$		$6.73 \times 10^{-2}$	
2	$3.02 \times 10^{-3}$	1.86	$2.81 \times 10^{-2}$	1.05	2	$3.89 \times 10^{-3}$	2.11	$3.75 \times 10^{-2}$	1.11
3	$9.39 \times 10^{-4}$	2.23	$2.11 \times 10^{-2}$	0.55	3	$1.24 \times 10^{-3}$	2.18	$2.48 \times 10^{-2}$	0.78
4	$3.05 \times 10^{-4}$	2.03	$9.43 \times 10^{-3}$	1.45	4	$4.23 \times 10^{-4}$	1.94	$1.15 \times 10^{-2}$	1.38

Note: The results of (29) using  $\beta_f$  in (22) and  $\bar{\beta}_f$  in (33) are presented on the left and right, respectively.





**FIGURE 6** (A,B) are the log-scaled graphs of  $E^1$  for the example in Section 4.2 on the computational domain  $\mathcal{M}_2^2$  in Table 1 with  $\beta_f$  in (22) and  $\bar{\beta}_f$  in (33), respectively. The labels indicate a different choice of initial conditions  $f_{i+1} = \psi_i, i \in \{-1, 0, 1\}$ , in (36) and the red and blue colored curves mean the results from the proposed method (29) and the previous method (26), respectively



**FIGURE 7** (A–C) are the iso-surfaces on  $xy$ -plane of the numerical results of the proposed algorithm (29) on the domain  $\mathcal{M}_4^3$  in Table 1 with Dirichlet boundary conditions,  $\phi_D^1, \phi_D^2, \phi_D^3$  in (41), respectively

### 4.3 | Mixed boundaries

Using the computational domain  $\mathcal{M}^3$  in Figure 3B, the proposed scheme (29) is tested to solve the cases having both the Dirichlet and the Soner boundary conditions. Such a mixed boundary condition can be useful to check visibility detection when an object of the interest is located on a part of the boundary of the computational domain. We consider the governing Equation 8 only with  $s(\mathbf{x}) = 1$  on  $\Omega = \Omega^+$  and  $\Lambda \subset \partial\Omega$  using the zero Dirichlet boundary condition on  $\Lambda$  and the Soner boundary condition on  $\partial\Omega \setminus \Lambda$ . Let us note that the exact solutions in such case is not a signed distance function but a distance function from the given  $\Lambda \subset \partial\Omega$ . Let us denote by  $\partial\Omega_{\text{blue}}$  and  $\partial\Omega_{\text{green}}$  the blue and green boundary surface on the computational domain  $\Omega = \mathcal{M}^3$  in Figure 3B. Then, we compute examples using three different Dirichlet boundary conditions:

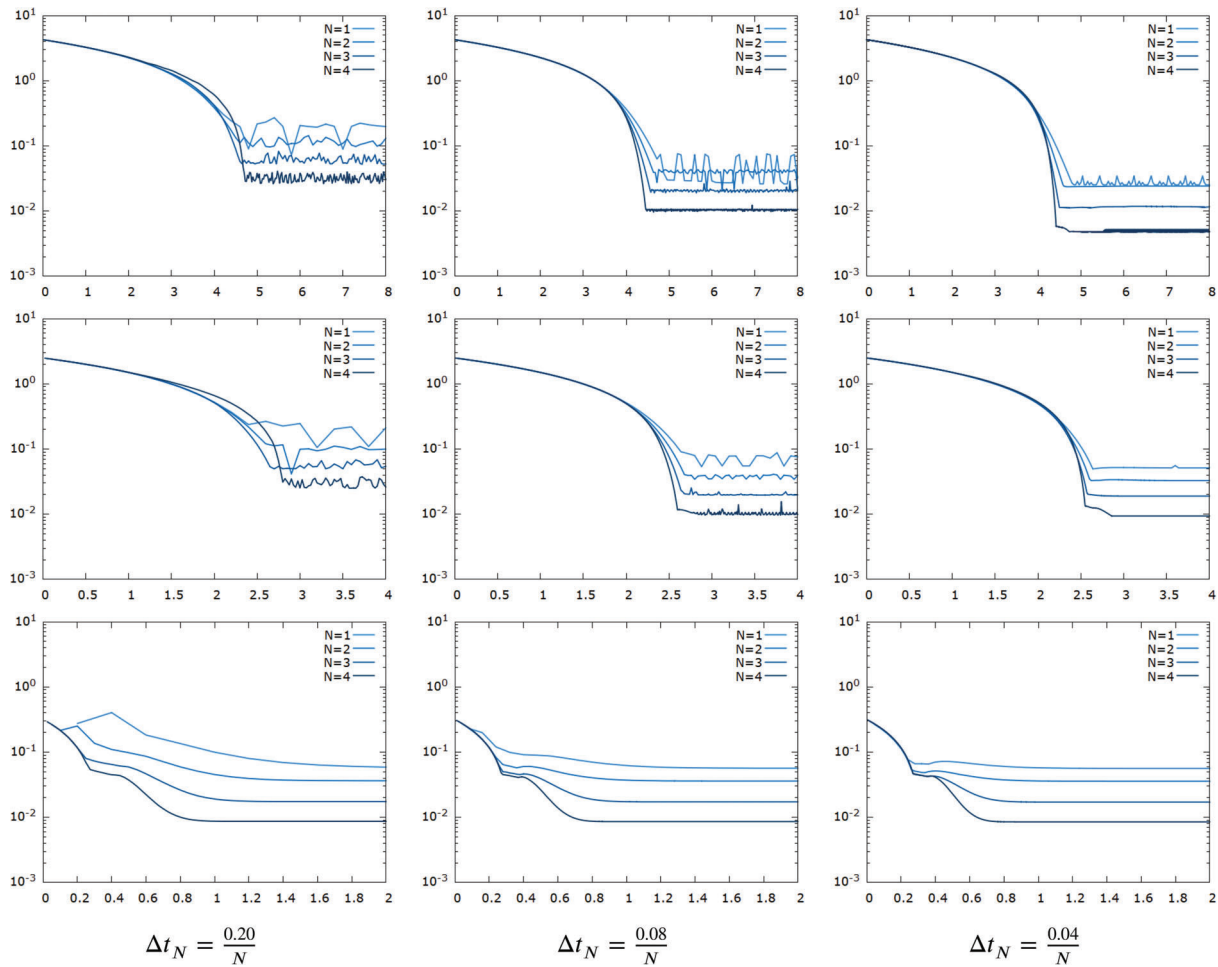
$$\begin{aligned} \phi_D^1(\mathbf{x}) &= 0, & \mathbf{x} \in \partial\Omega_{\text{blue}}, \\ \phi_D^2(\mathbf{x}) &= 0, & \mathbf{x} \in \partial\Omega_{\text{blue}} \cup \partial\Omega_{\text{green}}, \\ \phi_D^3(\mathbf{x}) &= 0, & \mathbf{x} \in \partial\Omega. \end{aligned} \tag{41}$$

