Comput Mech (2010) 45:321–334
DOI 10.1007/s00466-009-0449-5

ORIGINAL PAPER

Interpolation functions in the immersed boundary and finite element methods

Xingshi Wang · Lucy T. Zhang

Received: 30 July 2009 / Accepted: 11 November 2009 / Published online: 3 December 2009

Abstract In this paper, we review the existing interpolation functions and introduce a finite element interpolation function to be used in the immersed boundary and finite element methods. This straightforward finite element interpolation function for unstructured grids enables us to obtain a sharper interface that yields more accurate interfacial solutions. The solution accuracy is compared with the existing interpolation functions such as the discretized Dirac delta function and the reproducing kernel interpolation function. The finite element shape function is easy to implement and it naturally satisfies the reproducing condition. They are interpolated through only one element layer instead of smearing to several elements. A pressure jump is clearly captured at the fluid–solid interface. Two example problems are studied and results are compared with other numerical methods. A convergence test is thoroughly conducted for the independent fluid and solid meshes in a fluid–structure interaction system. The required mesh size ratio between the fluid and solid domains is obtained.

Keywords Immersed boundary method · Immersed finite element method · Convergence test · Fluid–structure interaction · Incompressibility

1 Introduction

Investigating complex physical phenomena in fluid–structure interactions requires reliable and efficient modeling technique and simulation tools. In the past few decades, numerous research efforts have been directed to method development for fluid–structure interactions. Methods developed by Tezduyar and his coworkers [1–4] are widely used in the simulation of fluid-particle and fluid–structure interactions, such as the studies on parachute aerodynamics [5]. The Arbitrary Lagrangian Eulerian (ALE) numerical approach [6–11] is another technique to accommodate the complicated fluid–structure interface. Nevertheless, mesh updating or re-meshing processes can be computationally expensive for the ALE algorithm. Glowinski et al. [12–14] developed the Distributed Lagrange Multiplier method to study particulate flows. Recently, researchers have applied the extended finite element method (XFEM) to study fluid–structure interactions [15–17].

Among the computational methods developed for fluid–structure interactions, one of the most noticeable contributions is the immersed boundary method [18,19], which was initially developed to study the blood flow around heart valves [20–26]. The mathematical formulation of the immersed boundary method employs a mixture of Eulerian and Lagrangian descriptions for fluid and solid domains. The interaction between the fluid and structure is accomplished by distributing nodal forces and interpolating nodal velocities between the Eulerian and Lagrangian domains through a smoothed approximation of the Dirac delta function. The advantage of the immersed boundary method is that the fluid–structure interface is tracked automatically, which circumvents costly mesh updating algorithms. The immersed boundary method inspired researchers around the world to further develop and enhance the accuracy and efficiency of the method. One of them is the immersed interface method [27,28] where LeVeque and Li obtained second-order accuracy in Peskin's immersed boundary method by imposing a derived second-order solution for Stokes fluid at the interface for finite difference method with uniform grid instead

X. Wang · L. T. Zhang (✉)
JEC 2049, Department of Mechanical, Aerospace,
and Nuclear Engineering, Rensselaer Polytechnic Institute,
Troy, NY 12180, USA
e-mail: zhanglucy@rpi.edu

of a smoothed Dirac delta function. This method was later extended to solve more complicated Navier–Stokes fluid flows [29–33]. Other methods include the extended immersed boundary method [34] which uses the finite element approach to calculate solid boundary nodal forces and the immersed boundary finite element method [35,36] which describes an immersed solid boundary in a fluid domain that is modeled using finite elements.

The immersed finite element method [37–42] is another method that extends the immersed boundary method to represent the background viscous fluid with an unstructured finite element mesh and nonlinear finite elements for the immersed deformable solid. Similar to the immersed boundary method, the fluid domain is defined on a fixed Eulerian grid and the solid domain is constructed independently with a Lagrangian mesh. The two major contributions of the immersed finite element method are: (1) the solid structure occupies volume space in the fluid domain and can be described using realistic material constitutive laws and (2) the reproducing kernel function [43–45] is introduced as the interpolation function in replace of the Dirac delta function to accommodate for nonuniform background meshes which are typically used in enhancing interfacial solutions. A new set of governing equations is derived and it allows the solid and fluid domains to be solved independently.

Due to the fact that the fluid and the solid computational domains are constructed independently, i.e. non-conforming mesh, an interpolation function is needed to interpolate or map the solutions at or near the interface from one domain to another. In both the immersed boundary method and the immersed finite element method, this interpolation function must satisfy the reproducing condition which requires that in the limit as the discretized size tends to be zero, the approximation should represent the given point value after the interpolation. The discretized Dirac delta function used in the immersed boundary method and the reproducing particle method used in the immersed finite element method satisfy this condition. However, it is computationally expensive to generate and solve for infinitesimally small grids; as the grid size gets larger, the interpolated region gets large, the interfacial solutions to be interpolated get smeared over several layers of fluid grids during the interpolation process, which numerically thickens the fluid–structure interface. Besides the reduced solution accuracy at the interface, these interpolation functions also cause problems when the solid boundary comes close to the fluid boundary, where a sufficient number of grid layers are required to reproduce the solutions.

Therefore, in this paper, we propose and implement a finite element interpolation function for non-uniform background fluid grid to capture a sharper fluid–structure interface than the reproducing kernel interpolation function used in the immersed finite element method and the Dirac delta function used in the immersed boundary method. They are interpolated through only one element layer instead of smearing to several elements. Using finite element interpolation function will produce sharper interfaces, capture expected pressure jumps without explicitly imposing solution jumps at the interface, and avoid the issue raised when the interface comes near the computational boundary. These advantages can be further enhanced when refined unstructured grids are used near the interface. The finite element interpolation function is easy to implement and it naturally satisfies the reproducing condition.

Furthermore, through the examples, a detailed convergence test will be conducted in this paper. We will also thoroughly examine the mesh ratio requirement for the solid and the fluid domains. When constructing independent fluid and solid meshes, the mesh size ratio between the fluid and the solid domains must be chosen with care. A heuristic estimation of mesh ratio of 2.0 is needed in order to prevent any 'leaking' phenomenon to occur [41]. The finding of this study can also be applied to other immersed boundary related methods.

The outline of this paper is as follows. In Sect. 2, we will present the formulations, review the discretized Dirac delta function and the reproducing kernel interpolation function, and introduce the proposed finite element interpolation function. In Sect. 3, two examples with detailed analysis will be shown. The solutions are examined thoroughly and compared with other published results. The convergence test will be performed and a range of allowable mesh size ratios between the fluid and solid domains will be identified. Finally, conclusions are drawn in Sect. 4.

2 Formulations

In this section, we will review the existing interpolation functions, i.e. the discretized Dirac delta function and the reproducing kernel function. We will then present the algorithm for the finite element interpolation function that produces sharper interface. The formulations are implemented based the immersed finite element method. However, it can be applied to any derived immersed methods that we discussed in the introduction section as long as the fluid and the solid domains are constructed independently and interpolation functions are required for any interface communications. For completeness, the original algorithm of the immersed finite element method is reviewed first.

2.1 Immersed finite element method algorithm

The fluid is assumed to be incompressible and fulfills the entire computational domain Ω that is composed of two distinct domains: the actual fluid $(\Omega^f, \mathbf{x})$ and the immersed solid $(\Omega^s, \mathbf{x})$. Since the fluid is everywhere in the domain, the solid

volume is also occupied by the fictitious fluid. With the current configuration of the solid $\mathbf{x}^s$ and its velocity $\mathbf{v}^s$ at time t, the fluid–solid interaction force in the solid domain $\mathbf{f}^{FSI,s}$ is derived as:

$$\mathbf{f}^{FSI,s} = -(\rho^s - \rho^f)\ddot{\mathbf{u}}^s + \nabla \cdot \sigma^s - \nabla \cdot \sigma^f + (\rho^s - \rho^f)\mathbf{g} \quad \text{in} \quad \Omega^s, \tag{1}$$

where ρ^s and ρ^f are the solid and fluid densities; σ^s and σ^f are the internal stress of solid and fluid, respectively; $\mathbf{u}^s$ is the solid displacement and $\mathbf{g}$ is the body force. The internal stress of the solid σ^s is determined by the material constitutive law.

The interaction force obtained from the solid domain Eq. (1) is then distributed to its surrounding fluid domain through an interpolation procedure as follows:

$$\mathbf{f}^{FSI,f} = \int_{\Omega^s} \mathbf{f}^{FSI,s}\phi(\mathbf{x} - \mathbf{x}^s)d\Omega, \tag{2}$$

where $\phi(\mathbf{x}-\mathbf{x}^s)$ is the interpolation function, which is a function of the distance between a solid node and its surrounding fluid nodes in its influence domain.

