Penalty Immersed Boundary Method for an Elastic Boundary with Mass

Yongsam Kim * Charles S. Peskin

Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012 USA.

Preprint submitted to Elsevier Science *26 August 2005*

Abstract

The immersed boundary (IB) method has been widely applied to problems involving a moving elastic boundary that is immersed in fluid and interacting with it. But most of the previous applications of the IB method have involved a massless elastic boundary and used efficient Fourier transform methods for the numerical solutions. Extending the method to cover the case of a massive boundary has required spreading the boundary mass out onto the fluid grid and then solving the Navier-Stokes equations with a variable mass density. The variable mass density of this previous approach makes Fourier transform methods inapplicable, and requires a multigrid solver. Here we propose a new and simple way to give mass to the elastic boundary and show that the new method can be applied to many problems for which the boundary mass is important. The new method does not spread mass to the fluid grid, retains the use of Fourier transform methodology, and is easy to implement in the context of an existing IB method code for the massless case.

Key words: immersed boundary method, penalty method, massive elastic boundary, filament, windsock, flag, inflow velocity, FFT

1991 MSC: 65-04, 65M06, 76D05, 76M20

* Corresponding author. Present address: ICES, University of Texas at Austin, Austin, TX 78712, USA.

Email addresses: kimy@ices.utexas.edu (Yongsam Kim),

peskin@cims.nyu.edu (Charles S. Peskin).

1 Introduction

The purpose of this paper is to propose a simple way to extend the immersed boundary (IB) method to a situation in which an elastic moving boundary has mass that cannot be neglected and to show several examples of the application of this new methodology. In the examples chosen, the boundary mass plays a crucial dynamical role. This is in contrast to most previous applications of the IB method, in which the immersed boundary was either massless or neutrally buoyant in the ambient fluid.

The IB method was developed to study flow patterns around heart valves, and is a generally useful method for problems in which elastic materials interact with a viscous incompressible fluid. In the IB formulation, the influence of the elastic material immersed in the fluid appears as a localized body force acting on the fluid. This body force arises from the elastic stresses of the material. Moreover, the immersed material is required to move at the local fluid velocity as a consequence of the no-slip condition. The central idea of the IB method is that the Navier-Stokes solver does not need to know anything about the complicated time-dependent geometry of the elastic boundary, and that therefore we can escape from the difficulties caused by the interaction between the elastic boundary and the fluid flow. This whole approach has been applied successfully to problems of blood flow in the heart [18,20–23,25], wave propagation in the cochlea [3,6], platelet aggregation during blood clotting [5], and several other problems [2,8,9,11,13].

In these applications, however, the elastic boundaries have no mass. (In the case of a thick "boundary" like the muscular heart wall, the corresponding

assumption is that the boundary is neutrally buoyant, so that the mass of the volume occupied by the boundary is the same as if it were occupied by fluid.) The massless assumption appears in one version of the mathematical derivation of the IB method [19,21,23], see however [7,28], and can be applied when the mass of the moving boundary is too small to have a significant effect on the overall motions of the fluid and the elastic material. With the massless assumption, however, we cannot approach many other problems for which the boundary mass is important. Consider, for example, the simulation of a flapping filament in a flowing soap film as studied by Zhu and Peskin [27,28], who have shown that a massless filament does not flap at all, and that filament mass is essential for the sustained flapping that is seen not only in simulations with mass but also in laboratory experiments [26]. Subsequent experiments (J.Zhang, personal communication) have shown as predicted that an extremely fine filament fails to flap under the same conditions in which flapping of a more massive filament had been seen. Many other examples, in which the massless assumption is not reasonable arise in aerodynamics. Since air is such a light fluid, it is usually the case that elastic boundaries immersed in air have mass that cannot be neglected. Not only the inertial effect of the boundary mass but also the effect of gravity on the boundary mass are important. We study several such examples in this paper.

