HIGH ORDER NUMERICAL METHODS TO THREE DIMENSIONAL DELTA FUNCTION INTEGRALS IN LEVEL SET METHODS

XIN WEN†

Abstract. In this paper we propose a class of high order numerical methods to delta function integrals appearing in level set methods in the three dimensional case by extending the idea for designing high order methods to two dimensional delta function integrals in [X. Wen, J. Comput. Phys., 228 (2009), pp. 4273–4290]. The methods comprise approximating the mesh cell restrictions of the delta function integral. In each mesh cell the three dimensional delta function integral can be rewritten as a two dimensional ordinary integral with the smooth integrand being a one dimensional delta function integral. A key idea in designing high order methods in this paper is that the high order accuracy of the methods is not ensured in each mesh cell, and the methods are designed in a consistent way to foster error cancelations in mesh cells where high order accuracy is not ensured in the individual cell. This issue is essentially related to the construction of integral area for the two dimensional ordinary integral in each mesh cell, which is achieved by considering the intersection points between the zero level set and the edges or sides of the mesh cell. The mesh cell restrictions of the three dimensional delta function integral are then approximated by applying standard two dimensional high order numerical quadratures and high order numerical methods to one dimensional delta function integrals. Numerical examples are presented showing that our methods in this paper achieve or exceed the expected second to fourth order accuracy and demonstrating the stability of shifting mesh for our methods and the advantage of using our high order numerical methods.

Key words. delta function integral, high order numerical method, level set method

AMS subject classification. 65JXX

DOI. 10.1137/090758295

1. Introduction. We study in this paper a class of high order numerical methods to the following type of delta function integrals

\[
\int_{\mathbb{R}^n} f(x) \|\nabla u(x)\| \delta(u(x)) dx,
\]

where \( f(x) \) is a weight function and \( u(x) \) is a level set function whose zero points define a manifold \( \Gamma \) of codimension one. In a recent paper we have studied the two dimensional case \( n = 2 \) [45]. In this paper we consider the three dimensional case \( n = 3 \) in which \( \Gamma \) is a two dimensional surface. The functions \( f(x), u(x) \) are assumed to have sufficient smoothness, and their values are only provided at grid points of a regular mesh. Numerical evaluations of delta function integrals (1.1) in two and three dimensions in the above context appear in many applications of level set methods; see, for example, [47, 29, 28, 6, 5].

One natural approach to approximate (1.1) is the numerical quadrature approach. Assume the values of \( f(x), u(x) \) are given at grid points of the following uniform mesh on \( \mathbb{R}^3 \):

\[
\{x_j\}_{j \in \mathbb{Z}^3}, \quad x_j = (x_{j1}^{(1)}, x_{j2}^{(2)}, x_{j3}^{(3)}),
\]
Then the numerical quadrature methods to delta function integrals (1.1) in three dimensions can be generally written in the form

$$h^3 \sum_{j \in \mathbb{Z}^3} f(x_j) \|\nabla_D u(x_j)\| \tilde{\delta}(x_j; u),$$

where $\nabla_D u(x_j)$ is the difference approximation to $\nabla u(x_j)$ and $\tilde{\delta}(x_j; u)$ is an approximate delta function which can be defined by grid point values of $u$. Constructing approximate delta functions is related to the regularization technique of the Dirac delta function. This problem has a wide range of applications, including the immersed boundary method [30, 31], computations of multiphase flows [41, 34, 32, 35, 40, 28], computational high-frequency waves [10, 11, 12, 13, 14, 9, 19, 20, 21, 15, 16, 23, 17, 18, 46], numerical approximations to differential equations with singular source terms [36, 37], discrete singular convolution algorithm for analyzing plate structures [42], dendritic solidification [22, 28], active contours in image processing [28], discretization in the vortex method [7, 4], and so on. In some of these applications, the issue of regularizing delta functions can be transformed into computing delta function integrals. For example, in treating the PDE with delta source term, by using the finite element method instead of the finite difference method, one needs to evaluate delta function integrals in the variational form. Thus the high order methods to delta function integrals considered in this paper can be possibly applied to these problems. The efficiency of the numerical quadratures (1.3) depends on appropriate choice of the approximate delta functions. However, in [37] Tornberg and Engquist showed that a common technique to construct the approximate delta function in (1.3) suffers from $O(1)$ errors. After that, different approximate delta functions have been designed in the literature which yield first or second order numerical quadratures (1.3). In [8] Engquist, Tornberg, and Tsai proposed a first order accurate approximate delta function based on one dimensional discrete delta functions and a variable support size formula. They also proposed in [8] a second order accurate approximate delta function based on approximations to product formula for multidimensional delta functions which is more complex to apply. The product formula method following Peskin [30, 31] has the advantage to achieve any desired accuracy by using one dimensional discrete delta functions with corresponding discrete moment conditions, as proved in [37]. However, the implementation of the high order version of this method in the case of $\Gamma$ being implicitly represented by a level set function is nontrivial and has not been reported yet. In [33] Smereka proposed both a first and second order accurate approximate delta function by using a technique for solving elliptic equations with discontinuous source terms. The proof of accuracy of this approximate delta function is given in [2]. In [38] Towers proposed both a first and second order accurate approximate delta function by using difference approximations to derivatives of the smoothed Heaviside function or those of the integral of the Heaviside function. The analysis of accuracy of these methods is given in [39]. However, approximate delta functions higher than second order accuracy for the numerical quadratures (1.3) have not been proposed and implemented yet and remain to be studied.

In a recent paper [45] we have designed and analyzed second to fourth order numerical methods to the delta function integrals (1.1) in two dimensions. The strategy of the methods in [45] is different from the numerical quadrature approach. These methods were constructed by considering the approximation of the restriction of the
two dimensional delta function integral in each mesh cell. This is also a natural strategy. By using this strategy, Min and Gibou designed in [25, 26] a second order geometric integration method for computing (1.1) via decomposing the zero level set \( \Gamma \) into simplices. Such a strategy has also been applied to construct second to fourth order numerical methods to another type of delta function integrals [43],

\[
\int_{\mathbb{R}^n} \alpha(x) \prod_{i=1}^{n} \delta(\beta_i(x)) \, dx, \quad n = 1, 2, 3,
\]

where the common zero points of the level set functions \( \beta_i(x) \) are essentially finite number of points in the space. The idea in [45] to approximate the two dimensional delta function integral restricted in mesh cells is based on that the two dimensional delta function integral in a mesh cell can be rewritten as a one dimensional ordinary integral with the smooth integrand being a one dimensional delta function integral. The form of the transformed one dimensional ordinary integral has two options according to the comparison of the components of gradient of the level set function. Therefore high order numerical methods to approximate the mesh cell restriction of the delta function integral (1.1) in two dimensions in principle can be constructed by applying standard high order numerical quadratures to one dimensional ordinary integrals and high order numerical methods to one dimensional delta function integrals which have been studied [43, 44]. The key point in designing high order numerical methods to delta function integrals (1.1) is, however, that one cannot ensure the high order accuracy for approximating the delta function integral restricted in each mesh cell. Therefore the strategy for designing high order methods in [45] is that for the cell in which the delta function integral is not approximated accurately enough, there exist the neighboring cells so that the approximate delta function integral restricted to the union of the cells has high order accuracy. The achievement of this strategy in [45] is essentially related to suitable determination of the integral interval for the transformed one-dimensional ordinary integral. The integral intervals are designed in a consistent way to ensure that when an integral interval for a mesh cell is not accurate enough, the union of integral intervals for neighboring cells can achieve high order accuracy.