After the force distribution, the interaction force acts as an external force in the fluid domain. The governing equations for the fluid domain are the Navier–Stokes equations:

$$\nabla \cdot \mathbf{v}^f = 0, \tag{3}$$

$$\rho(\mathbf{v}^f_{,t} + \mathbf{v}^f \cdot \nabla \mathbf{v}^f) = -\nabla p^f + \mu^f \nabla^2 \mathbf{v}^f + \mathbf{f}^{FSI,f}, \quad \text{in } \Omega. \tag{4}$$

The fluid velocities $\mathbf{v}^f$ and pressure p^f are solved implicitly with stabilized galerkin method. The fluid velocities are then interpolated back to the solid domain using the same interpolation function as in Eq. (2):

$$\mathbf{v}^s = \int_{\Omega} \mathbf{v}^f \phi(\mathbf{x} - \mathbf{x}^s)d\Omega. \tag{5}$$

Once the nodal solid velocities are obtained, the nodal displacement can be updated explicitly in time:

$$\mathbf{u}^{s,n+1} = \mathbf{v}^{s,n+1}\Delta t. \tag{6}$$

The entire procedure is then repeated for the next time step starting from Eq. (1). Both solid and fluid equations are solved using finite element method with linear elements.

2.2 Interpolation functions

The interpolation function, ϕ, to be used in Eqs. (2) and (5) must satisfy the following conditions: 1. The continuity of ϕ is imposed across the fluid–solid interface to satisfy the continuity of velocity and forces to be interpolated. 2. The interpolation functions must be complete and reproducible, i.e.

$$\int_{-\infty}^{+\infty} \phi(x - y)dy = 1, \tag{7}$$

and the reproduced function $u(x)$ is:

$$\int_{-\infty}^{+\infty} u(y)\phi(x - y)dy = u(x), \tag{8}$$

where x is the Lagrangian solid domain and y is the Eulerian fluid domain. The discretized forms of the above conditions become:

$$\sum_I \phi_I = 1, \tag{9a}$$

$$\sum_I u_I \phi_I = u^h, \tag{9b}$$

where I is the index for the number of nodes in the support domain. This support is typically bounded. The goal is to have the smallest possible support in order to maintain the sharpness of the interface for accuracy and to conserve computational efficiency.

To ensure the conservation of energy, the interpolation functions in both distribution of forces and interpolation of velocity field must be the same at each time step, i.e. the support of a solid point has to be identical in the distribution and interpolation processes [19].

There are many ways to generate an interpolation function that can satisfy all the above mentioned conditions. The choices depend on several factors: (1) the nature of the fluid mesh, whether it has structured or unstructured grids; (2) the size of the support, which results in varying degrees of interface sharpness; (3) easiness in implementation and efficiency. The following describes three different interpolation functions: the discretized dirac delta function, the reproducing kernel function, and our proposed finite element interpolation function.

2.2.1 Discretized Dirac delta function

In the original immersed boundary method, a discretized Dirac delta function $\phi(r)$, is used as the interpolation function, where r is the distance between a solid node and a surrounding fluid node normalized by the mesh size h. In Ref. [19], the function is defined such that:

$$\phi(r) = 0 \quad \text{for } |r| \geq 2, \tag{10a}$$

$$\sum_{j\text{even}} \phi(r - j) = \sum_{j\text{odd}} \phi(r - j) = 0.5 \quad \text{for all real } r, \tag{10b}$$

$$\sum_j (r - j)\phi(r - j) = 0 \quad \text{for all real } r, \tag{10c}$$

where j is an integer value for uniform grid index. The support is bounded to a 2-grid shift on each side of the point $\mathbf{x}$.

In a 1-D domain, the support covers 4 grid units. These equations are to maintain the reproducing conditions listed in Eq. (9) for uniform grid size. This function has C^0 continuity and is only appropriate when structured fluid mesh is used.

The above equations can be stipulated in a simpler form as the interpolation function in the immersed boundary method for uniform fluid background meshes [19],

$$\phi(r) = \begin{cases} \frac{1}{4}\left(1+\cos\left(\frac{\pi|r|}{2}\right)\right), & |r| \le 2 \\ 0, & |r| > 2. \end{cases} \tag{11}$$

In a 3-D model, the discretized Dirac delta function $\delta^h(\mathbf{x})$ becomes

$$\delta^h(\mathbf{x}) = \frac{1}{h^3}\phi\left(\frac{x_1}{h}\right)\phi\left(\frac{x_2}{h}\right)\phi\left(\frac{x_3}{h}\right), \tag{12}$$

where x_1, x_2, and x_3 are the distance between a solid node and a surrounding fluid node in the x, y, z directions, respectively.

2.2.2 Reproducing kernel function

In the immersed finite element method, the reproducing kernel interpolation function is employed to acquire a higher order interpolation and it is also suitable for both uniform and nonuniform meshes. This reproducing kernel particle method was first proposed as one of the meshfree methods [43,44,46,47]. A dilation parameter is introduced to control the size of the window function or the support size. When it is used as an interpolation function, this constant scales and controls the size of the fluid–solid interface. Users need to define this constant prior to the simulation in order to achieve the desired fluid–solid interfacial thickness. However, the interfacial thickness cannot be too small because a sufficient number of surrounding nodes are required to compute the interpolation function. For both Dirac delta function and reproducing kernel function interpolations, the size of the influence domain Ω_ϕ is at least 4 element-layers when uniform mesh is used. Larger influence domain will induce a smoothed or smeared velocity and force fields at the interface after the interpolations, which is not desired when more accurate interfacial solutions are required. Moreover, a search algorithm must be completed in each time step in order to identify new sets of neighbors after the solid advances to a new position from the previous time step.

As illustrated in Ref. [48], both wavelet and smooth particle hydrodynamics (SPH) methods belong to a class of reproducing kernel methods where the "reproduced" function $u^R(x)$ is derived as:

$$u^R(x) = \int_{-\infty}^{+\infty} u(y)\phi(x-y)dy, \tag{13}$$

with a projection operator or a window function $\phi(x)$.

The reproducing condition requires that up to n^{th} order polynomial can be reproduced, i.e.,

$$x^n = \int_{-\infty}^{+\infty} y^n \phi(x-y)dy. \tag{14}$$

To satisfy the reproducing condition, a correction function $C(x; x-y)$ is introduced in the finite domain of influence or support, so that the window function yields:

$$\int_\Omega C(x; x-y)a^{-1}\phi\left(\frac{x-y}{a}\right)d\Omega = 1 \tag{15}$$

where a is the dilation parameter or refinement of the window function and $x-y$ is the distance between the node x and its support node y. The additional constant, a^{-1}, scales the window function so that the integral over the domain of support equals one such that $\int_\Omega \phi(x)d\Omega = 1$. An example of a high order window function $\phi(x)$ for uniform grids is a cubic spline,

$$\phi\left(\frac{x-y}{a}\right) = \begin{cases} \frac{2}{3} - \left(\frac{x-y}{a}\right)^2\left(1-\frac{|x-y|}{2a}\right), & 0 \le \frac{|x-y|}{a} < 1 \\ \frac{1}{6}\left(2-\frac{|x-y|}{a}\right)^3, & 1 \le \frac{|x-y|}{a} \le 2 \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

The discretized reconstruction of the delta function for nonuniform spacing can be written as

$$\phi_I(x) = C(x; x-x_I)a^{-1}\phi\left(\frac{x-x_I}{a}\right)\Delta x_I. \tag{17}$$

The correction function C needs to be derived for nonuniform grids, as shown in [46]. The window function is used as the interpolation function for the interpolation of the nodal velocities and the distribution of the nodal forces between the fluid and solid domains.

2.2.3 Finite element interpolation function

In this section, we will introduce the finite element interpolation function that is capable of (1) producing sharp fluid–solid interface, (2) handling nonuniform or unstructured mesh, and (3) obtaining accurate interfacial solutions without the requirement of a minimum number of element layers in between a solid boundary and a fluid boundary. (This is explained further in the examples.)