Note that in the case of  $\phi_D^3$  the surface area of applying Soner boundary condition is empty, however, it is not empty in the cases of  $\phi_D^1$  and  $\phi_D^2$ . The solution of using  $\phi_D^3$  is called the wall distance function and the solution of using  $\phi_D^2$  or  $\phi_D^3$  is the shortest distance from the surface  $\Lambda \subset \partial\Omega$ . In Figure 7, we present the iso-surfaces on  $xy$ -plane of the numerical results of the proposed scheme (29) with three Dirichlet boundary conditions,  $\phi_D^1, \phi_D^2$ , and  $\phi_D^3$ , respectively. Considering

**TABLE 4** The EOCs for the examples in Section 4.3 are presented to compute the distance function on the domain  $\mathcal{M}_N^3$  in Table 1 using the proposed algorithm (29) with the Soner and Dirichlet boundary conditions

$N$	$E^1$	EOC	$E^\infty$	EOC	$N$	$E^1$	EOC	$E^\infty$	EOC
1	$6.22 \times 10^{-3}$		$6.09 \times 10^{-2}$		1	$6.74 \times 10^{-3}$		$2.52 \times 10^{-2}$	
2	$3.59 \times 10^{-3}$	1.08	$3.40 \times 10^{-2}$	1.15	2	$4.78 \times 10^{-3}$	0.68	$2.42 \times 10^{-2}$	0.08
3	$1.82 \times 10^{-3}$	1.24	$1.65 \times 10^{-2}$	1.32	3	$2.50 \times 10^{-3}$	1.18	$1.16 \times 10^{-2}$	1.34
4	$8.42 \times 10^{-4}$	1.48	$8.96 \times 10^{-3}$	1.17	4	$1.31 \times 10^{-3}$	1.25	$4.79 \times 10^{-3}$	1.70
$N$	$E^1$	EOC	$E^\infty$	EOC	$N$	$E^1$	EOC	$E^\infty$	EOC
1	$1.99 \times 10^{-3}$		$4.37 \times 10^{-2}$		1	$2.63 \times 10^{-3}$		$5.12 \times 10^{-2}$	
2	$1.22 \times 10^{-3}$	0.96	$2.93 \times 10^{-2}$	0.78	2	$1.57 \times 10^{-3}$	1.01	$3.28 \times 10^{-2}$	0.88
3	$6.24 \times 10^{-4}$	1.23	$1.61 \times 10^{-2}$	1.10	3	$7.73 \times 10^{-4}$	1.29	$1.89 \times 10^{-2}$	1.00
4	$3.14 \times 10^{-4}$	1.32	$8.08 \times 10^{-3}$	1.33	4	$3.92 \times 10^{-4}$	1.31	$9.39 \times 10^{-3}$	1.35
$N$	$E^1$	EOC	$E^\infty$	EOC	$N$	$E^1$	EOC	$E^\infty$	EOC
1	$4.83 \times 10^{-3}$		$5.18 \times 10^{-2}$		1	$6.10 \times 10^{-3}$		$5.61 \times 10^{-2}$	
2	$2.77 \times 10^{-3}$	1.09	$3.42 \times 10^{-2}$	0.81	2	$3.51 \times 10^{-3}$	1.08	$3.58 \times 10^{-2}$	0.88
3	$9.64 \times 10^{-4}$	1.93	$1.66 \times 10^{-2}$	1.33	3	$1.22 \times 10^{-3}$	1.93	$1.71 \times 10^{-2}$	1.35
4	$2.93 \times 10^{-4}$	2.29	$8.72 \times 10^{-3}$	1.23	4	$3.69 \times 10^{-4}$	2.30	$8.51 \times 10^{-3}$	1.35

Note: The  $\beta_f$  in (22) and  $\bar{\beta}_f$  in (33) are used in the left and right table, respectively. The top, middle, and bottom tables present the results at the end time  $T = 8$  with the Dirichlet boundary condition  $\phi_D^1$ ,  $T = 4$  with  $\phi_D^2$ ,  $T = 2$  with  $\phi_D^3$  in (41), respectively. The time step  $\Delta t = \frac{0.04}{N}$  is used.



**FIGURE 8** The graphs in the top, middle, and bottom row are the log-scaled values of  $E^\infty$  over the time for three Dirichlet boundary conditions  $\phi_D^1$ ,  $\phi_D^2$ , and  $\phi_D^3$  in (41) when using  $\bar{\beta}_f$  in (33), respectively. From the left to the right we use smaller time steps and the level  $N$  indicates the parameters of computational domain  $\mathcal{M}_N^3$  in Table 1



the surface area to which the Dirichlet boundary conditions are applied, the numerical results in Figure 7 qualitatively show the distance profiles from the zero Dirichlet boundaries.