In [7,28], the method used to handle boundary mass was to spread that mass out onto the fluid grid, in much the same manner as boundary force is conventionally applied to the fluid grid in the IB computations. When this is done, a variable density $\rho(\mathbf{x}, t)$ appears in the Navier-Stokes equations, and this complicates the numerical solver of those equations. Specifically, it makes Fourier methods inapplicable and requires that an iterative method like Multigrid be

used instead. This introduces the question of what stopping criterion to use for the iterative solver, i.e., how to optimize the tradeoff between accurate solution of the difference equations at each time step and the computational cost of doing more iterations per time step, a question which does not arise when a direct solution method is available.

Our current approach stays much closer to the IB method for the massless case. In particular, it does not spread mass to the fluid and so it requires the same constant-density Navier-Stokes solver as is used in the massless case. The key idea is to introduce a massive boundary point as a twin of each immersed boundary marker where mass is needed. We assume that the massive boundary thus introduced does not interact directly with the fluid and that the original immersed boundary, which will play the same role as in the original IB method, has no mass. The two boundaries are supposed to be the same, and when they move apart, a strong restoring force arises to pull them back together. This is done with a collection of stiff springs connecting the two boundaries. The massive boundary moves according to Newton's law ($\mathbf{F} = m\mathbf{a}$) in which the only forces acting are the forces of the stiff springs and the gravitational force. Meanwhile, at the other end of each spring, the massless boundary marker is moving at the local fluid velocity and spreading force locally to the fluid grid. Among the forces that it spreads is the force on the boundary marker due to the stiff spring (equal and opposite to the force acting on the massive boundary). Because the spring is stiff, the massive boundary point stays close to its twin immersed boundary marker, and the overall effect is that the boundary has mass.

Even though the two boundaries (which we call massive and massless boundaries) are supposed to coincide, we allow them to separate by an amount that

depends inversely on the stiffness of the spring connecting them. The stiffness parameter of this spring is analogous to a penalty parameter in a constrained optimization problem, which is why we call this new mass-handling idea the penalty immersed boundary (pIB) method. We shall see later that, in the limit of infinite spring stiffness, the two boundaries coincide, and the results obtained by the pIB method approach those of the version of the IB method used by Zhu and Peskin [27,28] for the massive case.

2 Equations of Motion

We begin by stating the mathematical formulation of the equations of the motion for a system comprised of a three-dimensional viscous incompressible fluid containing an immersed, elastic boundary with mass [19].

$$\rho(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u}) = -\nabla p + \mu \Delta \mathbf{u} + \mathbf{f}, \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2}$$

$$\rho(\mathbf{x}, t) = \rho_0 + \int M(r, s)\delta(\mathbf{x} - \mathbf{X}(r, s, t))drds, \tag{3}$$

$$\mathbf{f}(\mathbf{x}, t) = \int \mathbf{F}(r, s, t)\delta(\mathbf{x} - \mathbf{X}(r, s, t))drds, \tag{4}$$

$$\begin{aligned} \frac{\partial \mathbf{X}}{\partial t}(r, s, t) &= \mathbf{u}(\mathbf{X}(r, s, t), t) \\ &= \int \mathbf{u}(\mathbf{x}, t)\delta(\mathbf{x} - \mathbf{X}(r, s, t))d\mathbf{x}, \end{aligned} \tag{5}$$

$$\mathbf{F}(r, s, t) = -\frac{\partial E}{\partial \mathbf{X}}(\mathbf{X}(r, s, t), t). \tag{6}$$

Eqs. (1) and (2) are the familiar Navier-Stokes equations for a viscous incompressible fluid subject to an applied body force per unit volume $\mathbf{f}(\mathbf{x}, t)$.

The density $\rho(\mathbf{x}, t)$ is the non-uniform mass density of the whole system (fluid+boundary) and μ is the constant fluid viscosity. The unknown functions in the fluid equations are the fluid velocity, $\mathbf{u}(\mathbf{x}, t)$; the fluid pressure, $p(\mathbf{x}, t)$; and the force per unit volume applied by the immersed boundary to the fluid, $\mathbf{f}(\mathbf{x}, t)$, where $\mathbf{x} = (x, y, z)$ are fixed Cartesian coordinates, and t is the time.