The finite element basis function naturally satisfies the reproducing conditions. Therefore, no correction function is needed in the implementation. The influence domain of a solid node can be easily identified by locating the fluid element that the solid node resides in. Most importantly, it does not require a minimum number of elements near the boundary in order to produce enough grid layers for the interpolations when the interface approaches a fluid boundary, shown

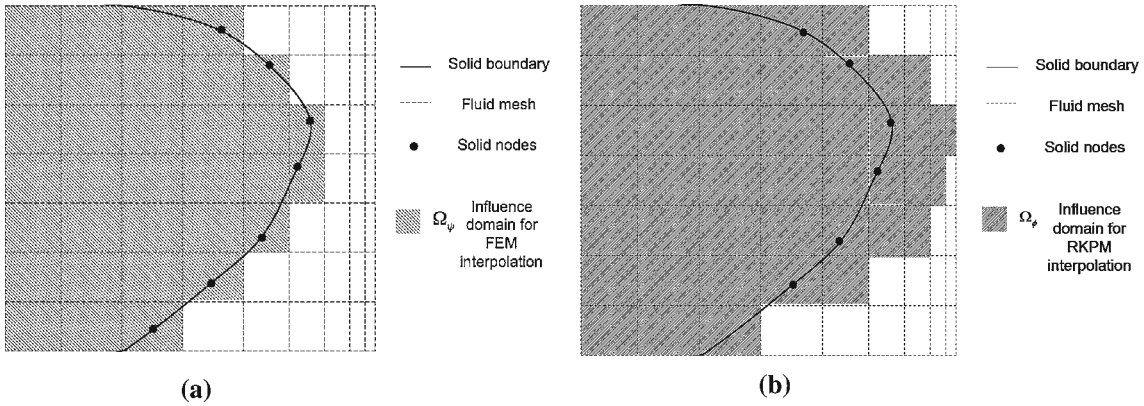

Fig. 1 Comparison of influence domains identified by **a** finite element interpolation function (Ω_ψ) and **b** reproducing kernel interpolation function (Ω_ϕ) in a nonuniform fluid mesh

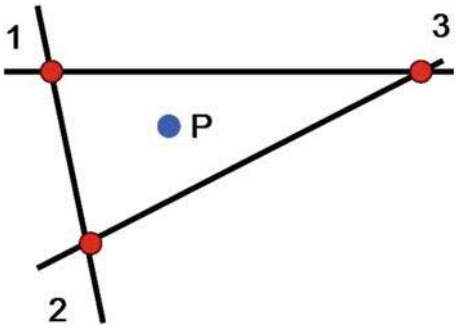

Fig. 2 Construction of finite element interpolation function for a point in a 2-D triangular element

in Fig. 1. The solid domain is always contained in the fluid domain, so the finite element interpolation functions can be constructed based on the relative position of the solid point within a fluid element.

The whole procedure can be performed in three steps: (1) identify the fluid element that contains a particular solid node; (2) calculate interpolation functions, and (3) conduct velocity interpolation or force distribution. At every time step, a search algorithm is performed to identify the corresponding fluid element that a solid node I resides in. The search algorithm is built based on "point in triangle test" for 2-D [49]. It can be easily extended as a point in tetrahedral test for 3-D or other types of elements (quadrilateral and hexahedral) due to the fact that these elements can always be divided into combinations of triangles or tetrahedrals. The element is considered as its influence domain $\Omega_{\psi I}$. The symbol ψ is to represent finite element interpolation influence domain. An example is shown in Fig. 2 where P is a solid node with coordinate (x_I^s, y_I^s) contained in a trianglular element with local node numbers J, which are labeled as 1, 2, 3 and coordinates of (x_1^e, y_1^e), (x_2^e, y_2^e), (x_3^e, y_3^e), respectively.

The local interpolation function N_J^e must satisfy the reproducing conditions such that:

$$\sum_{J=1}^{3} N_J^e = 1 \quad \mathbf{x}_J \in \Omega_{\psi I}, \tag{18a}$$

$$\sum_{J=1}^{3} N_J^e x_J^e = x^s \quad \mathbf{x}_J \in \Omega_{\psi I}, \tag{18b}$$

$$\sum_{J=1}^{3} N_J^e y_J^e = y^s \quad \mathbf{x}_J \in \Omega_{\psi I}. \tag{18c}$$

We can then assemble them into a matrix form as: $\mathbf{N}^e\mathbf{M}^e = \mathbf{P}$, where

$$\mathbf{P}(x^s, y^s) = \begin{pmatrix} 1 & x^s & y^s \end{pmatrix},$$

$$\mathbf{M}^e = \begin{pmatrix} 1 & x_1^e & y_1^e \\ 1 & x_2^e & y_2^e \\ 1 & x_3^e & y_3^e \end{pmatrix}.$$

Finally, the finite element interpolation function $\mathbf{N}^e$ can be solved as:

$$\mathbf{N}^e(x^s, y^s) = \mathbf{P}(x^s, y^s)(\mathbf{M}^e)^{-1}. \tag{19}$$

It is noted that when the solid node happens to be on the edge of a fluid element, the inverse of $\mathbf{M}^e$ becomes singular. However, as long as the distance from the solid point to an element boundary is larger than certain machine precision, e.g. 10^{-6}, the interpolation function can be solved without any difficulty. Once the finite element interpolation function is constructed, the force can be distributed from the fluid nodes to the solid nodes and the nodal solid velocities can be interpolated from the fluid velocities as:

$$\mathbf{v}^s = \sum_J N_J \mathbf{v}_J^f \quad \mathbf{x}_J \in \Omega_{\psi I}, \tag{20a}$$

$$\mathbf{f}_J^{FSI,f} = N_J \mathbf{f}^{FSI,s} \quad \mathbf{x}_J \in \Omega_{\psi I}. \tag{20b}$$

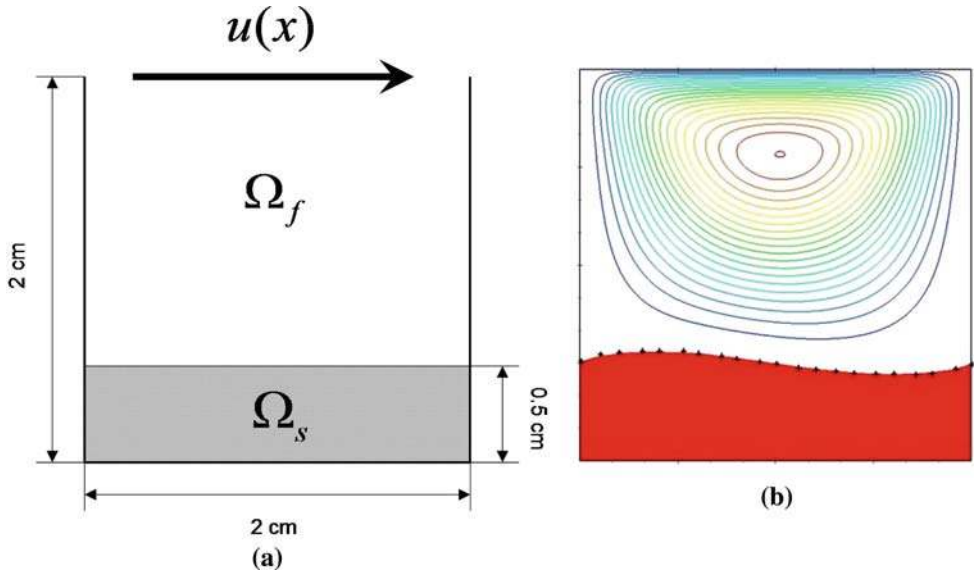

Fig. 3 A deformable wall in a cavity driven flow: **a** initial configuration (problem statement) and **b** steady state solution with streamlines ('+' represent the interface obtained by Dunne [50])

3 Examples

3.1 Example 1: an elastic wall in a lid-driven cavity flow

In this example, we study the deformation of a flexible wall that is placed at the bottom of a lid-driven cavity filled with fluid. This study was first presented in Ref. [50] using the Arbitrary Lagrangian Eulerian method. It was then repeated and confirmed by Zhao et al. in Ref. [51]. For validation purposes, we first compare our steady-state solution with the ones reported in Ref. [50] and then focus our study on the solutions captured at the fluid–solid interface, especially the pressure jump that is yielded by using the finite element interpolation shape function. A comprehensive convergence study of this fluid–structure interaction algorithm is also performed.

A hyperelastic wall with height 0.5 cm and width 2 cm is located at lower bottom of a cavity with size 2 cm × 2 cm. The space above is filled with fluid, as shown in Fig. 3a. The top lid of the cavity is driven by spatially dependent velocity functions:

$$u = 0.5 \begin{cases} \sin^2(\pi x/0.6), & x \in [0.0, 0.3]; \\ 1, & x \in (0.3, 1.7); \\ \sin^2(\pi(x-2.0)/0.6), & x \in [1.7, 2.0]. \end{cases} \quad (\text{cm/s}) \tag{21}$$

The time step used is 0.05 s. The fluid has density of $\rho^f = 1.0$ g/cm^3 and viscosity of $\mu^f = 0.2$ dyn· s/cm. The solid is described as Neo-Hookean hyperelastic material. For a hyperelastic material with Mooney-Rivlin [34] or Neo-Hookean description, the strain energy function W is given as:

$$W = C_1(J_1 - 3) + C_2(J_2 - 3) + \frac{\kappa}{2}(J_3 - 1)^2, \tag{22}$$

with

$$J_1 = I_1 I_3^{-1/3}, \; J_2 = I_2 I_3^{-2/3}, \; J_3 = I_3^{1/2}, \tag{23}$$

in which C_1, C_2, and κ are the material constants and I_1, I_2, and I_3 are functions of the invariants of the Cauchy-Green deformation tensor $\mathbf{C}$ which is defined as $C_{ij} = F_{im}F_{jm}$. When C_2 and κ are 0, the Mooney-Rivlin description can be reduced to Neo-Hookean description. Here, Neo-Hookean material is used with $C_1 = 0.1$ dyn/cm^2. When large displacements and deformations occur, the second Piola-Kirchhoff stress S_{ij} can be derived based on the Green-Lagrangian strain E_{ij}:

$$S_{ij} = \frac{\partial W}{\partial E_{ij}} \quad \text{and} \quad E_{ij} = \frac{1}{2}(C_{ij} - \delta_{ij}). \tag{24}$$

The first Piola-Kirchhoff stress P_{ij} can be transformed from the second Piola-Kirchhoff stress S_{ij} by applying $P_{ij} = S_{ik}F_{jk}$. The Cauchy stress σ can be transformed from the first Piola-Kirchhoff stress, $\mathbf{P}$. The transformations between stresses can be found in Ref. [52].