In this paper we extend the idea in [45] to design high order numerical methods to the delta function integral (1.1) in three dimensions. Similarly the three dimensional delta function integral in a mesh cell can be rewritten as a two dimensional ordinary integral with the smooth integrand being a one dimensional delta function integral. The transformed two dimensional integral takes one of three forms according to the comparison of the components of the gradient of \( u \) in the cell. We need to consider the approximation of the integral area of the two dimensional ordinary integral. Once this is achieved, we can use high order numerical quadratures to approximate the two dimensional ordinary integral with the evaluation of the integrand resorting to high order numerical methods to one dimensional delta function integrals. The key idea in designing high order methods in this paper, as in [45], is that we do not intend to design high order methods for computing the delta function integral restricted in each mesh cell. The high order accuracy of our methods is achieved by expecting the ensured accuracy in the union of certain cells, while the approximation in the individual cell may not be accurate enough. The achievement of this strategy is also essentially related to the construction of the integral area for the transformed two dimensional ordinary integral. In this paper we construct the integral area in each mesh cell by considering the projection of the intersection points between the zero level
set and the edges of the mesh cell and connecting these projecting intersection points using suitable procedure. Moreover in order to ensure the high order accuracy of the constructed integral area for most common cases, we also need to take into account the curved boundary of the integral area. This is achieved by considering the intersection points between the zero level set and the sides of the mesh cell. Numerical examples in this paper show that our numerical methods succeed in achieving the expected second to fourth order accuracy for general practical computations, indicating the algorithms designed in this paper achieve the effective error cancelations in the union of neighboring cells in the situation that the approximation in an individual cell is not accurate enough. The analysis of this property and the convergence rate estimates for our methods, however, are not straightforward and require further study in the future. We summarize that the design principle for our methods in this paper and in [45] is starting from transforming the delta function integral (1.1) into a lower dimensional ordinary integral with smooth integrand being a one dimensional delta function integral according to the comparison of the components of the gradient of the level set function. Therefore the computation of high dimensional delta function integrals is decomposed into computing lower dimensional integrals to be treated using already established techniques. With this strategy we developed high order methods to the delta function integrals (1.1) of codimension one type.

This paper is organized as follows. In section 2 we discuss the main strategy, numerical implementation, and algorithm description of the method proposed in this paper. In section 3 we present numerical examples in which our methods are shown to achieve the expected second to fourth order accuracy. Stability of shifting mesh for our methods and efficiency of using high order numerical methods are also demonstrated. We conclude the paper in section 4.

2. High order numerical methods.

2.1. Main strategy. Consider the delta function integral (1.1) in three dimensional case

$$\int_{\mathbb{R}^3} f(x, y, z) \sqrt{(u_x)^2 + (u_y)^2 + (u_z)^2} \delta(u(x, y, z)) dx dy dz.$$  

Assume $\mathbb{R}^3$ is covered by a uniform mesh $(x_i, y_j, z_k), (i, j, k) \in \mathbb{Z}^3$, with the mesh size $h$. Denote

$$I_{i,j,k} = \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} \int_{z_k}^{z_{k+1}} f(x, y, z) \sqrt{(u_x)^2 + (u_y)^2 + (u_z)^2} \delta(u(x, y, z)) dx dy dz.$$  

Then

$$\int_{\mathbb{R}^3} f(x, y, z) \sqrt{(u_x)^2 + (u_y)^2 + (u_z)^2} \delta(u(x, y, z)) dx dy dz = \sum_{(i,j,k) \in \mathbb{Z}^3} I_{i,j,k}.$$  

Therefore the delta function integral (2.1) can be computed by approximating each $I_{i,j,k}$. Let $\hat{I}_{i,j,k}$ be the approximation to $I_{i,j,k}$. Then our goal is to provide an algorithm to yield $\hat{I}_{i,j,k}$.

Denote $u_{l,m,n} = u(x_l, y_m, z_n) \forall (l, m, n) \in \mathbb{Z}^3$. Consider the cell $C_{i,j,k} = [x_i, x_{i+1}] \times [y_j, y_{j+1}] \times [z_k, z_{k+1}]$. If the eight vertex values of $u$ in this cell have the same sign, then this is the case that the cell has no or slight intersection with the zero level set $\Gamma$ of $u$. In this case, naturally, we set $\hat{I}_{i,j,k} = 0$. The essential part of
computing \( \hat{I}_{i,j,k} \) is to treat the nontrivial case that the eight vertex values of \( u \) in the cell do not all have the same sign.

Now consider the computation of \( \hat{I}_{i,j,k} \) in the nontrivial case. Observe that the three dimensional delta function integral (2.2) can be formally rewritten as the following three forms:

\[
(2.4) \quad I_{i,j,k} = \int_{[x_i, x_{i+1}] \times [y_j, y_{j+1}]} \left( \int_{z_k}^{z_{k+1}} f(x, y, z) \sqrt{(u_x)^2 + (u_y)^2 + (u_z)^2} \delta(u(x, y, z)) dz \right) dxdy,
\]

\[
(2.5) \quad I_{i,j,k} = \int_{[x_i, x_{i+1}] \times [z_k, z_{k+1}]} \left( \int_{y_j}^{y_{j+1}} f(x, y, z) \sqrt{(u_x)^2 + (u_y)^2 + (u_z)^2} \delta(u(x, y, z)) dy \right) dxdz,
\]

\[
(2.6) \quad I_{i,j,k} = \int_{[y_j, y_{j+1}] \times [z_k, z_{k+1}]} \left( \int_{x_i}^{x_{i+1}} f(x, y, z) \sqrt{(u_x)^2 + (u_y)^2 + (u_z)^2} \delta(u(x, y, z)) dx \right) dydz.
\]

Namely, the three dimensional delta function integral \( I_{i,j,k} \) can be regarded as a two dimensional ordinary integral with the integrand being a one dimensional delta function integral. The strategy in this paper to approximate \( I_{i,j,k} \), similarly to that adopted for the two dimensional case \[45\], is to transform the three dimensional delta function integral into the two dimensional ordinary integral illustrated by one of the forms (2.4) to (2.6), with the integrand being the one dimensional delta function integral. Therefore high order numerical methods for approximating \( I_{i,j,k} \) can be expected by applying standard two dimensional high order numerical quadratures and high order numerical methods to one dimensional delta function integrals.

The above strategy relies on the validity of the three forms (2.4) to (2.6) of \( I_{i,j,k} \). As described in the next subsection, for a well-defined level set function \( u \), at least one of the three forms (2.4) to (2.6) is valid, which can be determined by comparing the discrete version of the component of the gradient of \( u \). For the convenience of description, we discuss the case that the form (2.4) is valid. Denote

\[
(2.7) \quad \alpha(x, y, z) = f(x, y, z) \sqrt{(u_x)^2 + (u_y)^2 + (u_z)^2},
\]

\[
(2.8) \quad F(x, y) = \int_{z_k}^{z_{k+1}} \alpha(x, y, z) \delta(u(x, y, z)) dz, \quad (x, y) \in [x_i, x_{i+1}] \times [y_j, y_{j+1}].
\]

Then (2.4) is written to be

\[
(2.9) \quad I_{i,j,k} = \int_{[x_i, x_{i+1}] \times [y_j, y_{j+1}]} F(x, y) dxdy.
\]

However, it is easy to check that \( F(x, y) \) can be discontinuous on \([x_i, x_{i+1}] \times [y_j, y_{j+1}]\) (see Figure 2.1 in [45], which illustrates the similar situation for two dimensional case). Therefore it is improper to directly apply numerical quadrature to
(2.9) in order to obtain high order numerical results. Similarly to treating the two dimensional case in [45], we can transform (2.9) into the two dimensional integral with smooth integrand as follows. Since we are discussing that the form (2.4) is valid, one has \( u_z = O(1) \) away from zero near the cell \( C_{i,j,k} \). Thus we can introduce the definitions

\[
Z_{i,j,k}(x, y) \text{ satisfying that } (x, y, Z_{i,j,k}(x, y)), (x, y) \in [x_i, x_{i+1}] \\
\times [y_j, y_{j+1}] \text{ are points on } \Gamma,
\]

\[
Z_{i,j,k}^m = \min_{(x,y) \in [x_i, x_{i+1}] \times [y_j, y_{j+1}]} Z_{i,j,k}(x, y),
\]

\[
Z_{i,j,k}^M = \max_{(x,y) \in [x_i, x_{i+1}] \times [y_j, y_{j+1}]} Z_{i,j,k}(x, y),
\]

\[
\tilde{F}(x, y) = \int_{Z_{i,j,k}^m - h}^{Z_{i,j,k}^M + h} \alpha(x, y, z) \delta(u(x, y, z)) dz, (x, y) \in [x_i, x_{i+1}] \times [y_j, y_{j+1}],
\]

\[
\hat{\Omega}_{i,j,k} = \{ (x, y) \in [x_i, x_{i+1}] \times [y_j, y_{j+1}] | Z_{i,j,k}(x, y) \in [z_k, z_{k+1}) \}.
\]

