Solving Fluid/Rigid body Interaction Problem by a Discontinuous Galerkin Level Set Method

P. Hu, J. E. Flaherty, M. S. Shephard

Abstract

We consider problems involving high-speed moving rigid objects in a fluid flow. Instead of a more traditional deformable mesh approach, we describe a novel fixed mesh approach which use a level set function to implicitly track the fluid-solid interface, therefore no mesh motion or modification is required. The interface boundary conditions are also captured implicitly by combining a Ghost Fluid technique with the level set approach. This fixed mesh approach is a much more efficient than moving mesh methods since no mesh modification is necessary. It has a wide-spread applicability for problems with complex object motion and/or complex geometry. It is also dimension free and relatively simple to implement relative to moving mesh approaches. The discontinuous Galerkin method (DGM) is used to discretize the Euler equations associated with a compressible inviscid fluid.

1 Introduction

Many physical problems involve the interaction of two or more materials. Such multi-material interaction is wide spread in fluid mechanics, biomechanics, meteorology and many other fields. Therefore, the ability to numerically simulate the effects of two or more different, yet inter-related physical materials is important. Solving multi-materials interaction problems is difficult and several open issues remain. Although different approaches need to be applied according to the different combination of the multi-materials, they basically may be classified into two general categories.

With a moving mesh method, the surfaces of rigid objects are treated as boundaries and the mesh is moved to follow their motion. Formulations may be Lagrangian, or arbitrary Lagrangian Euleriian (ALE) [5], [6]. In contrast, a fixed mesh approach uses some technique to capture object surface locations, such as a volume of fluid (VOF) [2] [3], front tracking method [30] [31] [32] [33] or level set [23], [24], [25], [21]. Each approach has its advantages and disadvantages. With mesh moving, the object surfaces remain sharp and captured more precise. However, mesh motion is difficult and costly in three dimensions. Meshes tend to become distorted and introduce ill conditioning. Solutions must be transfered between meshes at different times and this may induce excessive diffusion. The fixed mesh approach avoids mesh motion and is more efficient and simpler to implement, but object surfaces are less sharp.

In this paper, problems involving motion of a high-speed rigid object in a compressible inviscid fluid are particularly interested and the thereafter discussion will be focused on this specific case. The fluid flowing around or within an object may contribute to its movement and/or spin due to flow-induced pressure. On the other hand, the

movement of the object will affect the fluid flow. Fluid-solid interaction is often a transient occurrence, where the structural motion is dynamics and varies continuously with time.

The basic and important characteristics of these problems include both the continuous change of the computational domain with respect to time and the strong discontinuities in the fluid because of the speed of the object.

In this paper, instead of using the conventional deformable mesh approach, where the interface between the fluid and embedded solid object is treated as a boundary and it is captured by changing the mesh, e.g. moving the mesh, remeshing, or mesh modification. A novel fixed mesh approach is provided. In this approach, an Eulerian mesh is created over the entire computational domain. The movement of the rigid object will cut some of the mesh entities and introduce new boundaries inside the fluid. We use the level set technique [18], [19] to track the interface between the rigid body and the compressible fluid, and adapt the Ghost Fluid Method (GFM) [12] to capture the correct boundary condition implicitly at the interface. Therefore, handling the boundaries is simplified and we have the advantage of doing a fluid calculation on the entire domain. By doing this, we can handle the contribution of the embedded moving boundaries in the fluid without mesh modification.

The paper is organized as the following. In Section 2, the nonlinear hyperbolic conservation laws, euler equations, are introduced and corresponding Discontinuous Galerkin Finite Element numerical formulations for our problems are provided. In Section 3, we will talk about how to track and handle the fluid-solid interface for the fixed mesh approach. In Section 4, several numerical examples and results are presented. At last, Conclusion in Section 5.

2 Numerical Formulation

We consider the two-dimensional motion of a rigid object in a compressible, inviscid fluid. Fluid motion is governed by the Euler equations.

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho v_1\\ \rho v_2\\ \rho v_3\\ E\end{pmatrix}+\nabla\cdot\begin{pmatrix}\rho\vec{v}\\ \rho v_1\vec{v}+p\vec{e}_1\\ \rho v_2\vec{v}+p\vec{e}_2\\ \rho v_3\vec{v}+p\vec{e}_3\\ (E+p)\vec{v}\end{pmatrix}=0. \tag{1}$$

For simplification, we denote

$$\mathbf{u}=\begin{pmatrix}u_1\\ u_2\\ u_3\\ u_4\\ u_5\end{pmatrix}=\begin{pmatrix}\rho\\ \rho v_1\\ \rho v_2\\ \rho v_3\\ E\end{pmatrix},\qquad \vec{\mathbf{F}}(u)=\begin{pmatrix}\vec{F}_1\\ \vec{F}_2\\ \vec{F}_3\\ \vec{F}_4\\ \vec{F}_5\end{pmatrix}=\begin{pmatrix}\rho\vec{v}\\ \rho v_1\vec{v}+p\vec{e}_1\\ \rho v_2\vec{v}+p\vec{e}_2\\ \rho v_3\vec{v}+p\vec{e}_3\\ (E+p)\vec{v}\end{pmatrix}.$$

Variables with a super-imposed arrow refer to physical vectors with two and three components, respectively in two and three dimensions. Bold type is used for vectors in an abstract *m*-dimensional space, and ∇ is the vector

valued divergence operator

$$\nabla = [\nabla, \nabla, ...\nabla]^T,$$

where $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})$. Here ρ is the fluid density, v_1, v_2 and v_3 are the x_1-, x_2- and x_3-components of the velocity $\vec{v}$, $\vec{e}_1$, $\vec{e}_2$ and $\vec{e}_3$ are unit vectors in x_1-, x_2- and x_3-direction, E is the total energy per unit volume, $p = p(\rho, e)$ is the pressure, and e is the internal energy per unit mass. The total energy is the sum of the internal energy and the kinetic energy, i.e.,

$$E = \rho e + \frac{1}{2}\rho(v_1^2 + v_2^2 + v_3^2). \tag{2}$$

An ideal gas also satisfies the equation of state

$$p = (\gamma - 1)\rho e, \qquad where \qquad \gamma > 1. \tag{3}$$

Consider a problem in a computational domain Ω with boundary $\partial\Omega$ that has been divided into a collection of N_h elements such that

$$\Omega = \bigcup_{j=1}^{N_h} \Omega_j.$$

With the DGM, a weak formulation of (1) is constructed on a single element Ω_j by multiplying (1) by a test function $\boldsymbol{\phi} \in (L^2(\Omega_j)^m$, integrating the result on Ω_j, and using the divergence theorem to obtain

$$\partial_t(\mathbf{u}, \boldsymbol{\phi})_{\Omega_j} - (\vec{\mathbf{F}}(\mathbf{u}), \nabla\, \boldsymbol{\phi})_{\Omega_j} + \langle \mathbf{F}^n, \boldsymbol{\phi}\rangle_{\partial\Omega_j} = 0, \qquad \forall \boldsymbol{\phi} \in (L^2(\Omega_j))^m. \tag{4}$$