The deformed wall reaches a steady state at $t = 8$ s, shown in Fig. 3. The streamline of the fluid is also presented. As shown on the figure, the steady state solution is nearly identical with the one reported in [50].

3.1.1 Sharp interface solutions

To examine the accuracy of the solutions at the interface from the interfacial thickness, we will focus on the pressure jump occurring at the fluid–solid interface. This pressure jump is expected to exist at any x-value along the fluid–solid interface. If the interpolation function does not produce a sharp interface, then this pressure difference is smeared through several layers of grids and the solutions at this interface cannot be captured accurately. Here, we examine the pressure

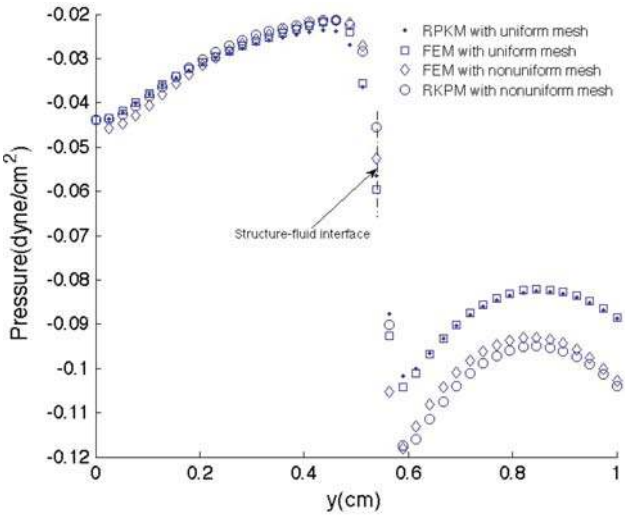

Fig. 4 Comparison of pressure profiles at $x = 0.5$ cm along y-axis using finite element interpolation and reproducing kernel interpolation function with both uniform and nonuniform mesh

at the interface by choosing an arbitrary location, say $x = 0.5$ cm, at the steady state. The interface is located at $y = 0.55$ cm when $x = 0.5$ cm.

Four results are compared: (1) using finite element interpolation with slight refinement near the interface while having slight coarser mesh size everywhere else (nonuniform grid), (2) using finite element interpolation with uniform grid, (3) using reproducing kernel interpolation function with nonuniform grid, and (4) using reproducing kernel interpolation function with uniform grid. The pressure profiles from these four cases along $x = 0.5$ cm are shown in Fig. 4.

The results show that by applying nonuniform mesh with finer mesh around the solid-fluid interface, the interface is *sharper* comparing with the uniform mesh while maintaining the overall accuracy everywhere else. When the finite element interpolation function is used as compared to the reproducing kernel interpolation function, it again yields a more defined interface. As the interface gets narrower, the distribution of the forces and the interpolation of the velocities are only spread within a thin layer, which reduce the errors generated during the interpolation processes and yield more accurate solutions. When uniform mesh is used, using finite element interpolation function can reduce the interfacial thickness by as much as 65.4% comparing to that of the reproducing kernel interpolation function. It can be further reduced by 80.3% when nonuniform mesh is used.

We can further look into the pressure jump by calculating the derivative of the pressure from these three cases. Ideally, this pressure jump should produce a Dirac delta function for the pressure derivative. As shown in Fig. 5, the finite element interpolation function with nonuniform mesh yields a very narrow band and has the maximum peak value, which has the closest resemblance of a Dirac delta function. From this result, we observe that the finite element interpolation can capture a sharper interface with a smaller support domain while conserving the reproducing condition. It can be further improved when nonuniform mesh is used.

3.1.2 Convergence studies

A comprehensive convergence test is performed using this example. We pay special attention to the allowable fluid–solid mesh size ratios that can be used to yield convergent solutions. For a coupled fluid–structure problem, the convergence rate is computed independently with Lagrangian mesh element size and Eulerian grid spacing. Since there is no analytical solution for this problem, the errors of fluid velocity and solid displacement are calculated based on the solution obtained from a finely discretized system. The convergence of the solid displacement is calculated by refining the Lagrangian mesh while keeping the Eulerian mesh fixed at a refined state. Similarly, the convergence of the N-S solver is studied by refining the fluid mesh while keeping the solid mesh at a very fine resolution. Both components are performed with uniform mesh spacings for consistencies. Errors in the fluid velocity and solid displacement are calculated in L_2 norms for steady state solutions.

The error in the fluid velocity field, $e_{\mathbf{v}^f}$ is defined as:

$$e_{\mathbf{v}^f} = \left(\frac{1}{A_{\Omega^f}} \left| \int_{\Omega^f} (\mathbf{v}^2 - \tilde{\mathbf{v}}^2) d\Omega \right| \right)^{1/2}, \tag{25}$$

where $\mathbf{v}$ is the velocity field to be examined; $\tilde{\mathbf{v}}$ is the velocity of the reference solution. The error is then normalized with the total area of the fluid domain A_{Ω^f}. The discretized form is as follows:

$$e_{\mathbf{v}^f} = \left(\left| \frac{1}{N} \sum_I \mathbf{v}_I{}^2 - \frac{1}{\tilde{N}} \sum_J \tilde{\mathbf{v}}_J^2 \right| \right)^{1/2}, \tag{26}$$

where I and J are the nodal index in the fluid domain for the solutions to be examined and the reference solutions, respectively. The sum of the kinetic energy in each solution is then normalized by the number of nodes used in the fluid domain where N is the number of nodes in the mesh to be examined and $\tilde{N}$ is the number of nodes in the reference mesh.

The error in the solid displacement, $e_{\mathbf{d}^s}$ is defined as:

$$e_{\mathbf{d}^s} = \left(\frac{1}{l_{\Gamma^{FSI}}} \int_x (y^s - \tilde{y}^s)^2 dx \right)^{1/2}, \tag{27}$$

where $l_{\Gamma^{FSI}}$ represents the total length of the fluid–solid interface in the reference solution Γ^{FSI}. The errors are evaluated on the position of the interface in the y-direction along the

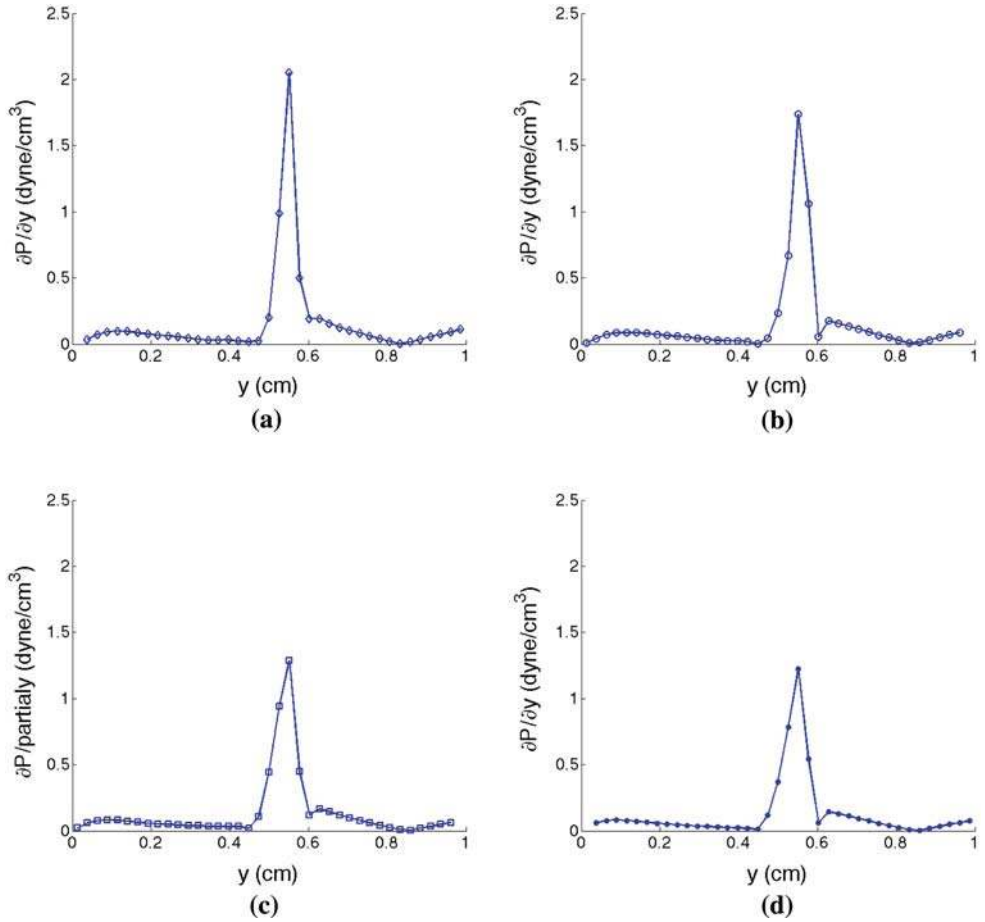

Fig. 5 Interface width or pressure jump can be observed from the derivative of pressure across an interface. **a** Finite element interpolation with nonuniform mesh. **b** Reproducing kernel interpolation with nonuniform mesh. **c** Finite element interpolation with uniform mesh. **d** Reproducing kernel interpolation with uniform mesh

x-axis, i.e. from $x = 0$ to $x = 2$. y^s and $\tilde{y}^s$ are the y-coordinates of the fluid–solid interface obtained from the mesh to be examined and the reference mesh, respectively.