In Table 4, we present quantitative results to compare EOCs of  $E^1$  and  $E^\infty$  errors from  $\beta_f$  in (22) on the left and  $\bar{\beta}_f$  in (33) on the right. The top, middle, and bottom tables present numerical results at the end time  $T = 8$  with the Dirichlet boundary condition  $\phi_D^1$ ,  $T = 4$  with  $\phi_D^2$ ,  $T = 2$  with  $\phi_D^3$ , respectively. The chosen end times are large enough to obtain the distance function from the zero Dirichlet boundaries. The error values are mostly smaller for the results of no restriction of the norm of the gradient than the ones with the restriction. The time step  $\Delta t_N = \frac{0.04}{N}$  is used. Since  $h_N^{\min} < \Delta t_N < h_N$  for  $N \in \{1, 2, 3, 4\}$ , the Courant number can be locally larger than 1, but mostly it is smaller than 1. Since the time discretization in the numerical algorithm (29) is the semi-implicit method, the size of the time step is not necessarily smaller than the minimum size of cells.<sup>43</sup> In the case of the explicit method, the size of the time step is restricted by the CFL condition.<sup>14,42</sup>

In order to see the complete behavior of the errors for the variations of time steps, depending on whether the restriction of the norm of the gradient is used or not, we present the log-scaled error graphs over the time of all test cases in this section. The  $E^\infty$  and  $E^1$  graphs for the results of using  $\bar{\beta}_f$  in (33) are presented in Figures 8 and 9, respectively. The  $E^\infty$  and  $E^1$  graphs for the results of using  $\beta_f$  in (22) are presented in Figures 10 and 11, respectively. For all mentioned figures, the top, middle, and the bottom rows are the graphs for three Dirichlet boundary conditions  $\phi_D^1$ ,  $\phi_D^2$ , and  $\phi_D^3$  in (41) and the

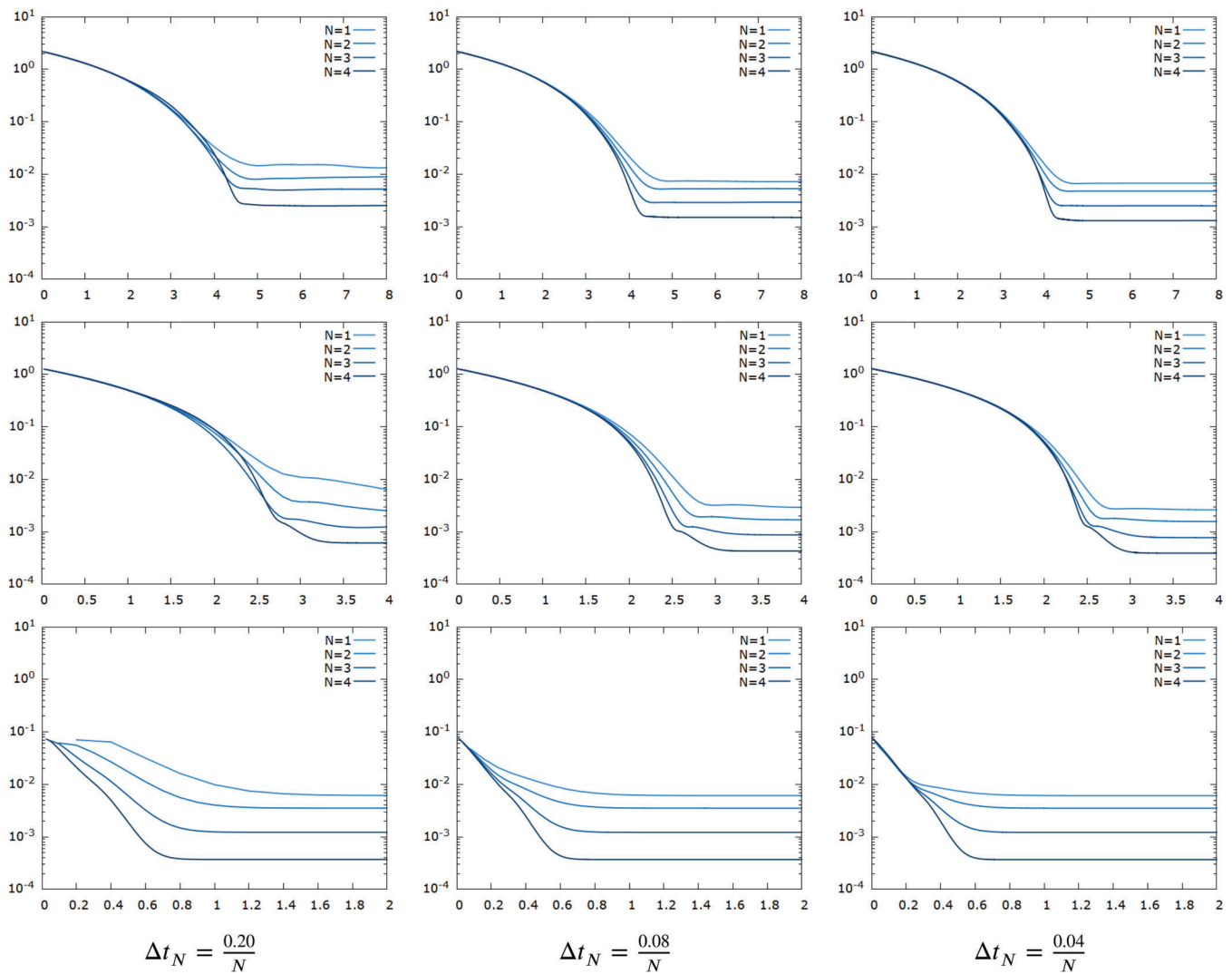
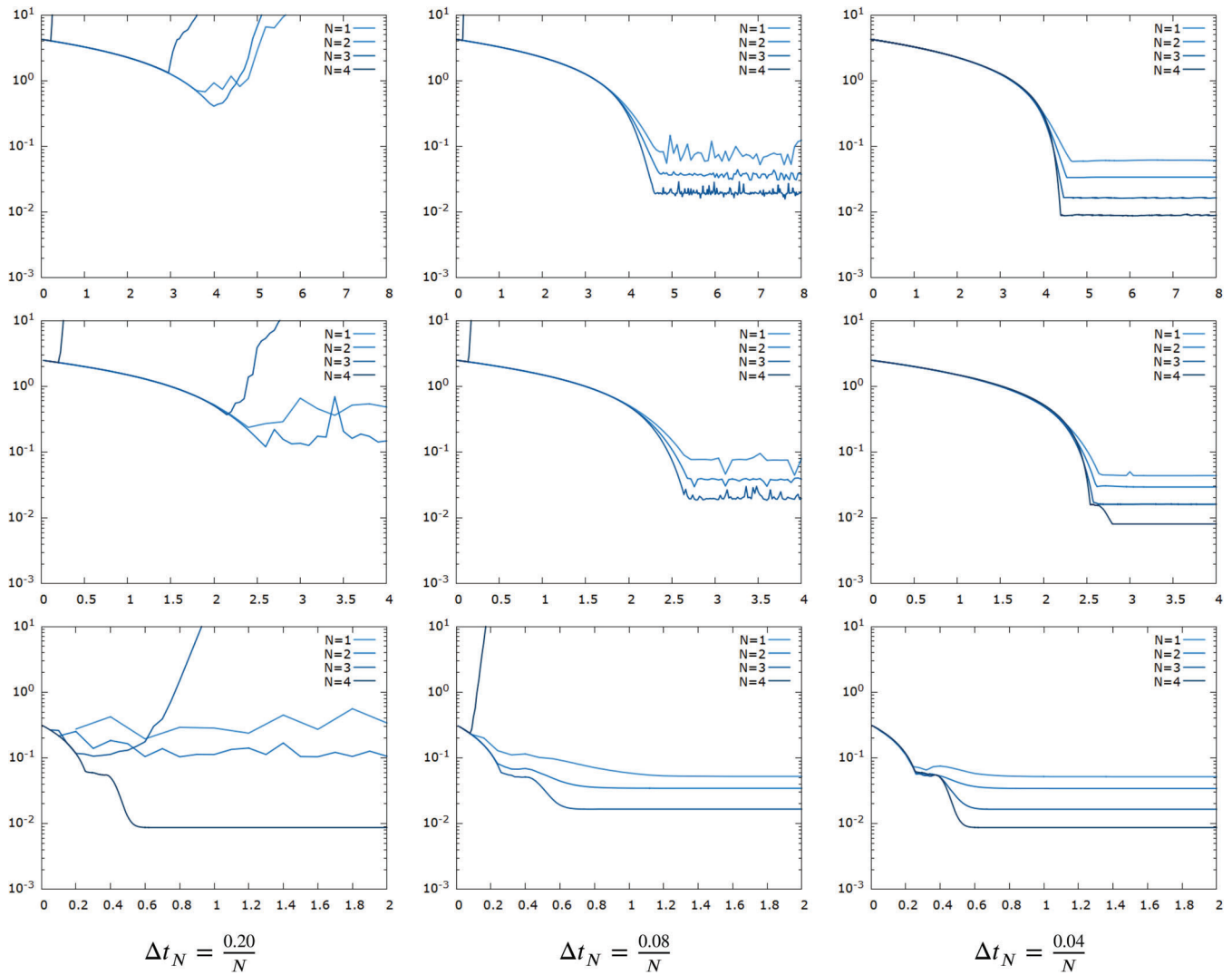