The L_2 volume and surface inner products on Ω_j are defined as

$$(\mathbf{u}, \boldsymbol{\phi})_{\Omega_j} = \int_{\Omega_j} \boldsymbol{\phi}^T \mathbf{u} ds, \qquad \langle \mathbf{u}, \boldsymbol{\phi}\rangle_{\partial\Omega_j} = \int_{\partial\Omega_j} \boldsymbol{\phi}^T \mathbf{u} d\tau.$$

$\mathbf{F}^n$ is the normal flux which is defined as $\mathbf{F}^n = \vec{\mathbf{F}}(\mathbf{u}) \cdot \vec{n}$, where $\vec{n}$ is the unit outer normal vectorto $\partial\Omega_j$. However, with discontinuous basis, $\mathbf{F}_n$ is not defined on $\partial\Omega_j$. In this situation, we use a numerical flux $\mathbf{F}^n(\mathbf{u}_j, \mathbf{u}_k)$ that depends on the solution $\mathbf{u}_j$ on Ω_j and $\mathbf{u}_k$ on the neighboring element Ω_k sharing the portion of the common boundary $\partial\Omega_{jk}$. The numerical flux must be consistent, so $\mathbf{F}_n(\mathbf{u}, \mathbf{u}) = \vec{\mathbf{F}}(\mathbf{u}) \cdot \vec{n}$. With such a numerical flux, the DG formulation can be rewritten as

$$\partial_t(\mathbf{u}, \boldsymbol{\phi})_{\Omega_j} - (\vec{\mathbf{F}}(\mathbf{u}), \nabla\boldsymbol{\phi})_{\Omega_j} + \sum_{k=1}^{n_{\Omega_j}} \langle \mathbf{F}^n(\mathbf{u}_{\Omega_j}, \mathbf{u}_{\Omega_k}), \boldsymbol{\phi}\rangle_{\partial\Omega_j} = 0 \tag{5}$$

where n_{Ω_j} is the number of faces of element Ω_j. Only the normal traces have to be defined on $\partial\Omega_j$ and several options are possible (see, e.g., [14] [15]). It is usual to define the trace as the solution of a Riemann problem across $\partial\Omega_j$. an exact Riemann solver is used to compute the numerical fluxes and a Barth limiter [16] is used to restrict spurious oscillations when polynomial degrees $p > 0$ are used. The Barth limiter is applied on each element after a new solution field is computed, so that the minimal and maximal value of the solution conservative variables in one element are bounded by their mean values on neighboring elements.

3 Treating Interface

For our fixed mesh approach, the object's surface is implicitly captured by the zero level of a level set function. Therefore, special treament is needed to define boundary conditions that simulate the fluid-rigid object interaction. We will talk about the level set method by which the interface is captured in Section 3.1, the Ghost Fluid idea to implicitly capture the boundary conditions in Section 3.2. In Section 3.3, the detail of boundary treatment will be discussed. in Section 3.4, we will talk about the force calculation on the rigid object induced by the fluid flow.

3.1 Level Set Method

The level set method was introduced by Osher and Sethian [17] for dynamic implicit surfaces. The basic idea of the level set method is implicit capturing of an interface by using a smooth time-dependent function, the level-set function, which represents the interface as the roots of this function. The interface motion satisfies a time-dependent Eulerian initial value partial differential equation, and, therefore, the position of the interface with respect to time can be updated by solving the appropriate differential equation.

In our problems, The level-set function $\phi(x,t)$ is defined as a signed distance function in the whole computational domain at any time t. If we denote the boundary of the rigid body, i.e., the interface between the fluid and the rigid body as B, the inside part of the rigid body as I, and the fluid as O, the level set function $\phi(x,t)$ will satisfy the following conditions at any time t

$$\begin{array}{ll} \phi(x,t) = 0, & \text{for all } (x) \in B, \\ \phi(x,t) < 0, & \text{for all } (x) \in I, \\ \phi(x,t) > 0, & \text{for all } (x) \in O. \end{array} \tag{6}$$

Thus, the interface is captured for all time implicitly by locating the set for which function ϕ vanishes. This property is significant when the location of the interface is hard to express explicitly.

The governing equation of the level set advection is

$$\frac{\partial \phi}{\partial t} + \vec{U} \cdot \nabla \phi = 0, \tag{7}$$

where $\vec{U}$ is the velocity of the level set advection, which is equal to the velocity of the interface propagation.

With the normalization $|\nabla \phi(x) = 1|$, the normal to a level surface satisfies

$$\vec{n}(x,t) = \nabla \phi(x,t). \tag{8}$$

We can calculate $\vec{n}$ at a point which need not necessarily lie on the internal boundary because the level-set function is defined on the whole computational domain and the normal function is constant along a trajectory perpendicular to any $\phi = constant$ surface. This is a great advantage for our calculations. It is also worth mentioning that we do not have to define the level set function on the whole domain. Level set functions need only be defined within a small region of the interface of the solid object and the fluid. This is important when it is hard to define a signed

distance function on the whole complex domain for a complex object.

We assume that the level set function can be expressed as an explicit function. For some problems, this may not be true. A numerical approximation can be used in that case.

3.2 Ghost Fluid Method

Ghost Fluid Method (GFM) was first proposed by Fedkiw et al. [12] to avoid the possible oscillations at multi-material interfaces. It is well known that under Eulerian schemes, large spurious oscillations may occur in the pressure and velocity fields near the material interface with ordinary computational techniques. The main reason for these non-physical oscillations is a change of the equation of state across the material interface. In [12], authors introduced GFM for two-phase compressible flow in which ghost nodes are defined at every grid point in the computational domain so that at every time moment, each grid node contains two sets of conservative variables: mass, momentum, and energy (or entropy) for the current fluid at this grid point, and ghost mass, ghost momentum, and ghost energy (or entropy) for the other fluid. After the ghost values have been set for all grid points, standard numerical schemes can be applied for each fluid separately. Therefore, instead of solving a two-phase fluid flow problem, two separate one-phase fluid flow problems are solved at every time step. Then a valid solution set for every grid point is chosen by some criteria and the invalid one is discarded. By doing this, oscillations at multi-material interfaces are avoided without explicitly using interface jump conditions.

GFM not only gives a way to remove the non-physical oscillations near the interface of multi-materials, but also provides an innovative way to handle problems involving multiphase flows. The existence of the multi-materials interface actually introduce a boundary for each fluid, therefore proper boundary conditions should be set near the interface for each fluid. By using GFM to define a ghost artificial fluid and to properly set the ghost values for it, the boundary conditions can be induced automatically and implicitly. It will be looks like that no boundary exists at all for each fluid flow. This is a big advantage because of the difficulty of applying the correct boundary conditions on cut entities is avoided .