The discretized form is as follows:

$$e_{\mathbf{d}^s} = \left(\frac{1}{N^s} \sum_J (y_J^h - \tilde{y}_J^h)^2 \right)^{1/2}, \tag{28}$$

where N^s is the number of solid nodes at the fluid–solid interface in the reference mesh and J is the nodal index of these point. y_J^h and $\tilde{y}_J^h$ are the y-coordinate of the solid node at a particular x-position along the interfaces obtained from the mesh to be examined and the reference mesh, respectively. When studying the convergence of the fluid mesh with a fixed finely-discretized solid mesh, the number of solid nodes remain the same on the interface. However, for the convergence study of the solid mesh with varying solid mesh size, the number of solid nodes are different. To calculate the error at any x with the finely-discretized reference mesh, the displacement solution of the mesh to be examined is interpolated, denoted by h, for that particular x along the continuous interface.

The convergence of the fluid mesh is examined by using a series of discretized fluid meshes (from 16384 to 256 elements) while keeping the solid mesh fixed. The data is listed in Table 1. The convergence of the fluid mesh element size is shown in Fig. 6. The error in the fluid velocity field in L_2 norm yields a convergence rate of $O(h^{0.8})$ and the error in the solid displacement in L_2 norm has a convergence rate of $O(h^{1.0})$.

Similarly, the convergence of the solid mesh is performed through a series of discretized meshes (from 8192 to 1296 elements) while the fluid mesh is fixed. The data used is shown in Table 2. The convergence of the solid mesh is shown in Fig. 7. The error in the fluid velocity field in L_2 norm yields a convergence rate of $O(h^{1.2})$ and the error in the solid displacement in L_2 norm yields a convergence rate of $O(h^{1.7})$.

When constructing independent fluid and solid meshes, the mesh size ratio between the fluid domain and the solid domain must be chosen with care. A heuristic estimation of mesh ratio of 2.0 is needed in order to prevent any 'leaking'

Table 1 Data used for convergence test of the fluid mesh (with solid mesh 128 × 64)

Fluid mesh	128 × 128	120 × 120	110 × 110	100 × 100	90 × 90	80 × 80	70 × 70	64 × 64	32 × 32	16 × 16
Mesh ratio	2.0	2.33	2.77	3.4	4.14	5.2	6.85	8	32	128
Error($\mathbf{v}^f$) (10^{-2} cm/s)	Reference	1.1	1.4	1.6	1.8	2.0	2.2	2.4	3.7	5.8
Error($\mathbf{d}^s$) (10^{-3} cm)	Reference	2.39	2.40	2.41	2.52	2.54	3.00	3.48	3.86	8.19

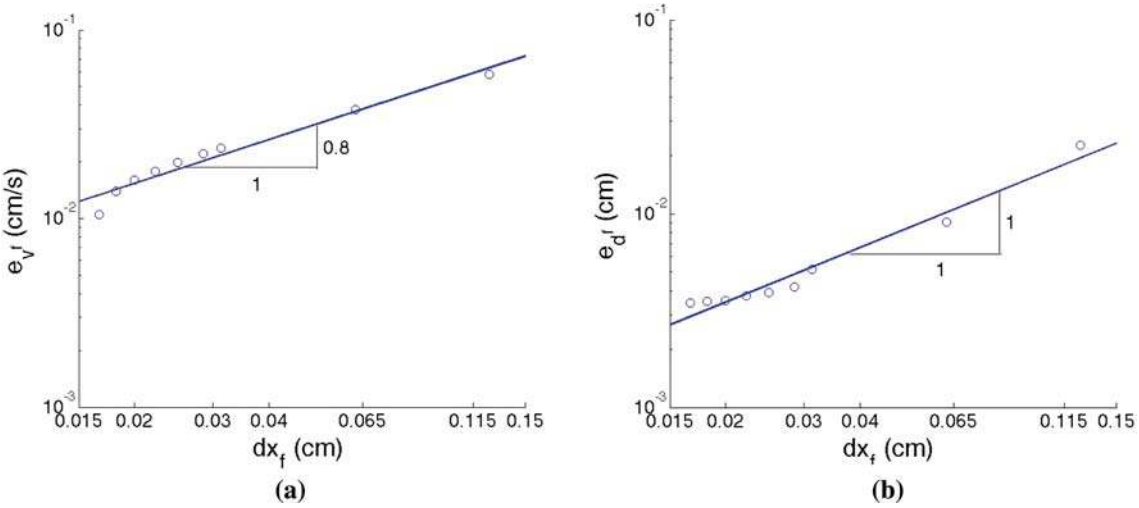

Fig. 6 Convergence test of the fluid mesh. **a** Error in fluid velocity field. **b** Error in solid displacement

Table 2 Data used for convergence test of the solid mesh (with fluid mesh 128 × 128)

Solid mesh	128 × 64	128 × 32	120 × 30	100 × 25	88 × 22	80 × 20	72 × 18
Mesh ratio	3.4	1.64	1.43	1.0	0.78	0.64	0.52
Error($\mathbf{v}^f$) (10^{-4} cm/s)	Reference	0.08	0.09	0.12	0.16	0.17	0.2
Error($\mathbf{d}^s$) (10^{-3} cm)	Reference	0.14	0.15	0.24	0.31	0.36	0.52

phenomenon to occur [41]. This 'leaking' refers to the numerical artifact that appears when the fluid element size is much smaller than that of the solid. This issue has never been investigated in detail. Through a series of tests, we found this mesh ratio to be approximately 0.5. If the fluid element is smaller than half of the solid element, then we indeed observed the fluid to 'penetrate' into the solid domain. This numerical error, of course, leads to unrealistic physical behavior of the interaction. This size ratio is found to be very consistent for all sets of mesh resolutions regardless of being coarse or fine. Here, we compare the pressure fields from 2 sets of meshes with mesh ratios of 3.4 and 0.025, respectively (Fig. 8). It is obvious that the one with mesh ratio of 3.4 yields the correct pressure distribution while the one from mesh ratio of 0.025 has the wrong solution. A source of 'leaking' is found in the middle of the solid domain to yield an unrealistically high concentrated pressure spot.

3.2 Example 2: disk in a lid-driven cavity

In this example, we study a deformable disk in a lid-driven cavity. The cavity is 1 cm by 1 cm and the radius of the disk is 0.2 cm centered at (0.6 cm, 0.5 cm) in the cavity, as shown in Fig. 9.

The top lid starts to move horizontally at a constant speed of $U_0 = 1$ cm/s at $t = 0$ s. The fluid viscosity is $\mu^f = 0.01$ dyn $\cdot$ s/cm^2. The solid material is modeled as hyperelastic model with Neo-Hookean description that has a material constant ranging from $C_1 = 5.0$ dyn/cm^2 (hard) to 0.05 dyn/cm^2 (soft). This example was briefly illustrated in Ref. [51]. In our study, we perform a more thorough analysis and examine the motion and deformation of the disk as time progresses. The snapshots of the solid deformation at different time steps are shown in Fig. 10. It is apparent that the hard disk deforms much less than the soft disk.