The function \( \tilde{F}(x, y) \) is smooth on \([x_i, x_{i+1}] \times [y_j, y_{j+1}]\) and thus on \( \hat{\Omega}_{i,j,k} \). We have the following equality:

\[
I_{i,j,k} = \int_{[x_i, x_{i+1}] \times [y_j, y_{j+1}]} F(x, y) dxdy = \int_{\hat{\Omega}_{i,j,k}} F(x, y) dxdy = \int_{\hat{\Omega}_{i,j,k}} \tilde{F}(x, y) dxdy.
\]

Thus by restricting the integral area, we transform \( I_{i,j,k} \) into the two dimensional integral with the smooth integrand \( \tilde{F}(x, y) \). In fact since the three dimensional delta function integral \( I_{i,j,k} \) represents surface integral, (2.14) implies we transform the surface integral into the equivalent form of the integral in \((x, y)\)-plane. Thus our strategy to approximate \( I_{i,j,k} \) in the case of form (2.4) being valid is to apply high order numerical quadratures to the last form in (2.14). This strategy requires constructing approximation to the integral area \( \hat{\Omega}_{i,j,k} \) and the computation of the values of \( \tilde{F} \) corresponding to computing one dimensional delta function integrals. The detailed strategy and numerical implementation for these issues will be discussed in the next subsection.

As described in the introduction, our high order method does not ensure the high order accuracy for computing \( I_{i,j,k} \) for each mesh cell. In fact considering that geometrically it is highly complicated to completely classify the cases that a smooth surface intersects with a cubic mesh cell and determine these cases with only the grid point values of the level set function, we do not aim to provide a method which is capable of computing \( I_{i,j,k} \) with high order accuracy for all mesh cells. Instead, the strategy of our method is to give high order accuracy for computing \( I_{i,j,k} \) for mesh cells of common cases. For certain mesh cells for which high order accuracy of computing \( I_{i,j,k} \) is not achieved, our method is designed to facilitate proper error cancelations between the mesh cell and its neighboring ones to expect the high order accuracy for approximating the delta function integral in the union of neighboring cells. The effect of this strategy is confirmed by numerical examples in this paper which show that our method achieves high order accuracy for general numerical computations.

With above discussions, the main strategy of our high order method to yield the approximate mesh cell restriction of the delta function integral \( \hat{I}_{i,j,k} \) is given as
follows. We only need to treat the nontrivial case that the eight vertex values of \( u \) in the cell \( C_{i,j,k} \) do not all have the same sign; otherwise, we set \( \tilde{I}_{i,j,k} = 0 \). In the nontrivial cases, we check which one of the three forms (2.4) to (2.6) is valid. In the case that the form (2.4) is valid, \( \tilde{I}_{i,j,k} \) is yielded by applying the high order numerical quadrature to the last form in (2.14) which consists of constructing approximation to the integral area \( \Omega_{i,j,k} \) and computing the values of \( \tilde{F} \) corresponding to evaluating one dimensional delta function integrals. Similar strategy can be applied in the case that the form (2.5) or (2.6) is valid.

2.2. Numerical implementation. In this subsection we describe the numerical implementation details for our method.

First we discuss the check of the validity of the three forms (2.4) to (2.6) of \( I_{i,j,k} \). This means the validity of the one dimensional delta function integrals in the three forms. Similar to the situation in the two dimensional case, we see that the validity of the three forms (2.4) to (2.6) depends on that \( u_z, u_y, \) and \( u_x \) at zero points of \( u(x,y,z) \) near the cell \( C_{i,j,k} \) are nonzero, respectively. Since \( \sqrt{(u_x)^2 + (u_y)^2 + (u_z)^2} \) is the normal derivative of \( u(x,y,z) \) at its zero level set, it should have \( O(1) \) positive lower bound at the zero level set for a well-defined level set function \( u \). Therefore at the zero points of \( u(x,y,z) \) at least one of \( u_x, u_y, \) and \( u_z \) should be \( O(1) \) away from zero. This implies that at least one of the three forms (2.4) to (2.6) is valid. To determine which of the three forms is valid, we can compare the absolute value of \( u_x, u_y, \) and \( u_z \) at the center point \((\frac{x_i+x_{i+1}}{2}, \frac{y_j+y_{j+1}}{2}, \frac{z_k+z_{k+1}}{2})\) of the cell \( C_{i,j,k} \). Since we only have the grid point values of \( u \), in practical computations we can compare the central difference of \( u_x, u_y, \) and \( u_z \). Namely, we consider the quantities

\[
\tilde{u}^x_{i,j,k} = \left| \sum_{j'=0}^{1} \sum_{k'=0}^{1} (u_{i+1,j+j',k+k'} - u_{i,j,j',k+k'}) \right|,
\]

\[
\tilde{u}^y_{i,j,k} = \left| \sum_{i'=0}^{1} \sum_{k'=0}^{1} (u_{i+i',j+1,k+k'} - u_{i,i',j,k+k'}) \right|,
\]

\[
\tilde{u}^z_{i,j,k} = \left| \sum_{j'=0}^{1} \sum_{k'=0}^{1} (u_{i,i',j+j',k+1} - u_{i,i',j+j',k}) \right|.
\]

The case that \( \tilde{u}^x_{i,j,k}, \tilde{u}^y_{i,j,k}, \) or \( \tilde{u}^z_{i,j,k} \) has the maximum value corresponds to that the form (2.4), (2.5), or (2.6) is valid, respectively.

In the following we discuss the case that the form (2.4) is checked to be valid in consistency with the discussion in the last subsection. The case that the form (2.5) or (2.6) is valid can be similarly treated.

Next we discuss the construction of the approximation to the integral area \( \Omega_{i,j,k} \) in (2.13) and numerical quadrature to the last form in (2.14). Let \( \hat{\Omega}_{i,j,k} \) be the approximation to \( \Omega_{i,j,k} \) we will construct. The set \( \hat{\Omega}_{i,j,k} \) is the projection to \((x,y)\)-plane of the zero level set \( \Gamma \) restricted in the cell \( C_{i,j,k} \). Define

\[
\Upsilon_{i,j,k} = \{(x,y,z)|(x,y,z) \text{ being the intersection point between the zero level set } \Gamma \text{ and the edges of the cell } C_{i,j,k}\},
\]

(2.18)

\[
\Upsilon^2_{i,j,k} = \{(x,y)|\exists z \in [z_k, z_{k+1}] \text{ such that } (x,y,z) \in \Upsilon_{i,j,k}\},
\]

(2.19)
(2.20) \( \Upsilon^3_{i,j,k} = \{ x \mid \exists y \in [y_j, y_{j+1}] \text{ such that } (x, y) \in \Upsilon^2_{i,j,k} \} \).