To apply GFM correctly, the information about interface of the multi-materials is required. The interface position is used to set the proper ghost values and to discard the invalid values for every grid node in the computational domain. Level set method has been used a lot and has been proved successfully for this purpose besides its own advantages to track moving interface. By using the level set method, the multi-materials interface is represented as the zero level of the level set function. The location of the interface is captured automatically by advance the level set function to the next time step, therefore the interface does not need to be tracked explicitly. And the sign of the level set function is used to distinguish the fluids. With the level set function to track the interface implicitly and GFM to capture the boundary condition implicitly, the whole computational scheme becomes simple and easy to implement. This holds true for even multi-dimension situations and for high-order time integration scheme, particularly Runge-Kutta methods. In the work of this paper, GFM is also combined with level set method to be benefit from those good properties.

When GFM was first introduced, it is for two-phase compressible flows problems with contact discontinuities at the interface. Since then, it has been applied to a lot of problems, including multi-component compressible and incompressible, viscous and inviscid flows as well as problems that couple fluid with deformable solid materials, etc. In this paper, we extend GFM to problems involving the interaction of inviscid compressible flow and a rigid

object.

3.3 Setting the Conservative Variables for the Fluid

Due to the introduction of the level set in our DG method, the position of the surface of the object in the fluid is represented by the zero level set. For different entities of the original Eulerian mesh, there are two different possible situations according to their position with respect to the zero level set of the level set function. Case (i), the entity is totally outside the zero level set. Case (ii), the entity is totally inside the zero level set or the entity is cut by the zero level set. Figure 1 shows both two cases, where I represents the fluid-object interface. There, entity $\Omega 1$ to $\Omega 7$ belong to case (ii) and $\Omega 8$ belongs to case (i). For all entities of case (i), the object in the fluid has no direct effect on this entity, so no special operations are needed. On the other hand, for all entities of case (ii), a new boundary was actually produced and the proper fluid variable should be set for the ghost part of the entity when using GFM to capture the correct boundary conditions; therefore, every entity of case (ii) is called *ghost entity*. The level set enables us to locate those entities that need special treatment, even when we do not know the position of the moving boundary in the entity. Actually, we also do not want to find it because it is complicated and expensive to do so, especially for high order approximation and for high dimension case.

To treat the interface conditions correctly, it is important to understand the dynamics of the fluid and rigid body motion. We can treat a rigid body moving in a fluid as a contact discontinuity because the fluid near the interface will move with the velocity of the rigid body. The Rankine-Hugoniot jump conditions imply that both the pressure and the normal velocity, $\overrightarrow{V}_N = \overrightarrow{V} \cdot \overrightarrow{N}$, are continuous across the interface. Therefore, we must set the values for the *ghost entity* to enforce these two conditions. There are three kinds of state variables that need to be set: velocity, density, energy (or pressure). Reference [12] discusses this for two-phase flow problems. We have similar rules. Continuity should be satisfied across the interface for density, the tangential component of velocity and pressure. At the same time, since the interface of the rigid body acts as a reflective wall boundary for the fluid, a reflective constraint could capture the boundary condition. In the original work of [12] [13], where GFM is designed and implemented for finite difference method on structured mesh grid. In that case, the approximated solution of the problem is represented at grid points, and ghost values are set at every ghost grid point. After the ghost values have been set, then finite difference scheme can be applied directly to advance PDEs to next time step. This is no longer true for our GFM under the Discontinuous Galerkin Finite Element Method(DGFEM). For DGFEM, the approximated solution is represented at every mesh entity. Therefore, instead of setting ghost values at every ghost grid point, ghost valued are set at every *ghost entity*. To set the ghost values for a *ghost entity*, every gauss point of the entity is first checked. If the level set function has negative value at a gauss point, namely the gauss point is inside the rigid object, we call this gauss point is a *ghost gauss point* of this *ghost entity*, and the ghost values at this point need to be set. Otherwise, nothing need to be done for this point. For examples, in Figure 1, all *ghost gauss points* are marked as $*$, all other gauss points are marked as $\cdot$. To set the ghost values for a *ghost gauss point* P_G inside the rigid body, we first find the reflective point P_R in the fluid with respect to the interface and compute ρ_R, E_R, and $\vec{v}_R$ at P_R. Then ρ_G, E_G, $\vec{v}_G$ at the ghost point P_G are setting as

$$\begin{aligned} \rho_G &= \rho_R, \\ \vec{v}_G &= \vec{v}_R^T - \vec{v}_R^N + 2\vec{U}_G^N, \\ E_G &= E_R - \tfrac{1}{2}\rho_R|\vec{v}_R|^2 + \tfrac{1}{2}\rho_R|\vec{v}_G|^2, \end{aligned} \tag{9}$$

where $\vec{v}_R^T$ and $\vec{v}_R^N$ are the velocity components of the fluid at P_R in the tangent and normal directions, respectively,

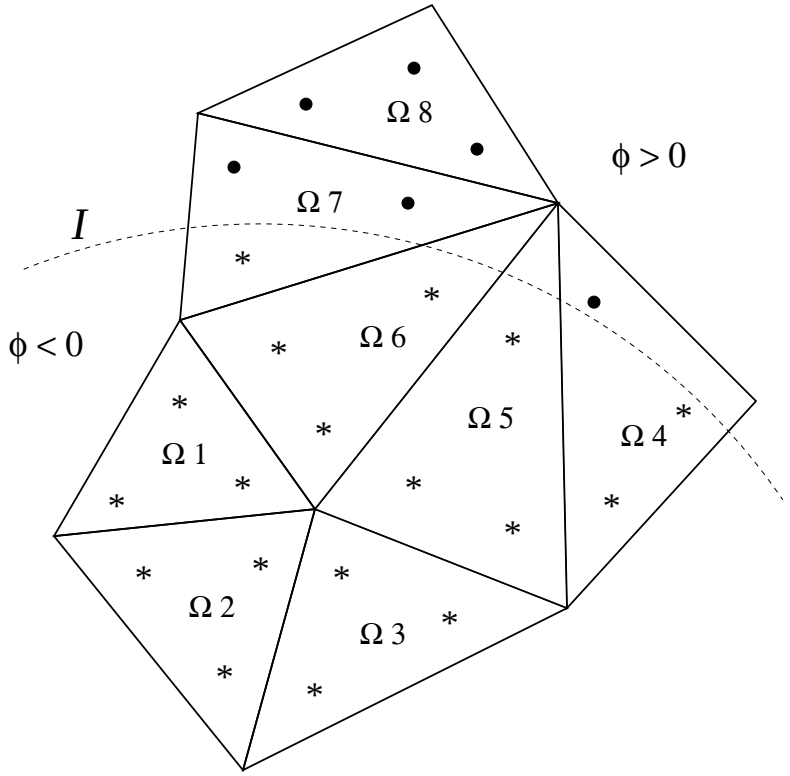

Figure 1: Entity and zero Level Set Relationship

and $\vec{U}_G^N$ is the normal component of the velocity of the moving object at P_G.