FIGURE 9 The graphs on the top, middle, and bottom rows are the log-scaled values of  $E^1$  over the time for three Dirichlet boundary conditions  $\phi_D^1$ ,  $\phi_D^2$ , and  $\phi_D^3$  in (41) when using  $\bar{\beta}_f$  in (33), respectively. From the left to the right, we use smaller time steps and the level  $N$  indicates the parameters of computational domain  $\mathcal{M}_N^3$  in Table 1

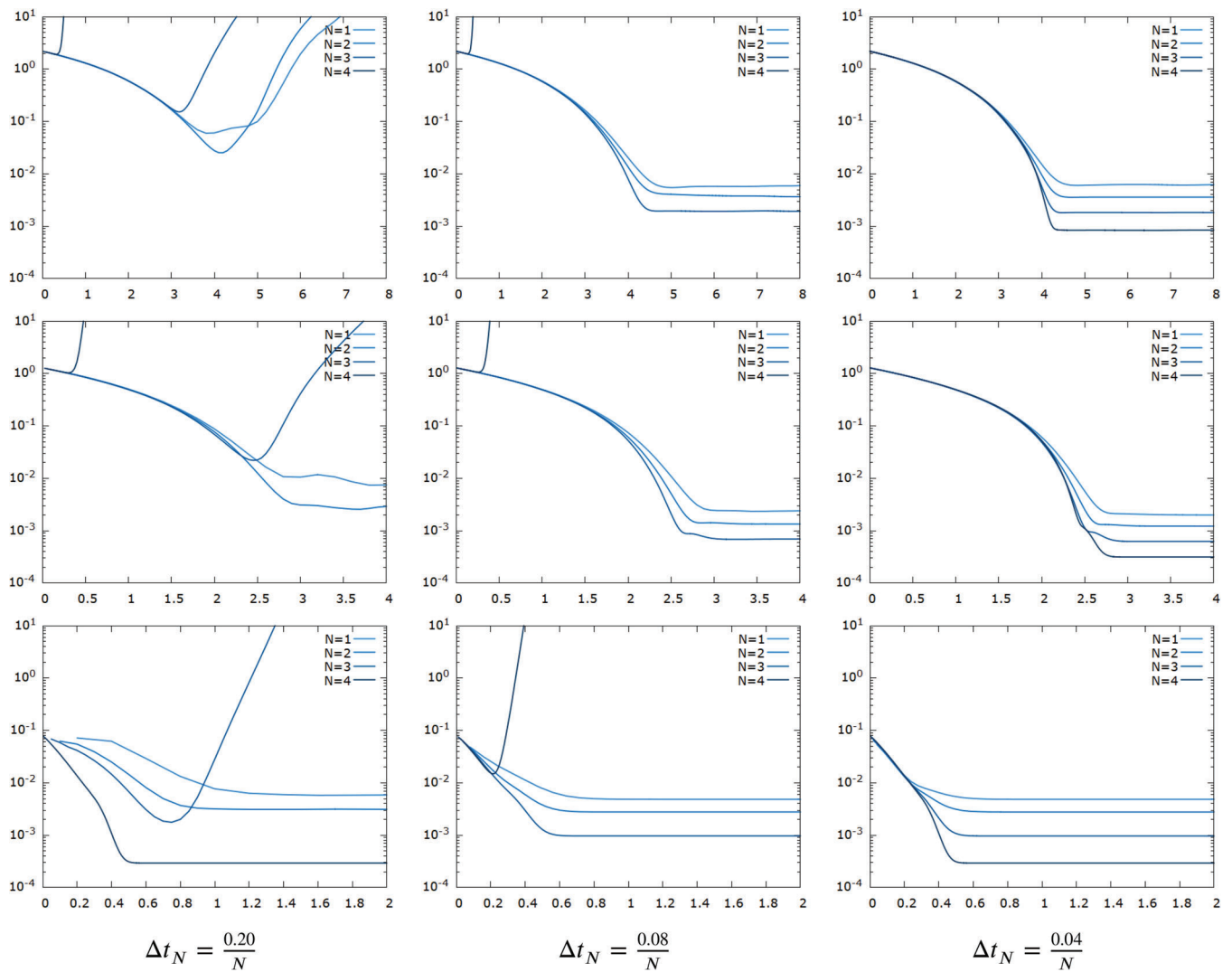


**FIGURE 10** The graphs on the top, middle, and bottom row are the log-scaled values of  $E^\infty$  over the time for three Dirichlet boundary conditions  $\phi_D^1$ ,  $\phi_D^2$ , and  $\phi_D^3$  in (41) when using  $\beta_f$  in (22), respectively. From the left to the right, we use smaller time steps and the level  $N$  indicates the parameters of computational domain  $\mathcal{M}_N^3$  in Table 1

left, middle, and the right columns are the graphs for time steps,  $\Delta t_N = \frac{0.2}{N}$ ,  $\Delta t_N = \frac{0.08}{N}$ , and  $\Delta t_N = \frac{0.04}{N}$ , respectively. In Table 5, the maximum, the average, and the minimum of Courant numbers of the computational domain  $\mathcal{M}_N^3$  in Table 1 are presented;

$$C_M = \frac{\Delta t_N}{h_N^{\min}}, \quad C_a = \frac{1}{|\mathcal{I}|} \sum_{p \in \mathcal{I}} \frac{\Delta t_N}{|\Omega_p|^{1/3}}, \quad C_m = \frac{\Delta t_N}{h_N^{\max}}. \quad (42)$$

Note that  $\Delta t_N = \frac{0.2}{N} > h_N$ , on the computational domain  $\mathcal{M}_N^3$  in Table 1, for  $N \in \{1, 2, 3, 4\}$  and that the percentage of cells whose the Courant number larger than  $C = \frac{\Delta t_N}{h_N} > 1$  is more than 50%. From the graphs, due to the CFL condition, it is clear to observe that the overall stability becomes better when a smaller time step is used. The results are more stably obtained by using the restriction of the norm of the gradient,  $\bar{\beta}_f$  in (33). There are some oscillations in  $E^\infty$  when the larger time step is used with the Soner boundary condition. Nevertheless, it shows stable convergence in  $L^1$  norm. The same idea is introduced in the computation of the wall distance function<sup>14</sup> or signed distance function.<sup>42</sup> When the results are computed by no restriction of the norm of the gradient,  $\beta_f$  in (22), many examples with the larger time step have diverged. The unstable or nonconvergent results on the polyhedron mesh are heavily related to the qualities of the mesh when it



**FIGURE 11** The graphs on the top, middle, and bottom row are the log-scaled values of  $E^1$  over the time for three Dirichlet boundary conditions  $\phi_D^1$ ,  $\phi_D^2$ , and  $\phi_D^3$  in (41) when using  $\beta_f$  in (22), respectively. From the left to the right, we use smaller time steps and the level  $N$  indicates the parameters of computational domain  $\mathcal{M}_N^3$  in Table 1

is generated. Especially, the accuracy of computing gradient is not reliable in the vicinity of low-quality polyhedral cells. The general treatment to avoid diverged results is to reduce the size of a time step, but the smaller time step increases the computational cost. Alternatively, keeping the large time step, the restriction of the norm of the gradient  $\bar{\beta}_f$  in (33) can be used to achieve the convergent results. Since there are so many factors to control the qualities of the mesh depending on the purpose, it is generally not advised to change the mesh quality only for a single goal. The fortunate factor of computing the wall distance function is that the *EOC* is not damaged by the restriction,  $|\nabla\phi| \leq 1$ , because we eventually look for the numerical solution close to  $|\nabla\phi| = 1$ . For the industrial application such as finding a distance function from the entire boundary of the computational domain, the restriction of the gradient in Remark 1 is practically better because the boundary shapes of the domain are highly complex and the mesh quality is not always the best.

#### 4.4 | General domain

The computational domain in Figure 12 is a part of the electric spark plug in the combustion engine chamber. After a high energy from the spark plug is released into the chamber, an initial flame can be ignited at multiple locations. Then, it is necessary to initialize the level set function from multiple spheres shown in Figure 12C. We use the given surface  $\Lambda$

TABLE 5 The maximum, the average, and the minimum Courant numbers of the computational domain  $\mathcal{M}_N^3$  in Table 1; see (42)

$N$	$\Delta t_N = \frac{0.2}{N}$			$\Delta t_N = \frac{0.08}{N}$			$\Delta t_N = \frac{0.04}{N}$		
	$C_M$	$C_a$	$C_m$	$C_M$	$C_a$	$C_m$	$C_M$	$C_a$	$C_m$
1	12.58	3.29	1.05	5.03	1.32	0.42	2.52	0.66	0.21
2	10.81	2.74	0.76	4.32	1.10	0.30	2.16	0.55	0.15
3	9.29	1.37	0.76	3.72	0.55	0.30	1.86	0.27	0.15
4	10.33	2.21	0.69	4.13	0.88	0.28	2.07	0.44	0.14