Eq. (6) is the immersed boundary equation which is written in Lagrangian form. The parameters r, s label a fixed material point. The unknown $\mathbf{X}(r, s, t)$ completely describes the motion of the immersed boundary, and also its spatial configuration at any given time. Note that $\mathbf{X}(r, s, t)$ represents a 2-dimensional surface in 3-dimensional space, which is the case in all the 3-D applications in this paper. $\mathbf{F} = \mathbf{F}(r, s, t)$ is the force density applied by the immersed boundary to the fluid, in the sense that $\mathbf{F}(r, s, t)drds$ is the force applied to the fluid by the patch of immersed boundary $drds$. The elastic contribution to this force density is given by the variational derivative $-\frac{\partial E}{\partial \mathbf{X}}$ of the elastic energy functional $E[\mathbf{X}(\cdot, \cdot, t)]$. This variational derivative is implicitly defined by

$$dE(t) = \int \frac{\partial E}{\partial \mathbf{X}}(r, s, t) \cdot d\mathbf{X}(r, s, t) drds, \tag{7}$$

where $d\mathbf{X}$ is a perturbation of the boundary configuration and dE is the resulting perturbation in the elastic energy of the boundary (to first order).

In this paper, the energy functional $E[\mathbf{X}(\cdot, \cdot, t)]$ is chosen a simple way. In a 2-D space, we consider an immersed boundary as a 1-D rod and use two elasticities: one that resists stretching and compression, and another that resists bending:

$$E[\mathbf{X}(\cdot)] = \frac{1}{2} c_s \int (|\frac{\partial \mathbf{X}}{\partial s}| - 1)^2 ds + \frac{1}{2} c_b \int |\frac{\partial^2 \mathbf{X}}{\partial^2 s}|^2 ds, \tag{8}$$

where c_s and c_b are constants. Regarding a 2-D boundary immersed in a 3-D space as a shell or membrane, one could use quite general strain-stress relations [1,9]. In all our 3-D simulations, however, we use a very simple law to generate the elastic force. The 2-D immersed boundaries are modeled as collections of 1-D rods. Suppose that $\mathbf{X}(r,s)$ is an immersed boundary. Then, for a fixed r or s, $\mathbf{X}(r,s)$ is a curve which is parameterized as a function of s or r, respectively. Now each such curve can be assigned the same strain-stress relation as is used for the 1-D immersed boundary in our 2-D simulation, and all the forces generated from all these rods are summed up to make an elastic force $\mathbf{F}(r,s)$.

Eqs. (3), (4) and (5), which we call interaction equations, involve the three-dimensional Dirac delta function $\delta(\mathbf{x}) = \delta(x)\delta(y)\delta(z)$, which expresses the local character of the interaction. ρ_0 is the constant fluid density and $M(r,s)$ is the mass density of the immersed boundary in the sense that $M(r,s)drds$ is the mass of the patch $drds$. In (3) and (4), the delta function $\delta(\mathbf{x}) = \delta(x)\delta(y)\delta(z)$ is still three-dimensional, but there are only two integrals $drds$. As a result, $\rho(\mathbf{x},t)$ and $\mathbf{f}(\mathbf{x},t)$ are each singular like a one-dimensional delta function with the singularity supported on the immersed boundary. Although $\rho(\mathbf{x},t)$ and $\mathbf{f}(\mathbf{x},t)$ are infinite on the immersed boundary, their integrals over any finite volume are finite. In the case of $\rho(\mathbf{x},t)$, for example, we may integrate over an arbitrary region Ω to find

$$\begin{aligned}\int_\Omega \rho(\mathbf{x},t)d\mathbf{x} &= \rho_0 \mathrm{Vol}(\Omega) + \int_\Omega \int M(r,s)\delta(\mathbf{x}-\mathbf{X}(r,s,t))drdsd\mathbf{x} \\ &= \rho_0 \mathrm{Vol}(\Omega) + \int_{\mathbf{X}(r,s)\in\Omega} M(r,s)drds, \end{aligned} \tag{9}$$

which represents the sum of the mass of the immersed material and the fluid that are contained within a fixed space region Ω. Similarly, the integral of $\mathbf{f}(\mathbf{x}, t)$ over such a volume is the total force applied to the fluid by the part of the immersed material living in the volume.