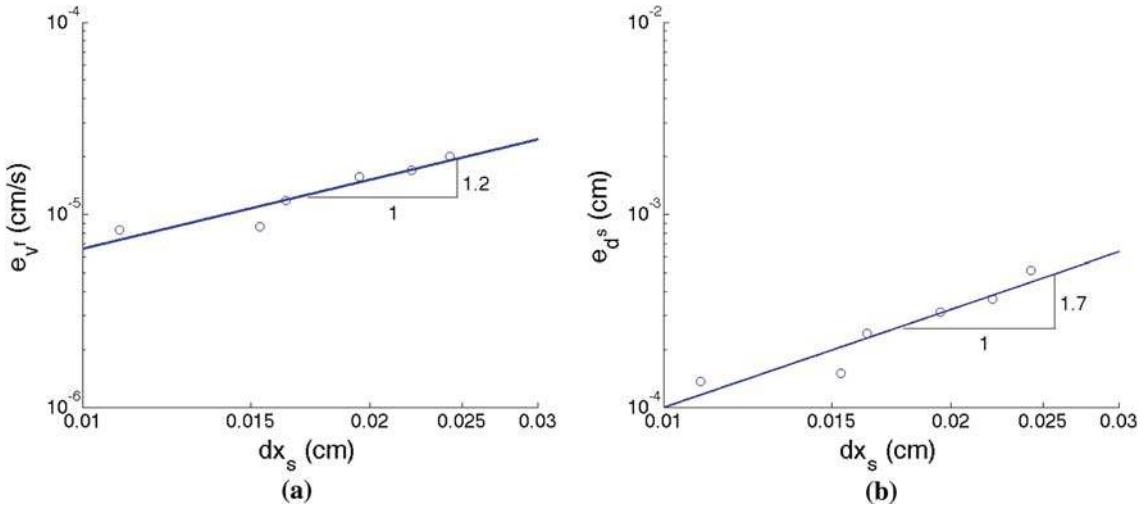

Fig. 7 Convergence test of the solid mesh. **a** Error in fluid velocity field. **b** Error in solid displacement

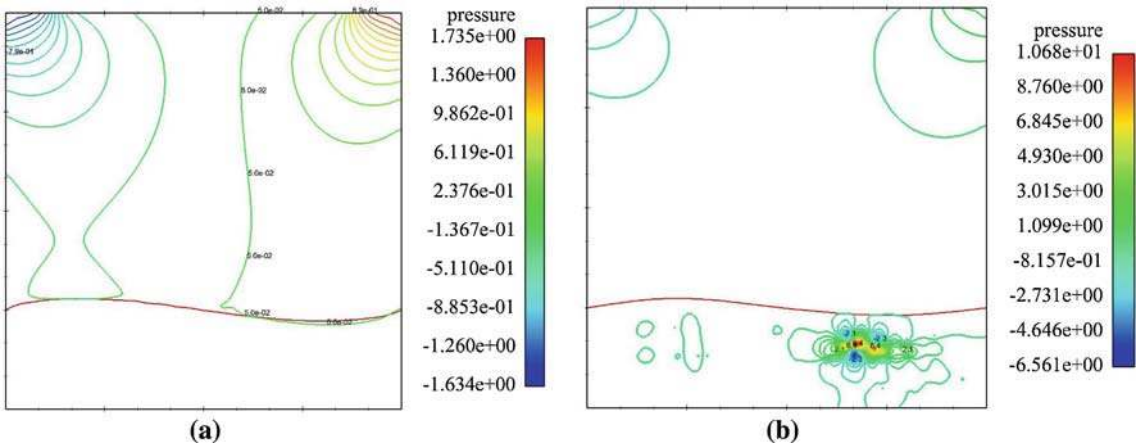

Fig. 8 **a** Pressure contours at steady-state when mesh ratio is 3.4 (>0.5). **b** Pressure contours when mesh ratio is 0.025 (<0.5)

The trajectories of the disc centroids for both the hard and the soft disks are tracked, shown in Fig. 11. The result shows that both disks eventually settle down at a fixed position in the fluid as they reach the steady states. It takes the hard disk approximately 60 s while the soft disk takes only 40 s. The centroid of the soft disk comes closer to the top boundary since the soft disk deforms more near the top region of the cavity. Similar observations are also reported in [51,53].

3.2.1 Incompressibility constraints

In the cluster of the immersed methods, there exists a common numerical issue when dealing with a solid object immersed in an incompressible fluid domain. When the fluid is incompressible the solid must remain incompressible in the fluid as well, due to the fact that the solid must always remain immersed in the fluid domain. The detailed discussion can be found in [42]. In some cases, a volume correction algorithm is required to enforce the incompressibility constraints. In the soft disk case example, it can be clearly seen in Fig. 12 that without the correction, there is a significant stretching of the soft disk as time progresses. The volume change is particularly noticeable as it comes close to a moving boundary, which induces large velocity gradients, hence high pressure, between the fluid boundary and solid boundary. In this case, constraining the incompressibility in the fluid domain alone is no longer sufficient in maintaining the solid volume, instead, a volume correction algorithm is needed. With the volume correction, the relative volume change is significantly reduced.

A comparison of the relative volume change from before and after applying the volume correction algorithm is shown

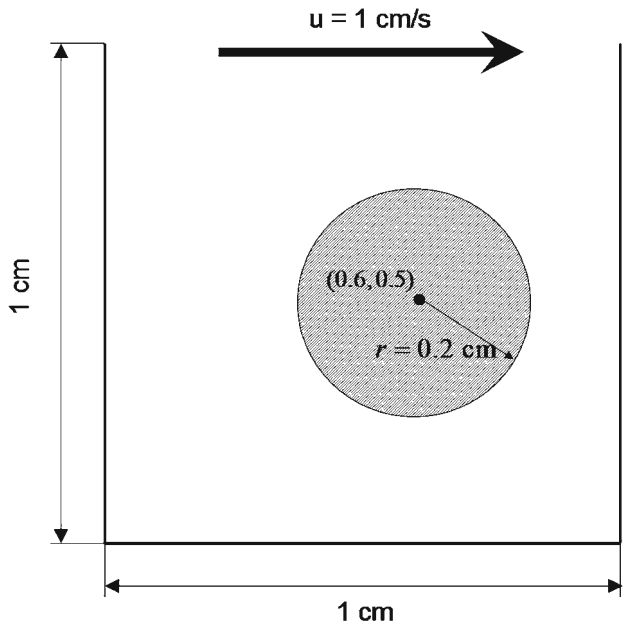

Fig. 9 A deformable disk in a lid-driven cavity

in Fig. 13a. Without the volume correction, the soft disk can increase as much as nearly 20% of its original volume by $t = 5$ s.

For the hard case, it is expected that the disk deforms much less compared to the soft case. We observed that as the solid material gets more stiff, the volume change is much less severe. The maximum volume change reaches only 1.6%, as shown in Fig. 13b. We, again, noticed that during $t = 4.0$ s to $t = 6.0$ s, when the disk is near the top region of the cavity, the volume changing rate increases sharply. This study confirms that when the solid-fluid interface is near a moving fluid boundary that can generate a large velocity gradient the volume correction algorithm is needed.

Even though the magnitude of the volume correction is small, it does introduce slight numerical error. Thus, it should be used only absolutely necessary. Softer material requires more frequent volume corrections. Applying this algorithm can also be computationally costly, especially with dense solid mesh. To avoid using this volume correction step, we can also consider using smaller time steps and refining fluid mesh near the driving boundary. However, these choices can also be computationally expensive.

3.2.2 Capturing the interface near the boundary

Using finite element interpolation, the solid interface can be interpolated through one grid layer, unlike using other interpolation function where several layers are required to

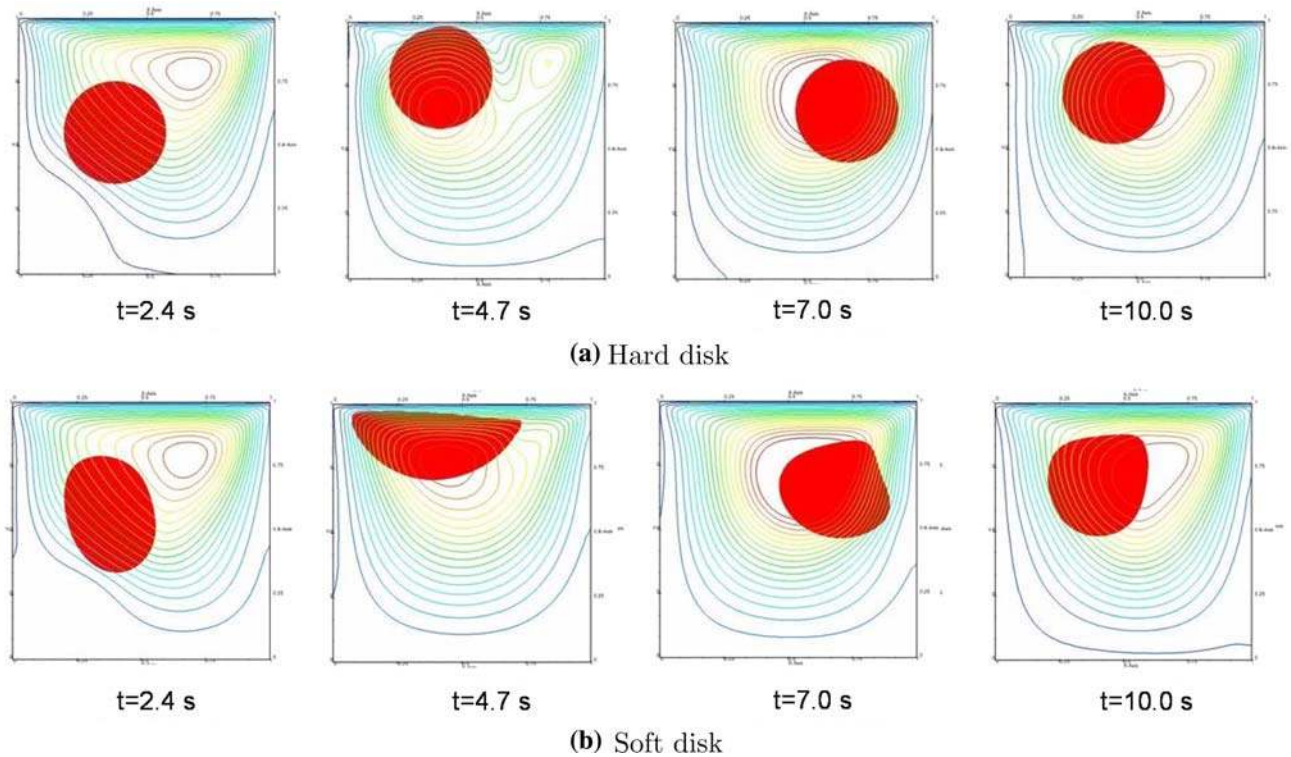

Fig. 10 Deformation of the disk and the streamlines of the fluid in a lid-driven cavity for hard ($C_1 = 5$ dyn/cm^2) and soft ($C_1 = 0.05$ dyn/cm^2) disks