The equation of state is

$$E = \rho e + \frac{1}{2}\rho|\vec{V}|^2;$$

therefore, the constraint [p] = 0, where [ψ]: $L^2(\mathbb{R}) \rightarrow \mathbb{R}$ is defined as the jump in any function ψ at position x. This is satisfied implicitly at the interface by setting the energy of the ghost fluid as above. Also, [$\vec{V}_N$]=0 because the fluid near the rigid surface has a normal velocity component equal to $\vec{U}_G^N$. Moreover, our choice of ghost values gives a continuous density, energy and tangential velocity component at the interface.

After setting the ghost values for every *ghost gauss point* of a *ghost entity*, the solution on this entity is reconstructed by L_2 projection by using the new setting values. L_2 projection is a good corresponding choice to the way of setting the ghost values because it only uses the values at the gauss points of the entity. And the L_2 projection also keeps the conservation of mass, momentum, and energy.

In general, finding the reflective point is a problem, especially in the high-dimension case. However, since we keep the level set function $\phi(x)$ as a signed distance function, we can use its nice properties to overcome this difficulty. For a given point $\vec{x}_1$, the closest point on the interface can be approximated as

$$\vec{x}_2 = \vec{x}_1 - \phi(\vec{x}_1)\nabla\phi(\vec{x}_1).$$

Similarly, the reflective point $\vec{x}_3$ is

$$\vec{x}_3 = \vec{x}_1 - 2\phi(\vec{x}_1)\nabla\phi(\vec{x}_1).$$

This approximation is straightforward even in the multi-dimensional case. This is a great advantage because we can treat the one-, two- and three-dimensional cases with the same scheme and the numerical implementation is simple and consistent for all cases.

It is possible that a reflective point does not belong to the computational domain of the problem. Then the way to set ghost values at a *ghost point* will fail because of not being able to find the mesh entity that contains this reflective point, therefore can not get the conservative variable values at the point. For example, in Figure 2, the reflective point P_2 of a *ghost gauss point* P_1 falls outside of domain Ω. In this situation, the ghost values at P_2 can not be set directly. There are two possible ways to solve the difficulty. Before we talk about the details of how to do it, Let's give some notations.

Refer to Figure 2. Let L denote the line across point P_1 and P_2. Recall Ω is the problem domain, $\Gamma = \partial\Omega$ is the domain boundary. Since $P_1 \in \Omega$ and $P_2 \notin \Omega$, we know

$$P_{bdy} = L \cap \Gamma \neq \emptyset$$

Then we denote the close point to P_1 in P_{bdy} by P_3. For P_2, there exists a point P_4 which lies in the same part of boundary Γ as point P_3 and has the shortest distance to P_2 from this piece of boundary. The reflective point of P_2 to the boundary Γ is denoted by P_5.

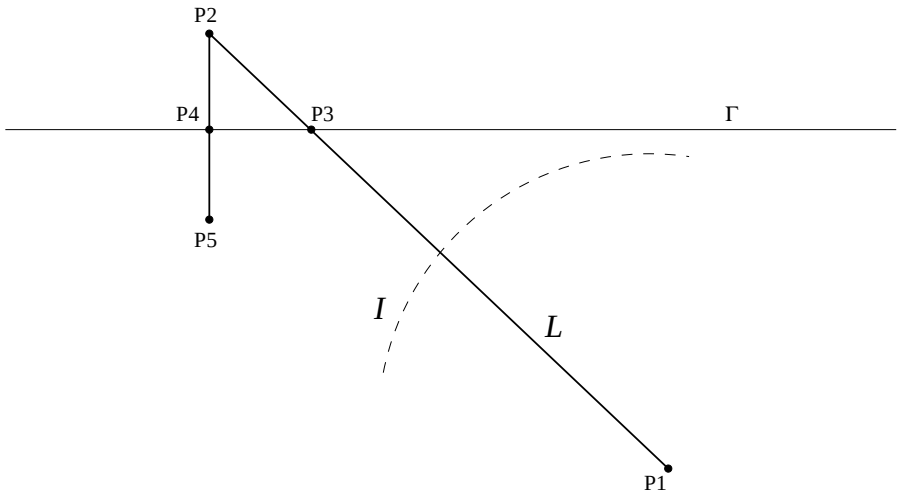

Figure 2: The situation for out of domain reflective point

Now we can describe the two possible ways to set the ghost values for P_1.One simple way is to just use the values at P_3, instead of those at P_2, to set the ghost values at P_1 without considering the bounary conditions. This is a relative easy way to realize. However, it is possible that relative large error will be introduced for some cases.

The Other way is to mark P_2 as a ghost point and try to set the ghost values at P_2. After that, the ghost values at P_1 are set by using the ghost values at P_2. Setting the ghost values for P_2 is depend on the boundary conditions at P_4. For example, if the boundary condition at P_4 is assumed to be

- Transmitting boundary condition. The value of P_4 is used to set ghost values at P_2 by just copying corresponding values.
- External boundary condition. If the external boundary condition is already know at P_2, then it can be applied at P_2 directly to get the ghost values at P_2, otherwise, use the boundary values at P_4 to set.
- Reflective boundary condition. Then point P_5 of P_2 respect to P_4 is used to set the ghost values at P_2 by applying reflective boundary principles. However, P_5 may falls out of the computational domain or falls into the rigid object part.

This is a more accurate way, however it is complicated and maybe time consuming for complex geometry.

To set the ghost values at a *ghost gauss point* P_G, the conservative variable values at the corresponding reflective point P_R are needed. This requires to find the mesh entity that contains P_R. For a general mesh database, it means a search to a linked list. Since this procedure is required for every *ghost gauss point* of every *ghost entity*. It will be very expensive for large mesh database and for high-order approximation. To improve the efficiency, a dynamic octree data structure is constructed and used to search the mesh entities. The octree is a dynamic one, therefore it will be suitable to our approach when combining with mesh adaptation procedure.

In the practical implementation, only those ghost entities within a small distance to the surface of the rigid body, i.e., the zero level set, need be considered. The bandwidth can be set to about 4-6 cells near the interface. This is consistent with defining the level set function only in a small bandwidth near the surface of the rigid body. By using this technique, the computation can be made efficient when the rigid body has a relatively large volume.

3.4 Force on the Rigid Object

When the movement of the rigid object is not prescribed, the flow has an effect on its movement. Therefore, the force of the flow on the object must be computed. We first assume that forces act on the centroid of the rigid object, therefore, spin of the rigid object is ignored. Then the force on the rigid object is

$$\int_I p \cdot \vec{n} ds, \tag{10}$$

where p is the pressure and I is the interface.

Calculating (10) is not obvious because of the lack of an explicit expression of the surface I, which is only implicitly represented as the zero level of level set function ϕ, However, since ϕ has definition on the whole computational domain, we can compute (10) indirectly by transfer the surface integral by a volume integral on the domain Ω as

$$\int_I p\vec{n} ds = \int_\Omega p\delta(\phi(x))\vec{n} dv, \tag{11}$$

where $\delta(x)$ is the Kronecker delta satisfying

$$\int_{-\infty}^{+\infty} \delta(x)g(x)dx = g(0) \tag{12}$$

for any continuous function $g(x)$.