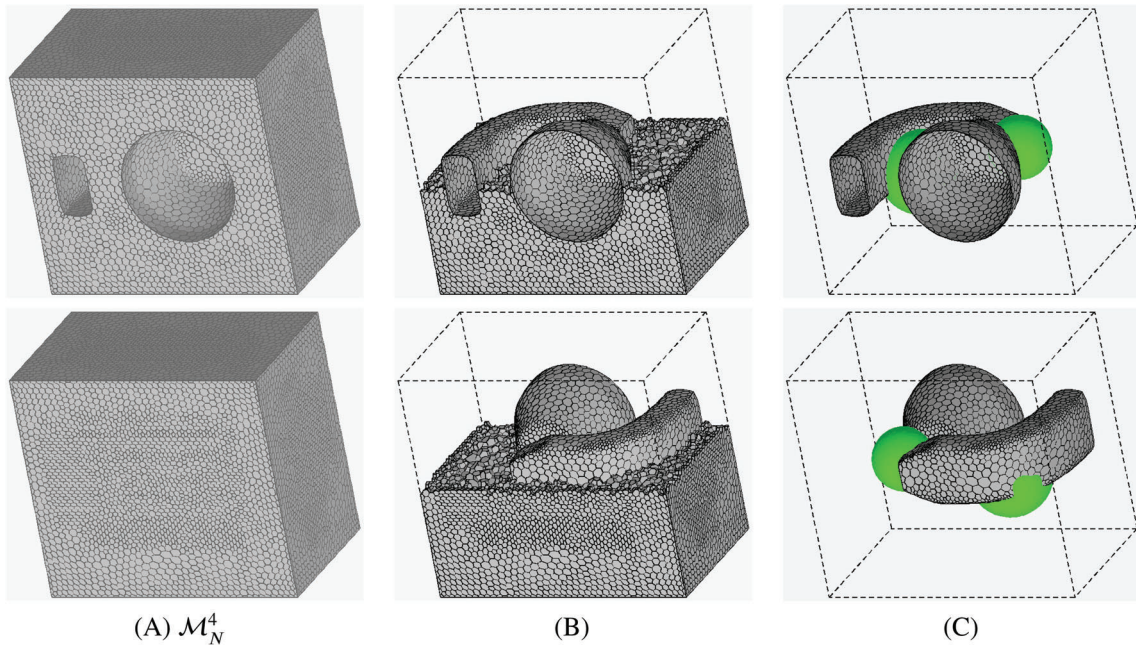


FIGURE 12 The first and second rows are the view from the top and bottom of the computational domain  $\mathcal{M}_N^4$ ,  $N = 1$ , in Table 1, respectively. (A) is the outer part of the domain. (B) is the inner part of the domain. (C) presents the green spheres which are the given surface  $\Lambda$  in (8)

in Figure 12C from the equation of spheres in (38):

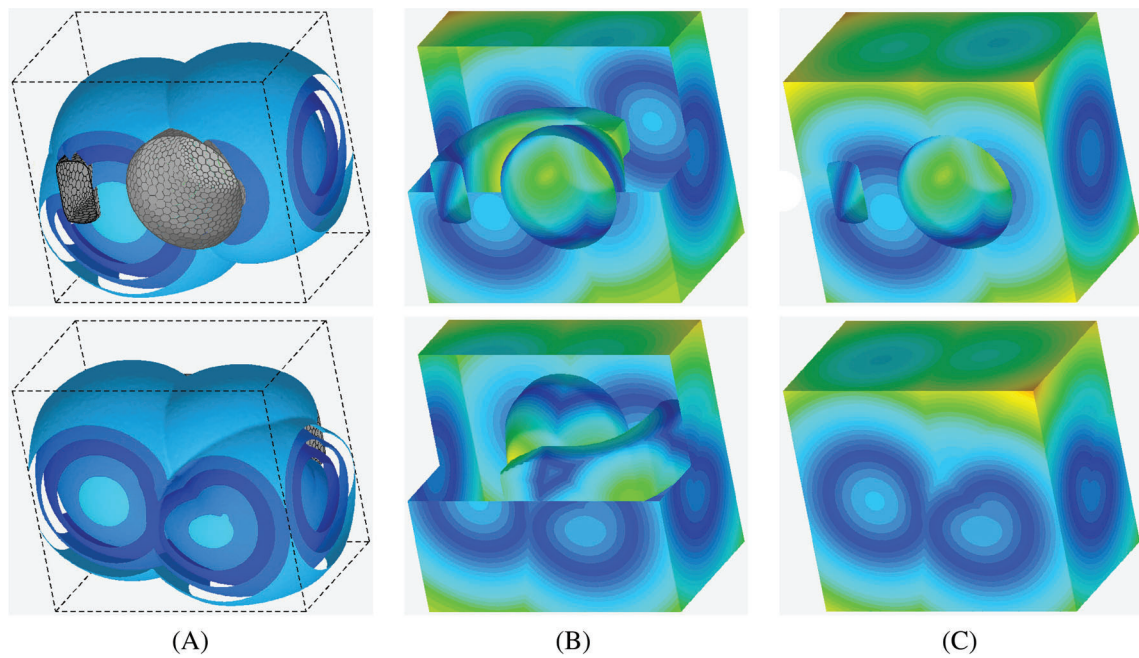
$$\Lambda = \bigcup_{i=1}^2 \{ \mathbf{x} \in \Omega : S_{r_i, \mathbf{c}_i}(\mathbf{x}) = 0 \}, \tag{43}$$

where  $r_1 = 0.3$ ,  $\mathbf{c}_1 = (-0.6, -0.1, 0)$ ,  $r_2 = 0.4$ ,  $\mathbf{c}_2 = (0.4, 0, -0.2)$  on the computational domain  $\Omega = \mathcal{M}^4$  in Figure 12A. The numerical results at  $T = 3$  for the proposed algorithm (29) are presented as the iso-surfaces and their profiles on the boundary of the domain in Figure 13. Since the exact solution cannot be given as an analytic function for a general shape of the computational domain, we use the example to check how well the proposed algorithm converges to the same numerical solution regardless of valid initial conditions. Let us denote the numerical solution  $\phi_N^i$ ,  $i \in \{1, 2, 3\}$ , for the initial function  $\psi_{i-2}$  in (36) on the computational domain  $\mathcal{M}_N^4$ . Then, we measure the average of differences between three numerical solutions:

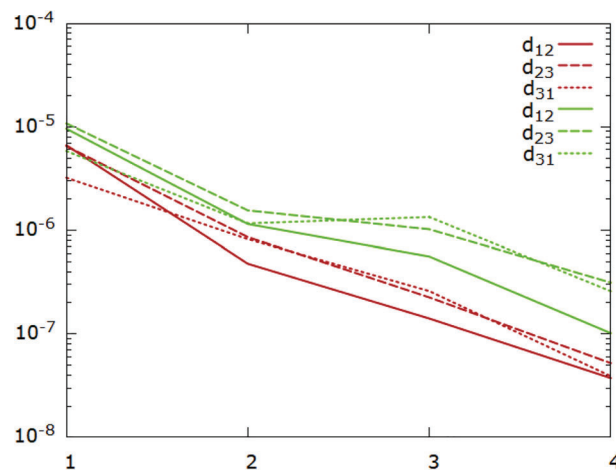
$$d_{ij} = \frac{1}{|\Omega|} \int_{\Omega} |\phi_N^i(\mathbf{x}) - \phi_N^j(\mathbf{x})| dV, \quad i \neq j, \quad i, j \in \{1, 2, 3\}, \quad N \in \{1, 2, 3, 4\}. \tag{44}$$

In Figure 14, the log-scaled graph of  $d_{ij}$  is presented over  $N$ . The green and red curves are the results of using  $\beta_f$  in (22) and  $\bar{\beta}_f$  in (33), respectively. When the average of cell size is smaller, the difference (44) is smaller. It means that the numerical solutions for different initial conditions are converging to the same solution.





**FIGURE 13** The first and second rows are the same view as Figure 12. (A) The iso-surfaces of the numerical result are presented by the proposed algorithm (29) at  $T = 3$  from the green surface in Figure 12C and the initial condition  $\psi_0$  in (36). (B,C) are the iso-surfaces on the boundary surface of the computational domain. (B) shows the inner part of (C)



**FIGURE 14** The average of difference between numerical solutions for different initial conditions (36) over the level  $N$  of the computational domain  $\mathcal{M}_N^4$  in Table 1 is presented; see the more details in (44). The green and red curves are the results of using  $\beta_f$  in (22) and  $\bar{\beta}_f$  in (33), respectively

## 5 | CONCLUSION

We present the cell-centered finite volume method with the Sonner boundary condition to compute the signed distance function from a given surface in the 3D computation domain. Due to the field of applications in industrial problems, the boundary of the domain is considered to be of a very general shape and the numerical algorithm has to robustly work on general 3D polyhedral meshes. By presented numerical experiments, we see that such requirements are fulfilled for the proposed method. The Sonner boundary condition has the important role to obtain the viscosity solution of the eikonal equation. The  $EOC$  in  $L^1$  and  $L^\infty$  norms is numerically shown to be the second order for smooth solutions. Moreover, the straightforward implementation of the Sonner boundary condition is possible for any existing codes based on the

cell-centered finite volume method. From the numerical investigation, the restriction of the norm of the gradient can be useful for general polyhedral meshes and various computational domains.

## ACKNOWLEDGMENTS

The authors would like to thank Dr. Dirk Martin in AVL List GmbH, Graz, Austria, for general advice to generate 3D polyhedral meshes for diverse industrial purposes. The authors sincerely appreciate Dr. Belyaev from Heriot-Watt University, United Kingdom and Dr. Fayolle from the University of Aizu, Japan sharing their comments and extra numerical verification for the first order of convergence to compute the wall distance function by the alternating direction method of multipliers. The authors also thank Dr. Tucker from Cambridge University, the UK for sharing his kind opinion on the second order of convergence in the transport form of the eikonal equation for capturing linear distance and Dr. Byungjoon Lee from the Catholic University of Korea, South Korea for commenting on the redistancing problem by using the Hopf-Lax formula. The work was supported by the grants VEGA 1/0709/19, VEGA 1/0436/20, and APVV-19-0460.

## CONFLICT OF INTEREST

The authors declare no potential conflict of interests.

## DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

## ORCID

Jooyoung Hahn  <https://orcid.org/0000-0003-4357-1009>

## REFERENCES

- Sethian JA. *Level set methods and fast marching methods, evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*. Cambridge University Press; 1999.
- Osher S, Fedkiw R. *Level set methods and dynamic implicit surfaces*. Springer; 2000.
- Gibou F, Fedkiw R, Osher S. A review of level-set methods and some recent applications. *J Comput Phys*. 2018;353:82-109.
- Peters N. *Turbulent combustion*. Cambridge Monographs on Mechanics, Cambridge University Press; 2000.
- Suckart D, Linse D, Schutting E, Eichlseder H. Experimental and simulative investigation of flame-wall interactions and quenching in spark-ignition engines. *Autom Engine Technol*. 2017;2(1):25-38.
- Manz A. *Modeling of end-gas autoignition for knock prediction in gasoline engines*. Logos Verlag; 2016.
- Sussman M, Smereka P, Osher S. A level set approach for computing solutions to incompressible two-phase flow. *J Comput Phys*. 1994;114:146-159.
- Fares E, Schröder W. A differential equation for approximate wall distance. *Int J Numer Methods Fluids*. 2002;39:743-762.
- Tucker PG. Differential equation-based wall distance computation for DES and RANS. *J Comput Phys*. 2003;190:229-248.
- Tucker PG. Hybrid Hamilton-Jacobi-Poisson wall distance function model. *Comput Fluids*. 2011;44:130-142.
- Roget B, Sitaraman J. Wall distance search algorithm using voxelized marching spheres. *J Comput Phys*. 2013;241:76-94.
- Calakli F, Taubin G. SSD: smooth signed distance surface reconstruction. *Comput Graph Forum*. 2011;30(7):1993-2002.
- Zollhöfer M, Dai A, Innmann M, et al. Shading-based refinement on volumetric signed distance functions. *ACM Trans Graph*. 2015;34(4):1-14.
- Xia H, Tucker PG. Finite volume distance field and its application to medial axis transforms. *Int J Numer Methods Eng*. 2010;82:114-134.
- Cadena C, Carlone L, Carrillo H, et al. Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. *IEEE Trans Robot*. 2016;32:1309-1332.
- Biswas A, Shapiro V, Tsukanov I. Heterogeneous material modeling with distance fields. *Comput Aided Geometr Des*. 2004;21:215-242.
- Lee Y, Lim C, Ghazialam H, Vardhan H, Eklund E. Surface mesh generation for dirty geometries by the Cartesian shrink-wrapping technique. *Eng Comput*. 2010;26:377-390.
- Perić M. Flow simulation using control volumens of arbitrary polyhedral shape. In: Hirsch CH, Laurence D, eds. *ERCOFTAC Bulletin*. 2004;62:25-29.
- Crandall MG, Lions PL. Two approximations of solutions of Hamilton-Jacobi equations. *Math Comput*. 1984;43:1-19.
- Tsitsiklis JN. Efficient algorithms for globally optimal trajectories. *IEEE Trans Automat Contr*. 1995;40:1528-1538.
- Sethian JA. A fast marching level set method for monotonically advancing fronts. *Proc Natl Acad Sci*. 1996;93:1591-1595.
- Sethian JA, Vladimirsky A. Ordered upwind methods for static Hamilton-Jacobi equations: theory and algorithms. *SIAM J Numer Anal*. 2004;41:325-363.
- Zhao HK. Fast sweeping method for eikonal equations. *Math Comput*. 2005;74:603-627.
- Kao CY, Osher S, Tsai YH. Fast sweeping methods for static Hamilton-Jacobi equation. *SIAM J Numer Anal*. 2005;42:2612-2632.
- Qian JL, Zhang YT, Zhao HK. Fast sweeping methods for eikonal equations on triangular meshes. *SIAM J Numer Anal*. 2007;31:83-107.
- Qian JL, Zhang YT, Zhao HK. A fast sweeping method for static convex Hamilton-Jacobi equations. *J Sci Comput*. 2007;31:237-271.