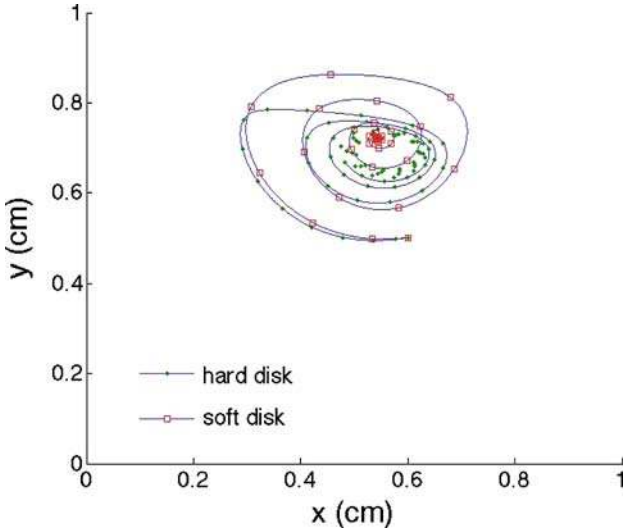

Fig. 11 Trajectories of the soft and hard disks in a lid-driven cavity from $t = 0$ s to $t = 40$ s

construct the function. This is especially useful when the fluid–solid interface comes near a fluid boundary. The location of the solid cannot be predicted a-prior to construct enough layers between the interface and the fluid boundary. With finite element interpolation, this restriction is alleviated. As the solid boundary comes near the top boundary, the distance between the solid boundary and the top boundary of the fluid domain can be slightly larger than one fluid element layer with finite element interpolation. Figure 14 shows the detailed mesh resolution as it reaches the fluid boundary for both soft and hard cases.

4 Conclusions

In this paper, we reviewed the interpolation functions used in the immersed boundary method and the immersed finite element method, i.e. the discretized Dirac delta function and the reproducing kernel function. We then proposed a

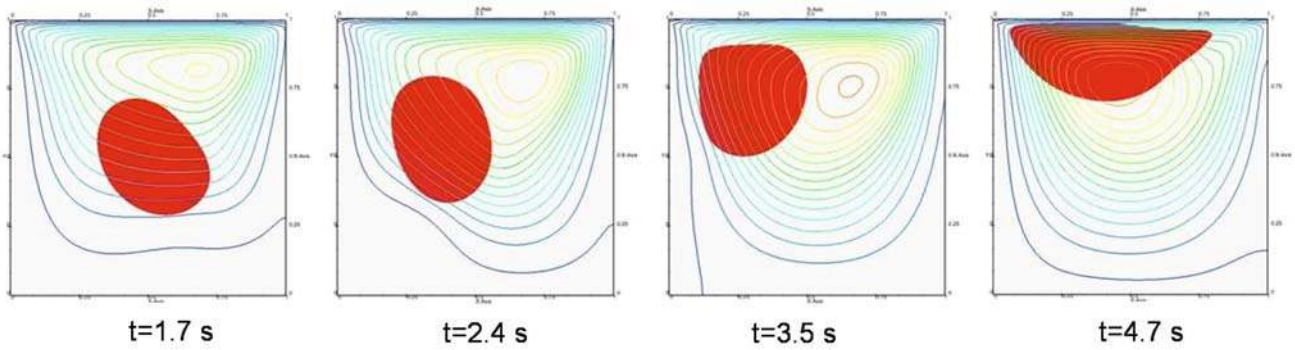

Fig. 12 Soft disk deformation in a lid-driven cavity without volume correction

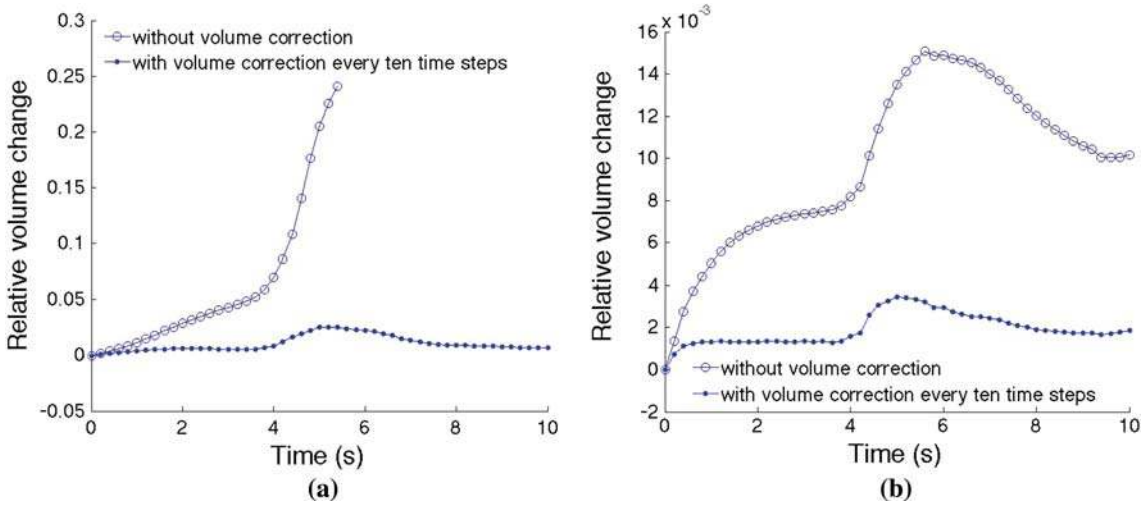

Fig. 13 Relative volume change versus time with and without volume corrections. **a** $C_1 = 0.05$ dyn/cm^2, **b** $C_1 = 5.0$ dyn/cm^2

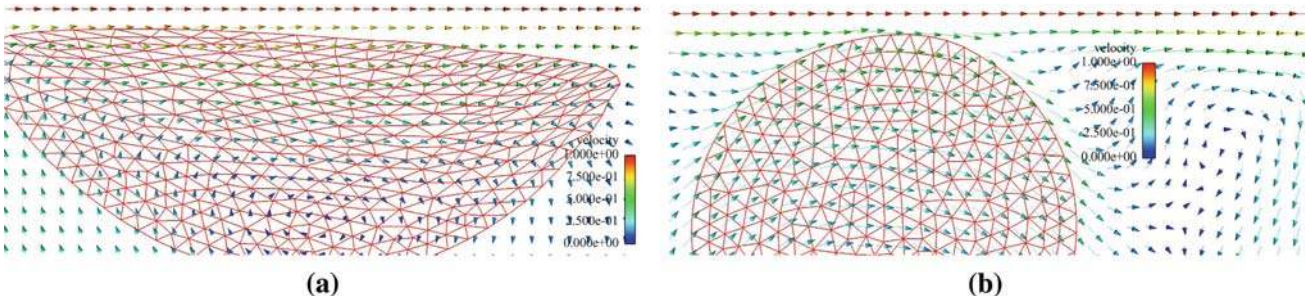

Fig. 14 Velocity fields as the disk comes near the top boundary of the cavity. **a** Soft case at $t = 4.9$ s. **b** Hard case at $t = 5.3$ s

straightforward finite element interpolation function that is capable of producing sharper interface that preserves the accuracy in interface solutions and to be used on unstructured background fluid meshes. The finite element interpolation function naturally satisfies the reproducing condition and it is easy to implement. Comparing to the previously mentioned techniques, the thickness of the interface can be narrowed by approximately 65% when using uniform grids, and can be improved even further when nonuniform or unstructured grids are used.

Through the example problems, we performed a thorough convergence test and examined the mesh size compatibility requirement for the fluid and solid domains. We found a mesh size ratio of 0.5 is required for the fluid and solid discretization to avoid numerical issues. If the fluid mesh size is less than half of the solid mesh size, then a leaking phenomenon would occur and lead the solutions to diverge. This value is consistent for several mesh resolutions. We also observed a relatively large volume change when the solid comes near a moving fluid boundary that generates large velocity gradient. A volume correction algorithm is imposed to enforce this incompressibility constraint. This correction algorithm can dramatically improve the durability of the incompressibility assumption and enhance the performance of the simulation. In summary, this paper introduces a finite element interpolation function to be used in the immersed finite element method and closely examines and resolves several detailed numerical issues that are present in the current non-conforming techniques. It provides a more accurate and a more reliable approach to be used in the simulations of fluid–structure interactions.