Let $\hat{\delta}$ be a smooth approximation of $\delta(x)$ and consider

$$\int_I p\vec{n}ds \approx \int_\Omega p\hat{\delta}(\phi(x))\vec{n}dv. \tag{13}$$

Since Ω is the union of N_h elements

$$\Omega = \bigcup_{j=1}^{N_h} \Omega_j,$$

We have

$$\int_I p\vec{n}ds \approx \sum_{j=1}^{N_h} \int_{\Omega_j} p\hat{\delta}(\phi(x))\vec{n}dv. \tag{14}$$

One choice of $\hat{\delta}$ is

$$\hat{\delta}(x) = \begin{cases} \frac{1}{2\epsilon}(1 + cos\frac{\pi x}{\epsilon}), & \text{if } |x| < \epsilon \\ 0, & \text{if } |x| > \epsilon \end{cases}, \tag{15}$$

where ϵ is a small constant representing the bandwidth of the region around the interface.

With this choice of $\hat{\delta}$ and the level set function $\phi(x)$ as a signed distance function, the formula for the boundary force is

$$\int_I p\vec{n}ds \approx \sum_{j=1}^{N_h} \int_{\Omega_j} \frac{1}{2\epsilon}(1 + cos\frac{\pi\phi}{\epsilon})p\nabla\phi dv. \tag{16}$$

Two other choices of the approximation function $\hat{\delta}$ of function δ are

$$\hat{\delta}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \tag{17}$$

and

$$\hat{\delta}(x) = \begin{cases} d(\epsilon^2 - x^2)^m, & \text{if } |x| < \epsilon \\ 0, & \text{if } |x| > \epsilon \end{cases}, \tag{18}$$

where σ, ϵ and m are free parameters to be set according to the problem at hand, and d depends on m. The basic rule is that the approximation errors should be small compared to the errors in solving the PDEs. The Appendix gives the details of the error estimation and control.

4 Examples and Results

In the numerical tests that follow, several aspects of the two approaches are considered, including different motions of the rigid body, the flexibility of the methods, and efficient implementation of the methods.

4.1 Translation of Rigid Body

Consider a uniform Mach 3 flow in a wind tunnel of unit width and 4 units long. The tunnel is assumed to have an infinite width in the direction orthogonal to the plane of the computation. The left end is an inflow boundary and all gradients vanish at the right. Initially the wind tunnel is filled with an ideal gas with $\gamma = 1.4$, $\rho = 1.0$, $p = 1.0$, and (v_1, v_2, v_3) = ($3\sqrt{1.4}/2$, 0, 0). Gas with this density, pressure and velocity is continually fed from the left-hand boundary. At the same time, a solid ball with radius 0.125 units is moving from right to left with constant velocity $3\sqrt{1.4}/2$. Along the walls of the tunnel reflecting boundary conditions are applied. The density field and velocity field at t = 0.1, 0.25, 0.5 and 1.6 are given in Figure 3. All calculations used a CFL=0.3.

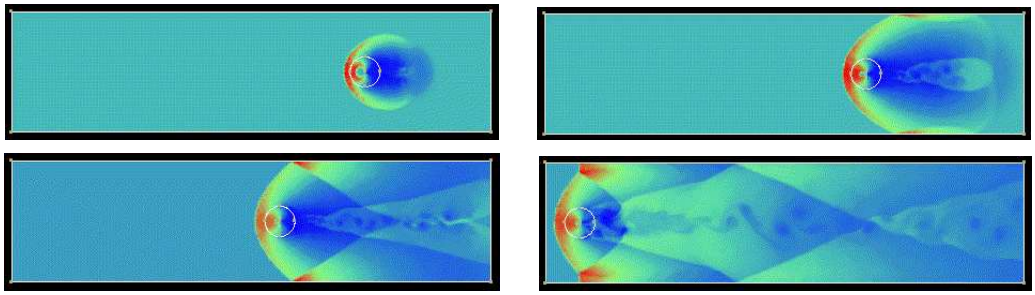

Figure 3: Density field at t = 0.1, 0.25, 0.5, and 1.6 from top left to bottom right.

The results are qualitatively good and compare well with the case of a fixed ball in a wind tunnel with the wind having a speed corresponding to the sum of the speed ball and the speed of the wind.

4.2 Rotation of a Rigid Body

A rectangle of 2 units wide and 1 unit high is filled with fluid. At all four boundaries, reflective wall boundary conditions are applied. Initially the rectangle is filled with an ideal gas with $\gamma = 1.4$, which everywhere has $\rho = 1.0$, $p = 1.0$, and (v_1, v_2, v_3) = (0, 0, 0). A rigid object centered at the origin rotates around its center in the fluid in the clockwise direction. The density field at t = 0.05, 0.1, 0.2 and 0.4 are shown in Figure 4.

The object rotates around its center with constant angular velocity. Since the absolute velocity at its ends is higher than the velocity of other parts, shocks develop near two ends. The top left and top right pictures show the density field of the fluid after the object rotates by an angle of $\pi/2$ and π respectively. The bottom left picture shows the shock is "broadcast" to the fluid and the bottom right picture shows the interaction of shocks after reflection from the walls.

4.3 A Example with Relative complex fluid Domain: Moving Square Object in a Tube

Consider a square object moving in a tube in the y direction with velocity $v_2 = 330.076096$ and Mach number 0.95. Initially, the fluid has $\rho = 1.17478$, $p = 101300$ and $(v_1, v_2, v_3) = (0, 0, 0)$. A transmission condition is assumed for the top and bottom edges and wall condition for other surfaces. The topology of the fluid domain

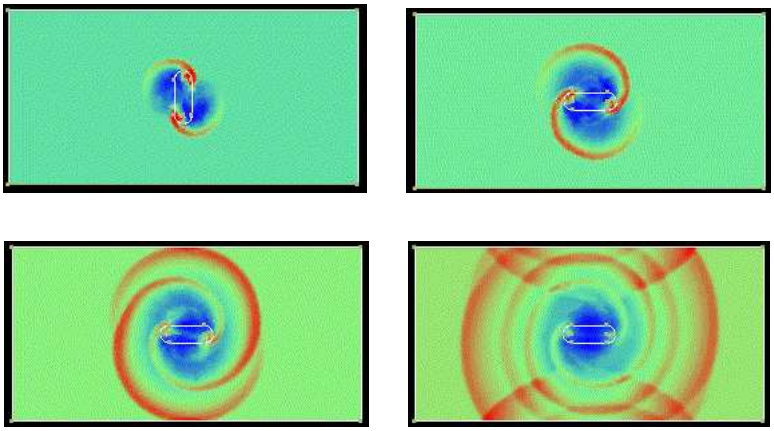

Figure 4: Density field at t = 0.05, 0.1, 0.2, and 0.4 from top left to bottom right.

changes during the simulation. This is very difficult for mesh moving. However, our fixed mesh approach works well.