The elements of \( \Upsilon_{i,j,k} \) are at the boundary of the zero level set \( \Gamma \) restricted in the cell \( C_{i,j,k} \). Thus the elements of \( \Upsilon^2_{i,j,k} \) are at the boundary of the area \( \Omega_{i,j,k} \). Therefore the natural strategy to construct \( \widehat{\Omega}_{i,j,k} \) is to connect each point in the set \( \Upsilon^2_{i,j,k} \). A natural way is as follows. Denote the elements of \( \Upsilon^3_{i,j,k} \) to be \( \gamma_1 < \gamma_2 < \cdots < \gamma_\rho \), and define

\[
(2.21) \quad \Phi_1 = \{ y \mid (\gamma_1, y) \in \Upsilon^2_{i,j,k} \}, \quad \Phi_\rho = \{ y \mid (\gamma_\rho, y) \in \Upsilon^2_{i,j,k} \},
\]

\[
(2.22) \quad \eta_{m} = \min_{y \in \Phi_1} y, \quad \eta_{M} = \max_{y \in \Phi_1} y, \quad \eta_{m} = \min_{y \in \Phi_\rho} y, \quad \eta_{M} = \max_{y \in \Phi_\rho} y,
\]

\[
(2.23) \quad \eta_{m} = y_j, \quad \eta_{M} = y_{j+1}, \quad 2 \leq l \leq \rho - 1,
\]

\( \Psi_l \) being the convex hull connected by

\[
(2.24) \quad (\gamma_l, \eta_{m}), (\gamma_l, \eta_{M}), (\gamma_{l+1}, \eta_{m}^{l+1}), (\gamma_{l+1}, \eta_{M}^{l+1}), \quad 1 \leq l \leq \rho - 1.
\]

Then the following area

\[
(2.25) \quad \widehat{\Omega}_{i,j,k} = \bigcup_{l=1}^{\rho-1} \Psi_l
\]

is the reasonable approximation to \( \Omega_{i,j,k} \) in most cases. For example, consider the case shown in the left part of Figure 2.1. In this case the set \( \Upsilon^2_{i,j,k} \) contains five points \( A, B, C, D, E \) shown in the figure. These five points are at the boundary of the area \( \Omega_{i,j,k} \). The area \( \widehat{\Omega}_{i,j,k} \) defined in (2.25) forms the pentagon connected by the five points, which is the suitable approximation to \( \Omega_{i,j,k} \) in this case.

However, the reasonableness of \( \widehat{\Omega}_{i,j,k} \) does not hold for all cases. Consider the case shown in the right part of Figure 2.1. In this case the set \( \Upsilon^2_{i,j,k} \) contains six points \( A, B, C, D, E, F \). The \( x \)-component of \( C \) is between the points \( A \) and \( F \), and the \( x \)-component of \( F \) is between the points \( C \) and \( D \). We see that the suitable approximation to \( \Omega_{i,j,k} \) is the hexagon connected by the six points \( A, B, C, D, E, F \). However,
\[ \tilde{\Omega}_{i,j,k} \] defined in (2.25) forms the hexagon connected by the six points \( A, B, H, D, E, G \), which is unreasonable to approximate \( \Omega_{i,j,k} \). In order to fix the problem, it is seen that we need to adjust \( \eta^M_l \) from \( y_{j+1} \) defined in (2.23) to the \( y \)-component of the intersection point between \( AF \) and \( CG \). For the same reason, we need to adjust \( \eta^m_M \) from \( y_j \) to the \( y \)-component of the intersection point between \( CD \) and \( HF \). Namely, we need to modify the definition of \( \eta^l_m, \eta^l_M, 2 \leq l \leq \rho-1 \), in this case. For discussing this case, we define

\begin{align*}
\zeta_1 &= \{ (x,y) | (x,y,z_k) \in \mathcal{T}_{i,j,k} \}, \quad \zeta_2 = \{ (x,y) | (x,y,z_{k+1}) \in \mathcal{T}_{i,j,k} \},
\end{align*}

and let \( \tau_1, \tau_2 \) be the number of elements in \( \zeta_1, \zeta_2 \), respectively. Then the case shown in the right part of Figure 2.1 belongs to one of the following two cases (2.27) or (2.28):

\begin{align*}
(2.27) & \quad \tau_1 = 2, \tau_2 = 2, \\
(2.28) & \quad \tau_1 \geq 3, \tau_2 = 0 \quad \text{or} \quad \tau_2 \geq 3, \tau_1 = 0.
\end{align*}

First we discuss the case (2.27). In this case let \((x_m^1, y_m^1), (x_m^2, y_m^2)\) be the elements of \( \zeta_m \) for \( m = 1, 2 \), respectively. We assume \( x_1^m \leq x_2^m, m = 1, 2 \). Then we give the modification of \( \eta^l_m, \eta^l_M, 1 \leq l \leq \rho \), by the following procedure.

**Procedure I.**

\begin{align*}
\hat{\eta}^l_m &= \eta^l_m, \quad \hat{\eta}^l_M = \eta^l_M, \quad l = 1, \rho, \\
& \text{for } l \text{ from } 2 \text{ to } \rho - 1 \\
& \text{for } m \text{ from } 1 \text{ to } 2 \\
& \quad \text{if } x_1^m < \gamma_l < x_2^m \\
& \quad \quad \text{let } (\gamma_l, y') \in \mathcal{T}_{i,j,k}^2 \text{ and } y'' = \frac{y_1^m (x_2^m - \gamma_l) + y_2^m (\gamma_l - x_1^m)}{x_2^m - x_1^m} \\
& \quad \quad \text{if } y' < y'' \\
& \quad \quad \quad \hat{\eta}^l_m = y_j, \quad \hat{\eta}^l_M = y'' \\
& \quad \quad \text{else} \\
& \quad \quad \quad \hat{\eta}^l_m = y'', \quad \hat{\eta}^l_M = y_{j+1} \\
& \quad \text{else} \\
& \quad \quad \hat{\eta}^l_m = y_j, \quad \hat{\eta}^l_M = y_{j+1} \\
& \text{end} \\
& \text{end}
\end{align*}

Correspondingly, define

\begin{align*}
\hat{\Psi}_l \text{ being the convex hull connected by} \\
\hat{\mathcal{T}}(\gamma_l, \hat{\eta}^l_m, \gamma_l, \hat{\eta}^l_M, \gamma_{l+1}, \hat{\eta}^{l+1}_m, \gamma_{l+1}, \hat{\eta}^{l+1}_M), \quad 1 \leq l \leq \rho - 1, \\
\hat{\Omega}_{i,j,k} &= \bigcup_{l=1}^{\rho-1} \hat{\Psi}_l.
\end{align*}

One can check that the area \( \hat{\Omega}_{i,j,k} \) so constructed is the suitable approximation to \( \Omega_{i,j,k} \) in the case shown in the right part of Figure 2.1 and when (2.27) holds.

Next we discuss the case (2.28). For example we consider \( \tau_1 \geq 3, \tau_2 = 0 \). In this case the zero level set \( \Gamma \) is nearly parallel to the \((x,y)\)-plane near the cell.
\( C_{i,j,k} \) and close to the cell’s \( z = z_k \) side. Therefore it holds that \( I_{i,j,k} + I_{i,j,k-1} = \int_{[x_i,x_{i+1}] \times [y_j,y_{j+1}]} F(x,y) \, dxdy \). In this case we do not need to define \( \Omega_{i,j,k}, \hat{\Omega}_{i,j,k-1} \) close to \( \Omega_{i,j,k}, \Omega_{i,j,k-1} \), respectively. Instead, we can just let \( \hat{\Omega}_{i,j,k} \) or \( \hat{\Omega}_{i,j,k-1} \) be \([x_i,x_{i+1}] \times [y_j,y_{j+1}]\) and the other be null set. This treatment obviously leads to \( \int_{\hat{\Omega}_{i,j,k}} F(x,y) \, dxdy + \int_{\hat{\Omega}_{i,j,k-1}} F(x,y) \, dxdy = I_{i,j,k} + I_{i,j,k-1} \). Thus the approximate delta function integral in the union of the two cells still can achieve high order accuracy. We notice that \( \tau_1 \geq 3 \) for the cell \( C_{i,j,k} \) implies that \( \tau_2 \geq 3 \) for the cell \( C_{i,j,k-1} \). Therefore we need to treat the cases \( \tau_1 \geq 3, \tau_2 = 0, \) and \( \tau_2 \geq 3, \tau_1 = 0 \), conversely. We can use the following strategy:

\[
(2.31) \quad \hat{\Omega}_{i,j,k} = \begin{cases} [x_i,x_{i+1}] \times [y_j,y_{j+1}], & \tau_1 \geq 3, \\ \varnothing, & \tau_2 \geq 3. \end{cases}
\]

From the above discussions, generally, we can introduce the following strategy to define \( \hat{\eta}_m^l, \hat{\eta}_M^l, 1 \leq l \leq \rho \).