27. Gremaud PA, Kuster CM. Computational study of fast methods for the eikonal equation. *SIAM J Sci Comput*. 2006;27:1803-1816.
28. Hysing SR, Turek S. The eikonal equation: numerical efficiency vs. algorithmic complexity on quadrilateral grids. *Proceedings of ALGORITMY 2005, 17th Conference on Scientific Computing*. 2005;22-31.
29. Franzone PC, Guerri L. Spreading of excitation in 3-D models of the anisotropic cardiac tissue. I. validation of the eikonal model. *Math Biosci*. 1993;113:145-209.
30. Bourgine P, Frolkovič P, Mikula K, Peyriéras N, Remešiková M. Extraction of the intercellular skeleton from 2D images of embryogenesis using eikonal equation and advective subjective surface method. In: Tai X.-C., Knut M, Marius L, Knut-Andreas L, eds. *Scale Space and Variational Methods in Computer Vision, SSVN 2009*. Springer; 2009:38-49.
31. Varadhan SRS. On the behavior of the fundamental solution of the heat equation with variable coefficients. *Commun Pure Appl Math*. 1967;20:431-455.
32. Gurumoorthy KS, Rangarajan A. A schrödinger equation for the fast computation of approximate Euclidean distance functions. In: Tai X-C, Knut M, Marius L, Knut-Andreas L, eds. *Scale Space and Variational Methods in Computer Vision, SSVN 2009*. Springer; 2009:100-111.
33. Belyaev AG, Fayolle PA. On variational and PDE-based distance function approximations. *Comput Graph Forum*. 2015;34(8):104-118.
34. Tomlinson KA, Hunter PJ, Pullan AJ. A finite element method for an eikonal equation model of myocardial excitation wavefront propagation. *SIAM J Appl Math*. 2002;63:324-350.
35. Glowinski R. *Variational methods for the numerical solution of nonlinear elliptic problems*. Society for Industrial and Applied Mathematics; 2015.
36. Crane K, Weischedel C, Wardetzky M. Geodesics in heat: a new approach to computing distance based on heat flow. *ACM Trans Graph*. 2013;32(5):1-11.
37. Crane K, Weischedel C, Wardetzky M. The heat method for distance computation. *Commun ACM*. 2017;60:90-99.
38. Fayolle PA, Belyaev AG.  $p$  Laplacian diffusion for distance function estimation, optimal transport approximation, and image enhancement. *Comput Aided Geometr Des*. 2018;67:1-20.
39. Fayolle PA, Belyaev AG. An ADMM-based scheme for distance function approximation. *Numer Algorithms*. 2020;84:983-996.
40. Royston M, Pradhana A, Lee B, et al. Parallel redistancing using the Hopf–Lax formula. *J Comput Phys*. 2018;365:7-17.
41. Lee B, Darbon J, Osher S, Kang M. Revisiting the redistance problem using the Hopf–Lax formula. *J Comput Phys*. 2017;330:268-281.
42. 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*. 2017;72:442-465.
43. 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*. Springer International Publishing; 2017:81-89.
44. Prados E, Faugeras O. Unifying approaches and removing unrealistic assumptions in shape from shading: mathematics can help. In: Matas J, Pajdla T, eds. *Computer Vision - ECCV*. Vol 2004. Springer; 2004:141-154.
45. Prados E, Faugeras O. Shape from shading. In: Paragios N, Chen Y, Faugeras O, eds. *Handbook of Mathematical Models in Computer Vision*. Springer; 2006:375-388.
46. Deckelnick K, Elliott CM, Styles V. Numerical analysis of an inverse problem for the eikonal equation. *Numer Math*. 2011;119:245-269.
47. Dumett MA, Ospino JE. Mimetic discretization of the eikonal equation with Soner boundary conditions. *Appl Math Comput*. 2018;335:25-37.
48. Rocca GD, Blanquart G. Level set reinitialization at a contact line. *J Comput Phys*. 2014;265:34-49.
49. 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. *Comp Math Appl*. 2019;77:1639-1654.
50. Rouy E, Tourin A. A viscosity solutions approach to shape-from-shading. *SIAM J Numer Anal*. 1992;29:867-884.
51. 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*. 2010;32:1527-1544.
52. Adalsteinsson D, Sethian JA. The fast construction of extension velocities in level set methods. *J Comput Phys*. 1999;148:2-22.

**How to cite this article:** Hahn J, Mikula K, Frolkovič P, Basara B. Finite volume method with the Soner boundary condition for computing the signed distance function on polyhedral meshes. *Int J Numer Methods Eng*. 2022;123(4):1057-1077. doi: 10.1002/nme.6888