The pressure field at time t = 0.25, 0.5, 0.7, 0.9 and 1.2 are shown in Figure 5.

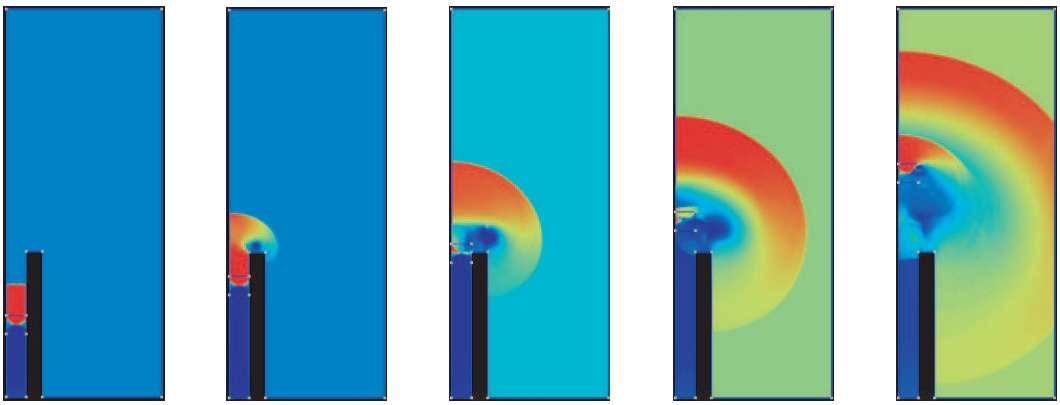

Figure 5: Pressure field at t = 0.25, 0.5, 0.7, 0.9 and 1.2 from left to right.

4.4 Adaptation Example

Combining DG and ghost fluid elements simplifies adaptivity. The rigid body has little influence on the adaptive procedure. As an example, consider the problem of section 5.1 with a coarser initial mesh using adaptive h-refinement. We show the pressure field and the corresponding mesh at t = 0.5 and 0.15 in the Figure 6. we see that adaptive h-refinement is refining the mesh near the shock and at the fluid-solid interface. Because of the ghost fluid method, adaptation is implemented inside of the rigid body. This is reasonable near the interface because of the symmetry property near the interface and it is necessary to capture the interface conditions correctly.

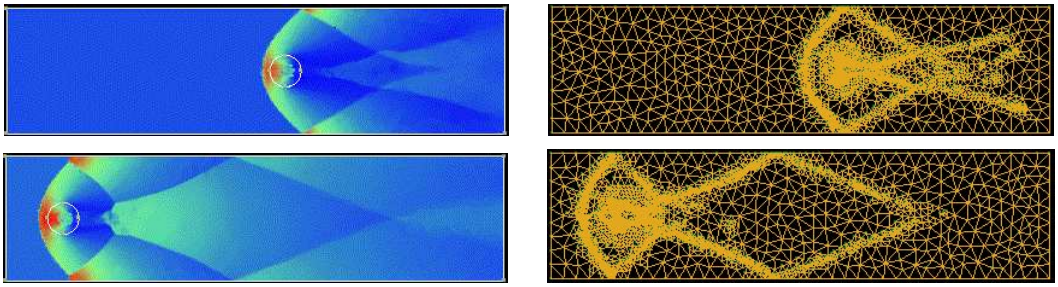

Figure 6: Pressure field (left) and corresponding mesh (right) at t = 0.5(top) and 1.5(bottom).

4.5 Comparison

We re-visit the example of the constant speed translation of rigid ball. To verify the correctness of our fixed mesh approach and compare it with conventional deformable mesh approaches, we solve the example by three ways. First way just uses the fixed mesh approach of this paper; the second way uses an ALE moving mesh idea similar to those work of Farhat [28] [29]; the third way avoids to solve the moving object problem, instead it tries to solve a reverse problem, in which the rigid ball is assumed fixed and the fluid moves with the relative sum speed of the fluid and the rigid ball of the original problem. The results are compared with each other with density and the density along the center line shown in Figure 7 at time t = 0.2.

We see that all three approaches give similar results. They capture the shock at almost the same position.

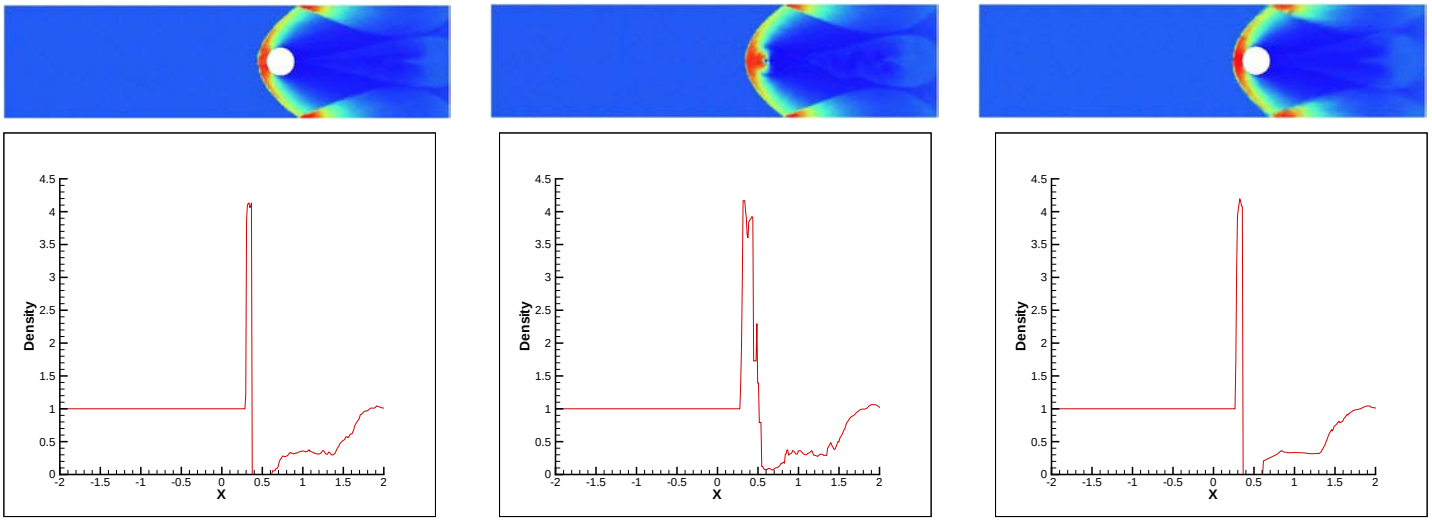

Figure 7: Density field at t = 0.37 (top) and Density field along the center straight line(bottom): reverse problem(left), fixed mesh approach(middle), deformable mesh approach(right).

h	Δm	$\frac{\Delta m}{m}$
0.04	1.23×10^{-2}	3.11×10^{-3}
0.02	4.67×10^{-3}	1.18×10^{-3}
0.01	2.32×10^{-3}	5.88×10^{-4}

Table 1: Translation of Rigid Ball at t = 0.37

4.6 Conservation

We conduct some experiments concerning the conservation of variables. Conservation of the mass should still hold if the corrected boundary condition is used at the interface. To simplify the process, we will apply the reflective wall conditions to all the computational boundaries for the experimental problems. Thus, neither inflow nor outflow exists in the test problems.