Eq. (5) is the equation of motion of the immersed elastic boundary. It is just the no-slip condition which says that the boundary moves at the local fluid velocity. This is rewritten in terms of the Dirac-delta function in the second form of Eq. (5). We do so in order to expose a certain symmetry with Eq. (4), in which the force generated by the immersed boundary is re-expressed as body force acting on the fluid. This symmetry is important in the construction of our numerical scheme. Note, however, that the integral in Eq. (5) is a triple integral($dxdydz$), unlike the integral in Eqs. (3) and (4). Thus $\partial\mathbf{X}/\partial t$ is finite, unlike ρ and $\mathbf{f}$.

Note that Zhu and Peskin [27,28] use this system of equations (1)-(6) and solve it directly. Since the fluid density $\rho(\mathbf{x}, t)$ is not a constant, which disables the use of FFT methodology, their numerical solver of the Navier-Stokes equations needs a multigrid method.

Now substitute the density defined in Eq. (3) into Eq. (1), and separate the left-hand side of (1) into two terms: one involving the constant density ρ_0, and the other containing the singular part of $\rho(\mathbf{x}, t)$ that comes from the immersed boundary. Then

$$\rho_0(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u}) = -\nabla p + \mu \Delta \mathbf{u} + \mathbf{f} - \mathbf{f}_D, \tag{10}$$

$$\mathbf{f}_D(\mathbf{x}, t) = \int \frac{D\mathbf{u}(\mathbf{x}, t)}{Dt} M(r, s)\delta(\mathbf{x} - \mathbf{X}(r, s, t))drds, \tag{11}$$

where $\frac{D}{Dt}$ is the material derivative. Using the relation:

$$\frac{\partial^2 \mathbf{X}}{\partial t^2}(r,s,t) = \frac{D\mathbf{u}}{Dt}(\mathbf{X}(r,s,t),t), \tag{12}$$

we can interpret $\mathbf{f}_D(\mathbf{x},t)$ in Eq. (11) as an Eulerian body force obtained by using the Dirac delta function to transform the Lagrangian expression

$$\mathbf{F}_D(r,s,t) = M(r,s)\frac{\partial^2 \mathbf{X}}{\partial t^2}(r,s,t), \tag{13}$$

which is known as the D'Alembert force. In a numerical procedure, one might try to evaluate $\mathbf{F}_D(r,s,t)$ by applying a backward time difference to $\partial\mathbf{X}/\partial t$ as evaluated by the Eq. (5). Then $\mathbf{F}_D$ could be applied to the fluid grid like any other immersed boundary force by using the Dirac delta function. This method was proposed by Xiaodong Wang [14] and works when the boundary mass is sufficiently small. Here we use a different way of approximating $\mathbf{F}_D$, as described in the following.

Now we modify the equations discussed so far to make what we call the penalty immersed boundary (pIB) method [12]. To do this, we introduce two boundaries immersed in a 3-D fluid: one is a massive boundary $\mathbf{Y}(r,s,t)$ having all the mass of the elastic immersed boundary, and the other is a massless boundary $\mathbf{X}(r,s,t)$. The massive boundary $\mathbf{Y}(r,s,t)$ has mass density $M(r,s)$ but does not interact with the surrounding fluid directly. On the other hand, the massless boundary $\mathbf{X}(r,s,t)$ interacts with the fluid: it moves with the local fluid velocity and exerts the elastic force locally on the fluid. Both the massless and massive boundaries are supposed to represent the same material surface. If a pair of corresponding boundary points moves apart, a restoring force appears in order to make them move close together (see Fig. 1). The