**Strategy I.**

if the cell \( C_{i,j,k} \) belongs to the case (2.27)

define \( \hat{\eta}_m^l, \hat{\eta}_M^l, 1 \leq l \leq \rho \), by Procedure I

else if the cell \( C_{i,j,k} \) belongs to the case (2.28)

if \( \tau_1 \geq 3 \)

let \( \hat{\eta}_m^l = y_j, \hat{\eta}_M^l = y_{j+1} \) for \( 1 \leq l \leq \rho \)

else if \( \tau_2 \geq 3 \)

let \( \hat{\eta}_m^l = y_j, \hat{\eta}_M^l = y_j \) for \( 1 \leq l \leq \rho \)

end

else

let \( \hat{\eta}_m^l = \eta_m^l, \hat{\eta}_M^l = \eta_M^l \), where \( \eta_m^l, \eta_M^l \) are defined in (2.22), (2.23) for \( 1 \leq l \leq \rho \)

end

Then we let the area \( \hat{\Omega}_{i,j,k} \) be constructed by (2.30).

With the above construction of \( \hat{\Omega}_{i,j,k} \), we then can introduce the approximation to the last form in (2.14) as follows. Let \( w_k \) and \( n_k \in [0,1], k = 1,2, \ldots, K \), be the weights and nodal points of an \( S \)th order quadrature rule. For example, the Simpson rule corresponds to \( S = 4, K = 3 \), and \( w_1 = w_3 = \frac{1}{6}, w_2 = \frac{4}{6}, n_1 = 0, n_2 = \frac{1}{2} \), and \( n_3 = 1 \). Then we have the following approximation to \( I_{i,j,k} \):

\[
(2.32) \quad \tilde{I}_{i,j,k} = \sum_{l=1}^{\rho - 1} \left( \gamma_{l+1} - \gamma_l \right) \sum_{k'=1}^{K} w_{k'} \left( \tilde{\eta}_M^{l,k'} - \tilde{\eta}_m^{l,k'} \right) \sum_{k''=1}^{K} w_{k''} \tilde{F} (\tilde{\gamma}_{l,k'}, \tilde{\eta}_{l,k',k''}),
\]

where

\[
(2.33) \quad \tilde{\gamma}_{l,k'} = \gamma_l + n_{k'} (\gamma_{l+1} - \gamma_l), \quad \tilde{\eta}_{l,k',k''} = \tilde{\eta}_m^{l,k'} + n_{k''} \left( \tilde{\eta}_M^{l,k'} - \tilde{\eta}_m^{l,k'} \right),
\]

\[
(2.34) \quad \tilde{\eta}_m^{l,k'} = \tilde{\eta}_m^{l,k'} + n_{k'} \left( \tilde{\eta}_M^{l,k'} - \tilde{\eta}_m^{l,k'} \right), \quad \tilde{\eta}_M^{l,k'} = \tilde{\eta}_M^{l,k'} + n_{k'} \left( \tilde{\eta}_M^{l,k'} - \tilde{\eta}_M^{l,k'} \right).
\]

Although \( \tilde{I}_{i,j,k} \) is the reasonable approximation to \( I_{i,j,k} \), it does not meet the requirement of designing high order numerical methods for computing (2.3). This is because the area \( \hat{\Omega}_{i,j,k} \) may have curved boundary. Using \( \hat{\Omega}_{i,j,k} \) defined by (2.30) and
Strategy I which has flat boundaries to approximate $\Omega_{i,j,k}$ can introduce second order error in the sum of $I_{i,j,k}$ to approximate (2.3). Therefore we need to pay attention to more details. Let us consider the case shown in Figure 2.2. In this case $\Omega_{i,j,k}$ forms the hexagon connected by the six points $A, B, C, D, E, F$. However, $\Omega_{i,j,k}$ contains two curved boundaries connecting $A, F$ and $C, D$. Denote $\Omega_1'$ to be the small area enclosed by the line $AF$ and the curve $\tilde{AGF}$, and $\Omega_2'$ to be the area enclosed by $CD$ and $\tilde{CHD}$. Then

$$\Omega_{i,j,k} = \tilde{\Omega}_{i,j,k} \cup \Omega_2' \setminus \Omega_1'.$$

In order to design high order numerical methods, the small areas $\Omega_1', \Omega_2'$ cannot be omitted. Define the two sides of the cell $C_{i,j,k}$

$$\theta_1 = \{(x, y, z_k)|(x, y, z_k) \in C_{i,j,k}\}, \quad \theta_2 = \{(x, y, z_{k+1})|(x, y, z_{k+1}) \in C_{i,j,k}\}.$$ 

We observe that the curved boundary of $\Omega_{i,j,k}$ corresponds to the intersection curve between the zero level set $\Gamma$ and $\theta_1$ or $\theta_2$. Moreover we only need to treat the curved boundary of $\Omega_{i,j,k}$ in the case that $\tau_1 = 2$ or $\tau_2 = 2$, where $\tau_1, \tau_2$ are defined after (2.26). For example, consider $\tau_1$. If $\tau_1 < 2$, then the side $\theta_1$ does not have an intersection curve with the zero level set $\Gamma$. $\tau_1 \geq 3$ is the case (2.28). As we discussed before, in this case we set $\tilde{\Omega}_{i,j,k}$ to be (2.31). Therefore in this case we do not need to consider the issue of the curved boundary.

Now consider the case $\tau_1 = 2$ or $\tau_2 = 2$. Let $(x_1^m, y_1^m), (x_2^m, y_2^m)$ be the elements of $\zeta_m$ for $m = 1$ or 2. We define the following:

$$\bar{x}^m(s), \bar{y}^m(s) \text{ satisfy } (\bar{x}^m(s), \bar{y}^m(s), z_{k+m-1}) \in \Gamma \text{ and the line connecting } (\bar{x}^m(s), \bar{y}^m(s)) \text{ and } (x_1^m + s(x_2^m - x_1^m), y_1^m + s(y_2^m - y_1^m)) \text{ is vertical to}$$

$$\text{the line connecting } (x_1^m, y_1^m) \text{ and } (x_2^m, y_2^m) \text{ for } s \in [0, 1].$$

$\Omega_{i,j,k}^m$ is the area enclosed by the line segment connecting $(x_1^m, y_1^m)$ and $(x_2^m, y_2^m)$ and the curve segment composed by $(\bar{x}^m(s), \bar{y}^m(s)), s \in [0, 1]$. 

Figure 2.2. Curved boundaries of $\Omega_{i,j,k}$. 
The area $\Omega^m_{i,j,k}, m = 1, 2$, so defined corresponds to the area $\Omega'_1$ or $\Omega'_2$ shown in Figure 2.2. Define

$$\Theta^m_{i,j,k} = \begin{cases} \int_{\Omega^m_{i,j,k}} F(x,y)dx\,dy, & \tau_m = 2, \\ 0, & \text{otherwise} \end{cases}$$ (2.38)

for $m = 1, 2$. From (2.35) and Figure 2.2, we see that to take into account the curved boundary of the area $\Omega_{i,j,k}$, we need to use the following strategy. Let $Q^m_{i,j,k} = Z_{i,j,k} \left( \frac{x^m + x_{i,j,k}^m}{2}, \frac{y^m + y_{i,j,k}^m}{2} \right)$, where $Z_{i,j,k}$ is defined in (2.10). If $Q^m_{i,j,k} \in [z_k, z_{k+1}]$, then the point $\left( \frac{x^m + x_{i,j,k}^m}{2}, \frac{y^m + y_{i,j,k}^m}{2} \right) \in \Omega_{i,j,k}$. In this case $\Omega_{i,j,k}$ should be approximated by the union of $\Omega^m_{i,j,k}$; namely, we should set $t^m_{i,j,k} = 1$. Otherwise, if $Q^m_{i,j,k} \notin [z_k, z_{k+1}]$, then we set $t^m_{i,j,k} = -1$. Therefore we have

$$t^m_{i,j,k} = \begin{cases} 1, & Q^m_{i,j,k} \in [z_k, z_{k+1}], \\ -1, & \text{otherwise,} \end{cases}$$ (2.40)