For the constant speed translation example, we present results for at $t = 0.5$ in Table 1.

In Table 1, h is the size of uniform mesh, m is the total mass in the domain at a given time. Δm is the absolute error of the approximated mass and $\frac{\Delta m}{m}$ is the relative error of the mass. The conservation of the mass is verified from both Tables because the error goes to zero when the mesh size goes to zero.

5 Conclusion

A novel approaches to solve fluid/rigid body interaction problems, fixed mesh approach, are considered from theoretical and numerical perspectives. Discontinuous Galerkin methods are used as the numerical method because of its advantages in several aspects and characteristics of our problems.

In this approach, we propose a new idea to solved fluid-rigid body interaction problems by a computationally simple approach. By this new method, no mesh modification or mesh moving are needed. Instead, the mesh in the computational domain can be fixed all the time. In our approach, level set technique and ghost fluid method are used to catch and handle the problems corresponding to the interface between the fluid and solid object. This method is efficient and it has wide-spread applicability to handle problems with complex movement or/and complex geometry shape of the moving object. It is also a dimension free method with great advantage to implement without difficulty in high dimension situation.

References

[1] A. SORIA AND F. CASADEI *Arbitrary Lagrangian-Eulerian Multicomponent Compressible Flow with Fluid-Structure Interaction*, International Journal for Numerical Methods in Fluids, 25, 1263-1284 (1997).

[2] W. J. RIDER AND D. B. KOTHE, *Reconstructiong Volume Tracking*, J. Comput. Phys. 141, 112 (1998).

[3] J. LI, Y. Y. RENARDY, AND M. RENARDY *Numerical Simulation of Breadup of a Viscous Drop in Simple Shear Flow through a Volume-of-Fluid Method*, Physics of Fluids, 12, 269 (2000).

[4] M. JAROSLAV *Fluid-structure Interaction Problems, Finite Element and Boundary Element Approaches - a Bibliography (1995-1998),* Finite Elements in Analysis and Design, v31, n3, Jan 1, 1999, p231-240.

[5] FLORYAN JM, RASMUSSEN H. *Numerical Methods for Viscous Flows with Moving Boundaries*, Applied Mechanics Reviews, V42:323-341 (1989).

[6] HIRT CW, AMSDEN AA, COOK JL. *An Arbitrary Lagrangian-Eulerian Computing Method for All Speeds*, Journal of Computational Physics, v14 227-253 (1974).

[7] G. TRYGGVASON, B. BUNNER, A. ESMAEELI, D. JURIC, N. AL-RAWAHI, W. TAUBER, J. HAN, S. NAS, Y.-J. JAN, *A Front-tracking Method for the Computations of Multiphase Flow*, J. Comput. Phys. 169 (2001) 708.

[8] T. Y. HOU, J. S. LOWENGRUB, AND M. J. SHELLEY, *Boundary Integral Methods for Multicomponent Fluids and Multiphase Materials*, J. Comput. Phys. 169, 302-362(2001)

[9] J. M. BOULTON-STONE AND BLAKE, *Gas Bubbles Bursting at a free surface*, J. Fluid Mech. 254, 437 (1991).

[10] S. XU, T. ASLAM, AND S. STEWART, *High resolution numerical simulation of ideal and non-ideal compressible reacting flows with embedded internal boundaries*, Combust. Theory Modelling 1(1997) 113-142.

[11] R. FEDKIW, *Coupling an Eulerian Fluid Calculation to a Lagrangian Solid Calculation with the Ghost Fluid Method*, J. of Comput. Phys. 175, 200-224 (2002).

[12] R. FEDKIW, B. MERRIMAN, R. DONAT, AND S. OSHER, *A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method)*, J. Comput. Phys. 152, 457 (1999).

[13] R. P. FEDKIW, T. ASLAM AND S. XU *The Ghost Fluid Method for Deflagration and Detonation Discontinuities*, J. Comput. Phys. 154, 393-427 (1999).

[14] B. VAN LEER *Flux Vector Splitting for the Euler Equations*, Technical report, ICASE Report, NASA Langley Research Center, 1995.

[15] P. WOODWARD AND P. COLELLA *The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks*, J. of Comput. Phys., 54, 115-173 (1984).

[16] T. J. BARTH, D. C. JESPERSEN *The Design and Application of Upwind Schemes on Unstructured Meshes*, AIAA Paper 89-0366, Jan. 1989.

[17] S. OSHER AND J. A. SETHIAN, *Fronts Propagating with Curvature-dependent Speed: Algorithms based on Hamilton-Jacobi Formulations*, J. Comput. Phys. 79, 12 (1988).

[18] J. SETHIAN , *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*, Cambridge, 1999.

[19] S. OSHER AND R. FEDKIW *Level Set Methods and Dynamic Implicit Surfaces*, Springr, 2002.

[20] E. TORO *Riemann Solver and Numerical Methods for Fluid Dynamics: A Practical Introduction*, 2nd Edition, Springer, 1999.

[21] M. SUSSMAN, P. SMEREKA, AND S. OSHER *A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow* J. Comput. Phys. 114, 146-159 (1994).

[22] M. SUSSMAN, A. ALMGREN, J. BELL, P. COLELLA, L. HOWELL AND M. WELCOME, *An Adaptive Level Set Approach for imcompressible Two-Phase Flows*, J. of Comput. Phys. 148, 81-124 (1999).

[23] Y. CHANG, T. HOU, B. MERRIMAN, S. OSHER, *A Level Set formulation of Eulerian Interface Capturing Methods for Incompressible Fluid Flows*, J. Comput. Phys. 124, 449-464 (1996).

[24] B. LAFAURIC, C. NANDONE, R. SCARDOVELLI, S. ZALESKI, G. ZANETTI, *Modelling Merging and Fragmentation in Multiphase Flows with SURFER*, J. Comput. Phys. 113, 134-147 (1994).

[25] W. MULDER, S. OSHER, J. SETHIAN, *Computing Interface Motion in Compressible Gas Dynamics*, J. Comput. Phys. 100, 209-228 (1992).

[26] B. COCKBURN, G. KARNIADAKIS, AND C. SHU *The Development of Discontinuous Galerkin Methods(Newport, RI, 1999)*, 3-50, Lect. Notes Comput. Sci. Eng., 11, Springer, Berlin, 2000. 65-02.

[27] J. E. FLAHERTY, L. KRIVODONOVA , J. F. REMACLE , AND M. S. SHEPHARD, *Aspects of Discontinuous Galerkin Methods for Hyperbolic Conservation laws*, Finite Element Analysis and Design, 38, 889-908 (2002).