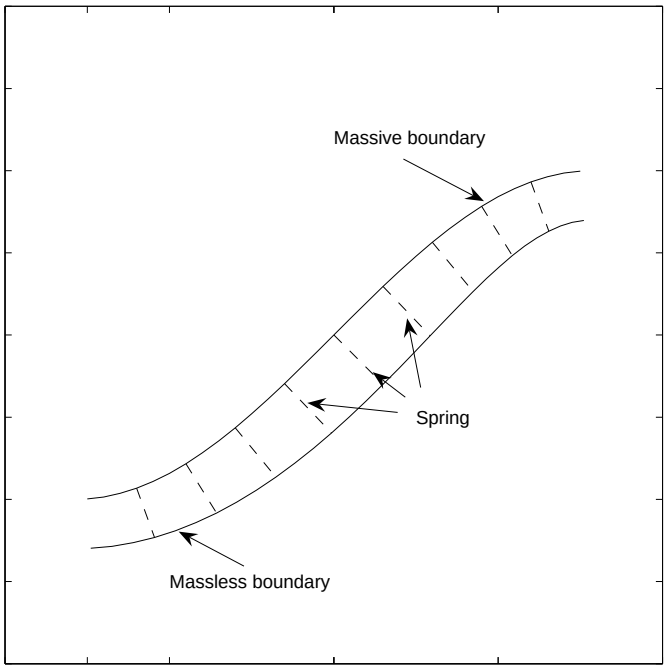

Fig. 1. Massive and massless boundaries are linked together by a collection of a very stiff springs with rest length zero.

restoring force $\mathbf{F}_K$ acting on the massless boundary is

$$\mathbf{F}_K(r,s,t) = K(\mathbf{Y}(r,s,t) - \mathbf{X}(r,s,t)), \tag{14}$$

where K is a large constant.

Now the massive boundary $\mathbf{Y}(r,s,t)$ moves according to the following equation:

$$M(r,s)\frac{\partial^2 \mathbf{Y}(r,s,t)}{\partial t^2} = -\mathbf{F}_K(r,s,t) \tag{15}$$

Note that the only force acting on the massive boundary is the reaction force $-\mathbf{F}_K$ (we shall add to this a gravitational force later), and that the massive boundary does not interact with the fluid directly. On the other end of the spring, the massless boundary $\mathbf{X}(r,s,t)$ applies to the nearby fluid both its

elastic force $\mathbf{F}_E = -\frac{\partial E}{\partial \mathbf{X}}$ and the spring force $\mathbf{F}_K$.

Consider the case in which K in (14) goes to infinity. Then the massive boundary $\mathbf{Y}(r,s,t)$ approaches the massless boundary $\mathbf{X}(r,s,t)$, in which case, using $\mathbf{X}(r,s,t)$ only, we can see that $\mathbf{F}_K(r,s,t)$ is same as $-\mathbf{F}_D(r,s,t)$ of Eq. (13). Note that, in practice, K cannot be infinite but we can keep $\mathbf{X}(r,s,t)$ and $\mathbf{Y}(r,s,t)$ as close as we like by choosing K sufficiently large.