We can approximate $\Theta^m_{i,j,k}$ by numerical quadratures as follows:

$$\Theta^m_{i,j,k} = \begin{cases} D^m \sum_{k'=1}^{K} w_{k'} \hat{D}^m_{k'} \sum_{k''=1}^{K} w_{k''} \hat{F} \left( \frac{x^m_{k',k''}, y^m_{k',k''}}{2}, \frac{y^m_{k',k''}}{2} \right), & \tau_m = 2, \\ 0, & \text{otherwise,} \end{cases}$$ (2.41)

where

$$D^m = \sqrt{ \left( x^m_2 - x^m_1 \right)^2 + \left( y^m_2 - y^m_1 \right)^2 },$$ (2.42)

$$\hat{D}^m_{k'} = \sqrt{ \left( x^m_{k'} + n_{k'} (x^m_2 - x^m_1) - \bar{x}^m(n_{k'}) \right)^2 + \left( y^m_{k'} + n_{k'} (y^m_2 - y^m_1) - \bar{y}^m(n_{k'}) \right)^2 },$$ (2.43)

$$\bar{x}^m_{k',k''} = \bar{x}^m(n_{k'}) + n_{k''} (x^m_1 - x^m_{k'}) - \bar{x}^m(n_{k'}),$$ (2.44)

$$\bar{y}^m_{k',k''} = \bar{y}^m(n_{k'}) + n_{k''} (y^m_1 - y^m_{k'}) - \bar{y}^m(n_{k'}).$$ (2.45)

Therefore the discrete analogue of (2.39) is given by

$$\hat{I}_{i,j,k} = \tilde{I}_{i,j,k} + \sum_{m=1}^{2} t^m_{i,j,k} \Theta^m_{i,j,k},$$ (2.46)

where $\tilde{I}_{i,j,k}, t^m_{i,j,k}$, and $\Theta^m_{i,j,k}$ are defined in (2.32), (2.40), and (2.41), respectively. $\hat{I}_{i,j,k}$ given by (2.46) should be the high order approximation to $I_{i,j,k}$ for most cells. However, we remind readers that $\hat{I}_{i,j,k}$ can still be not accurate enough to approximate $I_{i,j,k}$ for certain mesh cells. For example, when the curve segment $(\bar{x}^m(s), \bar{y}^m(s)), s \in$
\( [0,1] \), considered in (2.37) has intersection with the sides of \([x_i, x_{i+1}] \times [y_j, y_{j+1}] \) for \( s \in (0, 1) \), then the area \( \Omega_{i,j,k}^m \) defined in (2.37) is not totally contained in \([x_i, x_{i+1}] \times [y_j, y_{j+1}] \). Namely, in the treatment of the curved boundary of \( \Omega_{i,j,k} \), we may take into account a certain small area outside \([x_i, x_{i+1}] \times [y_j, y_{j+1}] \). Therefore in this case, \( \hat{I}_{i,j,k} \) given by (2.46) cannot be accurate enough to approximate \( I_{i,j,k} \) if high order accuracy is required. As mentioned in the introduction, our strategy to treat such cases is to expect that the delta function integral is approximated in a consistent way so that if the approximation in a mesh cell is not accurate enough, the approximation in its neighboring cells also is not, and moreover, the approximation in the union of the cells can achieve high order accuracy. The analysis of fulfillment of this strategy for our approximate delta function integral formula (2.46), however, could be complicated and will be left to future study. By numerical experiments in section 3 we show that our numerical methods designed by using formula (2.46) achieve the expected zero point inside the edge segment. However, if the interpolation polynomial of \( x_i \) is nonmonotone and the Newton iteration does not converge to the expected zero point, this only happens when the zero level surface \( \Gamma \) is nearly parallel to \((x, y)\)-plane near the cell \( C_{i,j,k} \). Nevertheless, in this case it is not detrimental if the Newton iteration for solving \( \hat{x}_m(s), \hat{y}_m(s) \) converges to a wrong zero point. In fact in
this case the area $\Omega_{i,j,k}$ defined in (2.37) for the cell $C_{i,j,k}$ and its one neighboring cell is the same while the sign $t_{i,j,k}^m$ for the two cells are different, and therefore, the summation of the computed delta function integrals (2.46) for the two cells can be high order accurate even if the curve segment point $\hat{x}^m(s), \hat{y}^m(s)$ is incorrectly computed. This embodies well our strategy of achieving high order accuracy for the method via error cancelation between neighboring cells. Therefore it is allowable that the Newton iteration converges to a wrong zero point for this case. However, we still need to avoid the nonconvergence of the Newton iteration. This can happen when the interpolation polynomial does not have a zero point due to interpolation errors. Our strategy in this paper to treat this case is that when the Newton iteration is performed more than 20 times, implying nonconvergence, we just quit the Newton iteration and set the computed $\hat{x}^m(s), \hat{y}^m(s)$ corresponding to the root of the first order interpolation polynomial. Our numerical experiments show that such nonconvergence of the Newton iteration in computing the curve segment points is generally not encountered for treating smooth level set functions. We observe such nonconvergence phenomenon in the computations of Example 3.1 in which the level set function is nonsmooth. In such cases our above strategy takes effect.

After the computation of the intersection points between the zero level set $\Gamma$ and the edge or side of the cell $C_{i,j,k}$, we need to compute the values of $\tilde{F}$ at a number of points inside or close to $[x_i, x_{i+1}] \times [y_j, y_{j+1}]$ in order to use the formulas (2.32) and (2.41). According to the definition (2.12), these correspond to approximating the one dimensional delta function integrals. High order numerical methods to approximate one dimensional delta function integrals have been studied in [43, 44]. In [45] for designing numerical methods to two dimensional delta function integrals, we used the method in [43] for computing one dimensional delta function integrals. The method in [43] can also be used for computing the values of $\tilde{F}$ in this paper. However, in this paper we use the natural strategy of the Newton iteration and interpolation to compute the one dimensional delta function integrals. The definition (2.12) gives

$$\tilde{F}(x, y) = \frac{\alpha(x, y, Z_{i,j,k}(x, y))}{|u_z(x, y, Z_{i,j,k}(x, y))|}.$$  \hspace{1cm} (2.47)$$

We can use the Newton iteration to compute $Z_{i,j,k}(x, y)$ and then use the interpolation to approximate (2.47). The advantage of the method in [43] is the avoidance of the Newton iteration, and thus it is convenient to use, especially in high dimensions. For computing one dimensional delta function integrals, the computational complexity is not a crucial issue. We adopt the method with the Newton iteration in this paper for computing the values of $\tilde{F}$ since numerical tests indicate that this method gives more accurate results for computing the three dimensional delta function integrals (2.1).

From above discussions we see that our method involves polynomial interpolation and the Newton iteration. Moreover our method also uses difference approximation. For example, in computing $\alpha(x, y, z)$ defined in (2.7) or $\tilde{F}(x, y)$ via the formula (2.47), we need to compute the derivative of $u$. However, we only have the grid point values of $u$. Therefore difference approximation formulas need to be used in these computations. We denote $R, E, T$ to be the order of interpolation polynomial, the tolerance of the Newton iteration, and the order of the difference approximation formula, respectively, in our method.

We give a comment on the bandwidth for our method. In level set methods concerning numerical efficiency, it is usually desirable to implement local version of the method which only computes the level set function values in the neighbor-
hood of the zero level set [1, 29, 3, 27, 24]. This is also the case for computing delta function integrals. In our method the level set functions need to be defined only in the neighborhood of the zero level surface. The bandwidth of the neighborhood domain depends on the parameters $R$ and $T$ in our method which are the order of interpolation polynomial and the order of the difference approximation formula, respectively. Assume we use the fourth order symmetric difference approximation formula presented in [45] as adopted in the numerical examples in this paper. Then we need the level set function to be defined two grid points away from the central point of the difference approximation formula. Applying $R$th order polynomial interpolation requires the level set function to be defined finitely at most $R + 1$ grid points in one direction. Considering that our method only treats the nontrivial cases in which the mesh cell intersects with the zero level surface, in the above context our method can be implemented when the level set functions are defined in mesh cells whose distance to a mesh cell intersecting with the zero level surface is no more than $D_b \equiv \lceil \frac{R - 1}{2} \rceil + 2$ mesh cells, where $\lceil x \rceil^+$ denotes the integer no less than $x$. As shown in the numerical examples in section 3, the numerical accuracy of our method is usually $R + 1$th order. Therefore our second to fourth order methods adopting the symmetric fourth order difference approximation formula correspond to $D_b = 2, 3, 3$, respectively. We also notice that the bandwidth $D_b$ for our method may be further reduced by adopting nonsymmetric difference approximation formula. We will not further discuss this in this paper.