References

1. Tezduyar TE, Behr M, Liou J (1992) A new strategy for finite element computations involving moving boundaries and interfaces -The DSD/ST procedure: I. The concept and the preliminary numerical tests. Comput Methods Appl Mech Eng 94:339–351
2. Johnson A, Tezduyar TE (1995) Numerical simulation of fluid-particle interactions. In: Proceedings of the international conference on finite elements in fluids, Venezia, Italy
3. Johnson A, Tezduyar TE (1997) 3D simulations of fluid-particle interactions with the number of particles reaching 100. Comput Methods Appl Mech Eng 145(3–4):301–321
4. Johnson A, Tezduyar TE (1999) Advanced mesh generation and update methods for 3D flow simulations. Comput Mech 23: 130–143
5. Stein K, Benney R, Tezduyar TE, Potvin J (2001) Fluid–structure interactions of a cross parachute: numerical simulation. Comput Methods Appl Mech Eng 191:673–687
6. Hughes TJR, Liu WK, Zimmermann TK (1981) Lagrangian-Eulerian Finite Element formulation for incompressible viscous flows. Comput Methods Appl Mech Eng 29:329–349
7. Liu WK, Ma DC (1982) Computer implementation aspects for fluid–structure interaction problems. Comput Methods Appl Mech Eng 31:129–148
8. Huerta A, Liu WK (1988) Viscous flow with large free surface motion. Comput Methods Appl Mech Eng 69:277–324
9. Liu WK, Chang H, Chen J, Belytschko T (1988) Arbitrary Lagrangian-Eulerian Petrov-Galerkin finite elements for nonlinear continua. Comput Methods Appl Mech Eng 68:259–310
10. Hu HH, Patankar NA, Zhu MY (2001) Direct numerical simulations of fluid–solid systems using the arbitrary Lagrangian-Eulerian technique. J Comput Phys 169:427–462
11. Zhang LT, Wagner G, Liu WK (2003) Modeling and simulation of fluid structure interaction by meshfree and FEM. Commun Numer Methods Eng 19:615–621
12. Fortin M, Glowinski R (1983) Augmented Lagrangian Method: applications to the numerical solution of boundary-value problems. North-Holland, Amsterdam
13. Glowinski R, Pan TW, Hesla TI, Joseph DD (1999) A distributed Lagrange multiplier/fictitious domain method for particulate flows. Int J Multiphase Flow 25:755–794
14. Glowinski R, Pan TW, Hesla TI, Joseph DD, Périaux J (2001) A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: Application to particulate flow. J Comput Phys 169:427–462
15. Wagner J, Moës N, Liu WK, Belytschko T (2001) The extended finite element method for rigid particles in stokes flow. Int J Numer Methods Eng 51(3):293–313
16. Chessa J, Smolinski P, Belytschko T (2002) The extended finite element method (XFEM) for solidification problems. Int J Numer Methods Eng 53:1959–1977
17. Chessa J, Belytschko T (2003) The extended finite element method for two-phase fluids. ASME J Appl Mech 70:10–17
18. Peskin CS (1972) Flow patterns around heart valves: a numerical method. J Comput Phys 10:252–270
19. Peskin CS (2002) The immersed boundary method. Acta Numer 11:479–517
20. Peskin CS (1977) Numerical analysis of blood flow in the heart. J Comput Phys 25:220–252

21. McCracken MF, Peskin CS (1980) A vortex method for blood flow through heart valves. J Comput Phys 35:183–205
22. McQueen DM, Peskin CS (1983) Computer-assisted design of pivoting-disc prosthetic mitral valves. J Comput Phys 86:126–135
23. Peskin CS, McQueen DM (1989) A three-dimensional computational method for blood flow in the heart. I. Immersed elastic fibers in a viscous incompressible fluid. J Comput Phys 81(2):372–405
24. Peskin CS, McQueen DM (1992) Cardiac fluid dynamics. Critical reviews in biomedical engineering. SIAM J Sci Stat Comput 20(6):451–459
25. Peskin CS, McQueen DM (1994) Mechanical equilibrium determines the fractal fiber architecture of aortic heart valve leaflets. Am J Physiol 266(1):H319–H328
26. Peskin CS, McQueen DM (1996) Case studies in mathematical modeling-ecology, physiology, and cell biology. Prentice-Hall, Englewood Cliffs
27. LeVeque RJ, Li Z (1994) The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. SIAM J Numer Anal 31(4):1091–1094
28. LeVeque RJ, Li Z (1997) Immersed interface methods for stokes flow with elastic boundries or surface tension. SIAM J Sci Comput 18(3):709–735
29. Fogelson AL, Keener JP (2000) Immersed interface method for Neumann and related problems in two and three dimensions. SIAM J Sci Comput 22(5):1630–1654
30. Lee L, LeVeque RJ (2003) An immersed interface method for incompressible Navier–Stokes equations. SIAM J Sci Comput 25(3):832–856
31. Li Z, Lai MC (2001) The immersed interface mehtods for the Navier–Stokes equations with singular forces. J Comput Phys 171:822–842
32. Wiegmann A, Bube KP (1998) The immersed interface method for nonlinear differential equations with discontinuous coefficients and singular sources. SIAM J Numer Anal 35(1):177–200
33. Wiegmann A, Bube KP (2000) The explicit-jump immersed interface method: finite difference methods for PDEs with piecewise smooth solutions. SIAM J Numer Anal 37(3):827–862
34. Wang X, Liu WK (2004) Extended immersed boundary method using FEM and RKPM. Comput Methods Appl Mech Eng 193: 1305–1321
35. Boffi D, Gastaldi L (2003) A finite element approach for the immersed boundary method. Comput Struct 81:491–501
36. Boffi D, Gastaldi L, Heltai L (2007) On the CFL condition for the finite element immersed boundary method. Comput Struct 85:775–783
37. Zhang LT, Gerstenberger A, Wang X, Liu WK (2004) Immersed finite element method. Comput Methods Appl Mech Eng 193:2051–2067
38. Liu WK, Liu Y, Zhang LT, Wang X, Gerstenberger A, Farrell D (2004) Immersed finite element method and applications to biological systems. In: Finite element methods: 1970's and beyond. International Center for Numerical Methods and Engineering
39. Liu Y, Liu WK (2006) Rheology of red blood cell aggregation in capillary by computer simulation. J Comput Phys 220:139–154
40. Gay M, Zhang LT, Liu WK (2006) Stent modeling using immersed finite element method. Comput Methods Appl Mech Eng 195: 4358–4370
41. Liu WK, Liu Y, Farrell D, Zhang LT, Wang S, Fukui Y, Patankar N, Zhang Y, Bajaj C, Lee J, Hong J, Chen X, Hsu H (2006) Immersed finite element method and its applications to biological systems. Comput Methods Appl Mech Eng 195:1722–1749
42. Zhang LT, Gay M (2007) Immersed finite element method for fluid-structure interactions. J Fluids Struct 23:839–857
43. Liu WK, Jun S, Zhang YF (1995) Reproducing kernel particle methods. Int J Numer Methods Fluids 20:1081–1106
44. Liu WK, Chen YJ (1995) Wavelet and multiple scale reproducing kernel method. Int J Numer Methods Fluids 21:901–932
45. Zhang LT, Wagner GJ, Liu WK (2002) A parallized meshfree method with boundary enrichment for large-scale CFD. J Comput Phys 176:483–506
46. Liu WK, Chen Y, Chang CT, Belytschko T (1996) Advances in multiple scale kernel particle methods. Comput Mech 18(2):73–111
47. Li S, Liu WK (1999) Reproducing kernel hierarchical partition of unity. part I: formulation and theory. Comput Methods Appl Mech Eng 145:251–288
48. Liu WK, Chen Y, Uras RA, Chang CT (1996) Generalized multiple scale reproducing kernel particle methods. Comput Methods Appl Mech Eng 139:91–158
49. Bradley CJ (2007) The algebra of geometry: cartesian, areal and projective coordinates. Highperception Ltd, Bath, UK
50. Dunne T (2006) An Eulerian approach to fluid–structure interaction and goal-oriented mesh adaptation. Int J Numer Methods Fluids 51:1017–1039
51. Zhao H, Freund BJ, Moser DB (2008) A fixed-mesh method for incompressible flow-structure systems with finite solid deformations. J Comput Phys 227:3114–3140
52. Belytschko T, Liu WK, Moran B (2000) Nonlinear finite elements for continua and structures. Wiley, New York
53. Sugiyama K, Takeuchi S, Ii S, Takagi S, Matsumoto Y (2008) Full Eulerian finite difference computation for fluid–structure coupling problem. In: The 61st annual APS division of fluid dynamics, San Antonio