[28] P. GEUZAINE, C. GRANDMONT, C. FARHAT *Design and Analysis of ALE Schemes with Provable Second-order Time-accuracy for Inviscid and Viscous Flow Simulations*, J. of Comput. Phys. 191, 206-227 (2003).

[29] C. FARHAT, P. GEUZAINE, G. BROWN *Application of a Three-field Nonlinear Fluid-structure Formulation to the Prediction of the Aeroelastic Parameters of an F-16 Fighter* Computers & Fluids, 32 3-29(2003)

[30] J. GLIMM, E. ISAACSON, D. MARCHESIN,D. AND O.MEBRYAN *Front tracking for hyperbolic ststems*, Adv.Appl.Math, 2, 91-119 (1981)

[31] J.M. HYMAN *Numerical methods for tracking interfaces*, Physica, 12D 396-407 (1984).

[32] R. YOUNG, J. GLIMM, AND B. BOSTON *Proceedings of the 5th International Workshop on Compressible Turbulent Mixing*, World Scientific, River Edge, NJ, 1995

[33] J. GLIMM, J. W. GROVE, X. L. LI, K.-M. SHYUE, Y. ZENG, AND Q. ZHANG, *Three-dimensional front tracking*, SIAM J. Sci. Comput., 19, 703-727 (1998).

Appendix

Approximation Functions of the Force on the Rigid Object and Corresponding Error Estimation

Three approximation functions for the Kronecker delta function δ has been given in Section 3.4. Here the error induced by each approximation will be discussed. The one-dimensional case will first be considered. Without

losing generality, we assume that the interface is at $x = 0$ and the level set function $\phi(x) = x$. For any function f

Let

$$f_0 = \int_{-\infty}^{+\infty} f(x)\delta(x)dx, \qquad \hat{f}_0 = \int_{-\infty}^{+\infty} f(x)\hat{\delta}(x)dx.$$

The approximation $\hat{\delta}(x)$ should have the property

$$\int_{-\infty}^{+\infty} \hat{\delta}(x)dx = 1.$$

In our case, the pressure p is a piecewise polynomial and our calculation is on the element level. Therefore, we can assume that $f(x)$ is a polynomial and write it as

$$f(x) = \sum_{k=0}^{n} a_k x^k.$$

It is obvious that $f_0 = f(0) = a_0$

Case (1): The approximation is

$$\hat{\delta}(x) = \begin{cases} \frac{1}{2\epsilon}(1 + cos\frac{\pi x}{\epsilon}), & \text{if } |x| < \epsilon \\ 0, & \text{if } |x| > \epsilon \end{cases} .$$

We have

$$\begin{aligned} |\hat{f}_0 - f_0| &= |\tfrac{1}{2\epsilon}\textstyle\int_{-\epsilon}^{+\epsilon}(1 + cos\frac{\pi\phi}{\epsilon})\sum_{k=0}^{n} a_k x^n dx - a_0| \\ &= |\tfrac{1}{2\epsilon}\textstyle\int_{-\epsilon}^{+\epsilon}\sum_{k=0}^{n} a_k x^n dx + \frac{1}{2\epsilon}\int_{-\epsilon}^{+\epsilon} cos\frac{\pi\phi}{\epsilon}\sum_{k=0}^{n} a_k x^n dx \\ &\leq |\tfrac{1}{2\epsilon}\textstyle\int_{-\epsilon}^{+\epsilon}\sum_{k=0}^{n} a_k x^n dx - a_0| + |\frac{1}{2\epsilon}\int_{-\epsilon}^{+\epsilon} cos\frac{\pi\phi}{\epsilon}\sum_{k=0}^{n} a_k x^n dx|. \end{aligned}$$

By using partial integration to the second term of the above inequality, we have

$$|\hat{f}_0 - f_0| = o(\epsilon^2).$$

Case (2): The approximation is

$$\hat{\delta}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}.$$

We have

$$\begin{aligned} |\hat{f}_0 - f_0| &= |\textstyle\int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \sum_{k=0}^{n} a_k x^n dx - a_0| \\ &= |\textstyle\int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \sum_{k=1}^{n} a_k x^n dx| \\ &= o(\sigma^2). \end{aligned}$$

The deduction of the above result uses the following results:
Let

$$g_n = \int_{-\infty}^{+\infty} \frac{x^n}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} dx.$$

Then

$$g_n = \begin{cases} 1, & \text{for } n = 0 \\ 0, & \text{for } n = 1 \\ (n-1)\sigma^2 g_{n-2}, & \text{for } n \geq 2 \end{cases} .$$

Case (3): The approximation function is

$$\hat{\delta}(x) = \begin{cases} 0, & \text{if } |x| > \epsilon \\ d(\epsilon^2 - x^2)^p, & \text{if } |x| \leq \epsilon \end{cases} .$$

here d is a constant and

$$d = \frac{\Gamma(p + 3/2)}{\epsilon^{2p+1}\sqrt{\pi}\Gamma(p+1)}.$$

For this approximation function of $\delta(x)$, it can also be verified that it is a positive function in $C_0^{p-1}(R)$ and satisfies (integration= 1). In this case, we have

$$\begin{aligned} |\hat{f}_0 - f_0| &= |\int_{-\epsilon}^{+\epsilon} d(\epsilon^2 - x^2)^p \sum_{k=0}^{n} a_k x^n dx - a_0| \\ &= |\int_{-\epsilon}^{+\epsilon} d(\epsilon^2 - x^2)^p \sum_{k=1}^{n} a_k x^n dx| \\ &= o(\epsilon^2). \end{aligned}$$

by using the fact that:

$$\begin{aligned} \int_{-\epsilon}^{+\epsilon} x^n d(\epsilon^2 - x^2)^p dx &= c_n \frac{\epsilon^n \Gamma(p+3/2)}{\Gamma(p + \frac{n+3}{2})} \\ &= o(\epsilon^n). \end{aligned}$$

c_n is a constant only depend on n.

Now consider the higher dimensional case. The integral we want to find is

$$\int_I f ds = \int_\Omega f \delta d\Omega,$$

with $I \in R^{n-1}$ and $\Omega \in R^n$.

With the approximation function $\hat{\delta}(x)$, the integration is

$$I_1 = \int_\Omega f(x)\hat{\delta}(\phi(x))dx, \qquad x \in R^n.$$

Noticing that the function $\hat{\delta}(x)$ vanishes quickly away from the interface I. We can approximate I_1 as

$$I_2 = \int_V f(x)\hat{\delta}(\phi(x))dx, \qquad x \in R^n.$$

here $V = I \times [-\epsilon, \epsilon]$

Rewrite I_2 as

$$I_2 = \int_I \int_{-\epsilon}^{\epsilon} f(x)\hat{\delta}(\phi(x)).$$

and noticing that the projection of the polynomial $f(x)$ on $[-\epsilon, \epsilon]$ is also a polynomial.

Therefore, we still have the conclusion as in the 1D case because the internal integration is exactly what we discussed in 1D case.