2.3. Algorithm description. With the discussions of strategy and numerical implementation of our method in the above subsection, we now present the algorithm of our method. The algorithm is given by the following.

Algorithm I.
For each cell $C_{i,j,k}$ compute $\hat{I}_{i,j,k}$ which is the approximation to $I_{i,j,k}$.
Sum up $\hat{I}_{i,j,k}$ for all mesh cells to give the numerical approximation to the delta function integral (2.1).

In the above algorithm the key algorithm to compute $\hat{I}_{i,j,k}$ for each cell is given as follows.

Algorithm II: Give $\hat{I}_{i,j,k}$.
If the eight vertex values of $u$ in the cell $C_{i,j,k}$ have the same sign, set $\hat{I}_{i,j,k} = 0$.
Otherwise, compare the quantities $\tilde{u}^x_{i,j,k}$, $\tilde{u}^y_{i,j,k}$, and $\tilde{u}^z_{i,j,k}$ defined in (2.15)–(2.17).
If $\tilde{u}^z_{i,j,k}$ has the maximum value
$\hat{I}_{i,j,k}$ is computed according to the formula (2.46), with $\tilde{\eta}_m, \tilde{\eta}_M$.
If $\tilde{u}^y_{i,j,k}$ or $\tilde{u}^x_{i,j,k}$ has the maximum value,
$\hat{I}_{i,j,k}$ can be computed in similar principle based on the form (2.5) or (2.6).

Algorithm II includes the following parameters: $R$, $E_T$, $S$, and $T$, where $R$, $E_T$, and $T$ are the order of interpolation polynomial, the tolerance in the Newton iteration, and the order of difference approximation, respectively, and $S$ is the order of the quadrature rule used in the formulas (2.32) and (2.41).

3. Numerical examples. In this section we present numerical examples to demonstrate the accuracy of our methods. In the following examples we choose $S = 4$
in Algorithm II by using the Simpson rule, $T = 4$ in Algorithm II by using the fourth order difference approximation formula as adopted in [45], and set $E_T = 10^{-12}$. We test different values of the parameter $R$ as follows: method A: $R = 1$, method B: $R = 2$, and method C: $R = 3$.

From the two dimensional numerical examples and analysis in [45], it is known that parameter $R$ can be, generally, one order lower than the expected accuracy of the method in the two dimensional case. This leads us to expect methods A, B, and C are second to fourth order accurate for general computations in this paper. This will be demonstrated by numerical results. In the first example the level set function has low regularity, and numerical methods cannot achieve higher than second order accuracy. In other examples the level set functions are smooth. We show that methods A, B, and C achieve or are better than the expected second to fourth order accuracy in these examples. We will also demonstrate the stability of shifting mesh for our methods in Example 3.3 and the advantage of using high order numerical methods for computing (2.1) in Example 3.4.

Example 3.1. This is an example tested in [8, 38]. Let $u(x, y, z)$ be the signed distance function to the capsule-shaped surface appearing in Figure 12 of [8] and $f(x, y, z) = \nabla u \cdot \nabla u$, where $F(x, y, z) = (y^2, z^2, x^2)$. The exact value of (2.1) is zero. For a given mesh size we randomly shift the uniform mesh in each coordinate direction for 20 times. Table 3.1 presents the averaged absolute errors of the three methods over the 20 trials. The last column in the table presents the estimated convergence rates from numerical errors. The level set function in this example has low regularity, and standard numerical methods cannot achieve higher than second order accuracy. Table 3.1 shows that the three methods achieve or are slightly better than second order accuracy. Observing that methods B and C are much more accurate than method A for this example, we see that our high order method can give reasonable results even for the level set function with low regularity.

<table>
<thead>
<tr>
<th>Mesh size (m)</th>
<th>Method A</th>
<th>Method B</th>
<th>Method C</th>
<th>$R_e$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>3.42E-2</td>
<td>3.34E-3</td>
<td>1.20E-2</td>
<td>2.00</td>
</tr>
<tr>
<td>0.05</td>
<td>9.02E-3</td>
<td>6.52E-4</td>
<td>5.60E-4</td>
<td>2.22</td>
</tr>
<tr>
<td>0.025</td>
<td>2.28E-3</td>
<td>7.25E-5</td>
<td>4.70E-5</td>
<td>2.56</td>
</tr>
<tr>
<td>0.0125</td>
<td>5.26E-4</td>
<td>3.54E-5</td>
<td>3.75E-5</td>
<td></td>
</tr>
<tr>
<td>0.00625</td>
<td>1.39E-4</td>
<td>6.57E-6</td>
<td>6.44E-6</td>
<td></td>
</tr>
</tbody>
</table>

Example 3.2. This is an example tested in [33, 38]. Let $u(x, y, z) = x^2 + y^2 + z^2 - 1, f(x, y, z) = 4 - 3x^2 + 2y^2 - z^2$. The exact value of (2.1) is $\frac{40\pi}{3}$. Table 3.2 presents the averaged relative errors of the three methods over 20 trials in which the uniform computational mesh is randomly shifted in each coordinate direction. One sees that methods A and C achieve the second and fourth order accuracy, while method B is better than third order accuracy. We also test the case when the level set function is the nonpolynomial function. Table 3.3 presents the same type of errors of the three methods when choosing the level set function $u(x, y, z) = e^{x^2 + y^2 + z^2 - 1} - 1$ and the same weight function. Similarly, methods A and C attain the expected accuracy. It is interesting that method B also achieves fourth order accuracy in this test.

Example 3.3. This is an example tested in [33, 25, 26]. Let $u(x, y, z) = \frac{x^2}{(y^{120})^2} + \frac{y^2}{(0.5^2)} + \frac{z^2}{(0.5^2)} - 1, f(x, y, z) = 1$. The exact value of (2.1) is $\approx 9.90182152049617$. The last column in the table presents the estimated convergence rates from numerical errors.
Table 3.2
Example 3.2, averaged relative errors of methods A, B, and C; \( u(x, y, z) = x^2 + y^2 + z^2 - 1 \).

<table>
<thead>
<tr>
<th>Mesh size</th>
<th>0.1</th>
<th>0.05</th>
<th>0.025</th>
<th>0.0125</th>
<th>0.00625</th>
<th>( R_e )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method A</td>
<td>1.90E-3</td>
<td>4.71E-4</td>
<td>1.20E-4</td>
<td>2.98E-5</td>
<td>7.43E-6</td>
<td>2</td>
</tr>
<tr>
<td>Method B</td>
<td>6.48E-6</td>
<td>2.86E-7</td>
<td>1.04E-8</td>
<td>2.97E-9</td>
<td>5.26E-10</td>
<td>3.38</td>
</tr>
<tr>
<td>Method C</td>
<td>1.18E-5</td>
<td>7.37E-7</td>
<td>4.61E-8</td>
<td>2.87E-9</td>
<td>1.73E-10</td>
<td>4.01</td>
</tr>
</tbody>
</table>

Table 3.3
Example 3.2, averaged relative errors of methods A, B, and C; \( u(x, y, z) = e^{x^2+y^2+z^2} - 1 \).

<table>
<thead>
<tr>
<th>Mesh size</th>
<th>0.1</th>
<th>0.05</th>
<th>0.025</th>
<th>0.0125</th>
<th>0.00625</th>
<th>( R_e )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method A</td>
<td>3.98E-3</td>
<td>1.01E-3</td>
<td>2.54E-4</td>
<td>6.35E-5</td>
<td>1.59E-5</td>
<td>1.99</td>
</tr>
<tr>
<td>Method B</td>
<td>2.00E-4</td>
<td>1.18E-5</td>
<td>8.66E-7</td>
<td>4.21E-8</td>
<td>1.57E-9</td>
<td>4.2</td>
</tr>
<tr>
<td>Method C</td>
<td>2.23E-4</td>
<td>1.36E-5</td>
<td>8.40E-7</td>
<td>5.27E-8</td>
<td>3.30E-9</td>
<td>4.01</td>
</tr>
</tbody>
</table>

Table 3.4 presents the averaged relative errors of the three methods over 20 trials in which the uniform computational mesh is randomly shifted in each coordinate direction and rotated. As in Example 3.2, methods A and C achieve the expected accuracy, while method B also has fourth order accuracy and is even slightly better than method C. We then choose the level set function \( u(x, y, z) = e^{x^2/(1.5)^2+y^2/(0.75)^2+z^2/(0.5)^2} - 1 \) and the same weight function, and we present the same type of errors of the three methods in Table 3.5. We observe that the three methods attain similar accuracy. However, method B is not generally better than third order accuracy for smooth level set functions which will be shown in the next example.

Table 3.4
Example 3.3, averaged relative errors of methods A, B, and C; \( u(x, y, z) = \frac{x^2}{(1.5)^2} + \frac{y^2}{(0.75)^2} + \frac{z^2}{(0.5)^2} - 1 \).

<table>
<thead>
<tr>
<th>Mesh size</th>
<th>0.1</th>
<th>0.05</th>
<th>0.025</th>
<th>0.0125</th>
<th>0.00625</th>
<th>( R_e )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method A</td>
<td>1.32E-3</td>
<td>2.24E-4</td>
<td>5.46E-5</td>
<td>1.62E-5</td>
<td>4.25E-6</td>
<td>2.03</td>
</tr>
<tr>
<td>Method B</td>
<td>1.04E-4</td>
<td>3.56E-6</td>
<td>1.99E-7</td>
<td>4.38E-9</td>
<td>5.44E-10</td>
<td>4.48</td>
</tr>
<tr>
<td>Method C</td>
<td>8.67E-5</td>
<td>1.48E-5</td>
<td>4.73E-7</td>
<td>1.60E-8</td>
<td>1.09E-9</td>
<td>4.24</td>
</tr>
</tbody>
</table>

Table 3.5
Example 3.3, averaged relative errors of methods A, B, and C; \( u(x, y, z) = e^{x^2/(1.5)^2+y^2/(0.75)^2+z^2/(0.5)^2} - 1 \).

<table>
<thead>
<tr>
<th>Mesh size</th>
<th>0.1</th>
<th>0.05</th>
<th>0.025</th>
<th>0.0125</th>
<th>0.00625</th>
<th>( R_e )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method A</td>
<td>1.60E-3</td>
<td>5.50E-4</td>
<td>1.33E-4</td>
<td>4.44E-5</td>
<td>8.01E-6</td>
<td>1.89</td>
</tr>
<tr>
<td>Method B</td>
<td>5.28E-4</td>
<td>2.47E-5</td>
<td>1.67E-6</td>
<td>8.58E-8</td>
<td>6.78E-9</td>
<td>4.07</td>
</tr>
<tr>
<td>Method C</td>
<td>4.74E-4</td>
<td>2.67E-5</td>
<td>1.82E-6</td>
<td>1.15E-7</td>
<td>7.60E-9</td>
<td>3.97</td>
</tr>
</tbody>
</table>

We also test the stability of shifting mesh for our methods as performed in [25]. We again choose \( u(x, y, z) = \frac{x^2}{(1.5)^2} + \frac{y^2}{(0.75)^2} + \frac{z^2}{(0.5)^2} - 1, f(x, y, z) = 1 \). Table 3.6 presents the averaged, maximum, and minimum relative errors over 20 trials in which the uniform computational mesh is randomly shifted in each coordinate direction for the three methods. The stability of shifting mesh for our methods are shown in this table, indicating the potential advantage of our methods in applying to numerical
solutions to PDEs in which the stability of shifting mesh is important to ensure well-behaved results with mesh refinement.

Example 3.4. Let \( u(x, y, z) = x^2 + y^2 + \pi z^3 \). The exact value of (2.1) is \( \approx 1.83053049903717 \). Table 3.7 presents the averaged relative errors of the three methods over 20 trials in which the uniform computational mesh is randomly shifted in each coordinate direction and rotated. In this test we see that methods A, B, and C attain second, third, and fourth order accuracy, respectively. Therefore method B is essentially third order accurate, but it can attain higher than third order accuracy in certain cases as in the previous two examples. We then test the level set function \( u(x, y, z) = e^{x^2/(1.5)^2+y^2/(0.75)^2+z^2/(0.5)^2} - 1 \) and the same weight function. Table 3.8 presents the same type of errors of the three methods. Similarly, we observe that the three methods attain the expected second to fourth order accuracy.
Example 3.4. averaged relative errors of methods A, B, and C; 
\[ u(x, y, z) = e^{x^2/(1.5)^2 + y^2/(0.75)^2 + z^2/(0.5)^2 - 1} - 1. \]

### Table 3.8

<table>
<thead>
<tr>
<th>Mesh size</th>
<th>0.1</th>
<th>0.05</th>
<th>0.025</th>
<th>0.0125</th>
<th>0.00625</th>
<th>( R_e )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method A</td>
<td>1.27E-2</td>
<td>2.93E-3</td>
<td>6.14E-4</td>
<td>1.68E-4</td>
<td>3.83E-5</td>
<td>2.09</td>
</tr>
<tr>
<td>Method B</td>
<td>2.96E-2</td>
<td>3.61E-3</td>
<td>4.57E-4</td>
<td>5.36E-5</td>
<td>6.96E-6</td>
<td>3.02</td>
</tr>
<tr>
<td>Method C</td>
<td>7.96E-3</td>
<td>1.13E-4</td>
<td>8.71E-6</td>
<td>4.30E-7</td>
<td>2.63E-8</td>
<td>4.45</td>
</tr>
</tbody>
</table>

Figure 3.1. Relation between \( \log_{10}(|E|) \) and the relative time for different methods; 
\[ u(x, y, z) = e^{x^2/(1.5)^2 + y^2/(0.75)^2 + z^2/(0.5)^2 - 1} - 1. \]

order accurate. We test the mesh sizes from 0.1 to 0.0125. We denote \( \log_{10}(|E|) \) to be the \( \log_{10} \) of the averaged relative error, and we define the relative time to be the actual computational time divided by the computational time used by method D on the coarsest mesh. Figure 3.1 shows the relation between \( \log_{10}(|E|) \) and the relative time for different methods. From the comparison we see that methods A, B, and C are more efficient than method D, and the fourth order method is much more efficient than the second and third order methods. This comparison shows the advantage in computational efficiency of using fourth order methods compared with lower order methods for computing the three dimensional delta function integrals (2.1).

4. Conclusion. In this paper we designed a class of high order numerical methods to delta function integrals appearing in level set methods in the three dimensional case. This work is the extension of high order numerical methods to the same type of delta function integrals in two dimensions [45]. The methods were constructed by approximating the mesh cell restrictions of the delta function integral. In each mesh cell the three dimensional delta function integral can be rewritten as a two dimensional ordinary integral with the smooth integrand being a one dimensional delta function integral. The two dimensional integral takes one of three forms according to the comparison of the components of the gradient of the level set function. The main complication in designing our high order numerical methods is that the high order
accuracy for approximating delta function integrals cannot be achieved in each mesh cell. Therefore the high order methods were designed in a consistent way to facilitate proper error cancelations in mesh cells where high order accuracy are not achieved. This issue is essentially related to the construction of integral area for the transformed two dimensional ordinary integral in each mesh cell. This was performed by connecting the projection of intersection points between the zero level set and the edge of the mesh cell using proper procedure. We also treated the curved boundary of the integral area by considering the intersection points between the zero level set and the side of the mesh cell. Consequently, the mesh cell restrictions of the three dimensional delta function integral were approximated by applying standard two dimensional high order numerical quadratures and high order numerical methods to one dimensional delta function integrals. We presented numerical examples in which second to fourth order methods were implemented and shown to achieve or exceed the expected accuracy. We also demonstrated by numerical tests the stability of shifting mesh for our methods, which indicates the potential advantage of our methods in application to numerical solutions to PDEs, and the advantage in computational complexity of using high order numerical methods for computing the three dimensional delta function integrals. The establishment of error analysis for our methods in this paper, however, is a complicated issue and will be considered in the future.

REFERENCES