With a large constant K (not infinity), we can use Eqs. (14) and (15) instead of Eq (13) (D'Alembert force) and derive the equations of motion for the pIB method as follows:

$$\rho_0(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}) = -\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{f}, \tag{16}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{17}$$

$$\mathbf{f}(\mathbf{x},t) = \int \mathbf{F}(r,s,t)\delta(\mathbf{x}-\mathbf{X}(r,s,t))drds, \tag{18}$$

$$\begin{aligned}\frac{\partial \mathbf{X}}{\partial t}(r,s,t) &= \mathbf{u}(\mathbf{X}(r,s,t),t) \\ &= \int \mathbf{u}(\mathbf{x},t)\delta(\mathbf{x}-\mathbf{X}(r,s,t))d\mathbf{x},\end{aligned} \tag{19}$$

$$\mathbf{F} = \mathbf{F}_E + \mathbf{F}_K, \tag{20}$$

$$\mathbf{F}_E = -\frac{\partial E}{\partial \mathbf{X}}, \tag{21}$$

$$\mathbf{F}_K(r,s,t) = K(\mathbf{Y}(r,s,t) - \mathbf{X}(r,s,t)), \tag{22}$$

$$M(r,s)\frac{\partial^2 \mathbf{Y}}{\partial t^2} = -\mathbf{F}_K(r,s,t) - M(r,s)g\mathbf{e}_3, \tag{23}$$

where g is the gravitational acceleration and $\mathbf{e}_3$ is a unit vector in the vertical direction (against gravity). We have here added a gravitational term on the

right-hand side of Eq. (23) in order to allow for the influence of gravity on the massive immersed boundary. (The corresponding term $-\rho_0 g\mathbf{e}_3$ is not included in Eq. (16) because there it only amounts to a redefinition of the pressure.)

3 Numerical Implementation of the pIB method

We now describe a formally second-order IB method to solve the equations of motion [15,20]. The word 'formally' is used as a reminder that this scheme is only second-order accurate for problems with smooth solutions. Even though our solutions are not smooth (the velocity has jumps in derivative across the immersed boundary), the use of the formally second-order method results in improved accuracy, see [15].

The specific formally second-order method that we use is the one described in [20], generalized here to take into account the massive boundary that is linked to the immersed elastic boundary by stiff springs. In this method, each time step proceeds in two substeps, which are called the preliminary and final substeps. In the preliminary substep, we get data at time level $n + \frac{1}{2}$ from data at time level n by a first-order accurate method. Then the final substep starts again at time level n and proceeds to time level $n + 1$ by a second-order accurate method. This Runge-Kutta framework allows the second-order accuracy of the final substep to be the overall accuracy of the scheme.

We use a superscript to denote the time level. Thus $\mathbf{X}^n(r, s)$ is shorthand for $\mathbf{X}(r, s, n\Delta t)$, where Δt is the duration of the time step, and similarly for all other variables. Our goal is to compute updated $\mathbf{u}^{n+1}$, $\mathbf{V}^{n+1}$, where $\mathbf{V}{=}d\mathbf{Y}/dt$, $\mathbf{X}^{n+1}$, and $\mathbf{Y}^{n+1}$ from given data $\mathbf{u}^n$, $\mathbf{V}^n$, $\mathbf{X}^n$ and $\mathbf{Y}^n$.

Before describing how this is done, we have to say a few words about the spatial discretization. There are two such discretizations: one for the fluid and the other for the elastic boundary.

The grid on which the fluid variables are defined is a fixed uniform cubic lattice of meshwidth $h = \Delta x_1 = \Delta x_2 = \Delta x_3$. Now we introduce the central difference operator D_i, defined for $i = 1, 2, 3$, as follows:

$$(D_i\phi)(\mathbf{x}) = \frac{\phi(\mathbf{x} + h\mathbf{e}^i) - \phi(\mathbf{x} - h\mathbf{e}^i)}{2h}, \tag{24}$$

where $\mathbf{e}^i$ is the unit vector in the i-th coordinate direction. As the notation suggests, the difference operator in i-th direction D_i corresponds to the i-th component of the differential operator ∇. Thus $\mathbf{D}p$ will be the discrete gradient of p, and $\mathbf{D} \cdot \mathbf{u}$ will be the discrete divergence of $\mathbf{u}$.

We shall also make use the central difference Laplacian L:

$$(L\phi)(\mathbf{x}) = \sum_{i=1}^{3} \frac{\phi(\mathbf{x} + h\mathbf{e}^i) + \phi(\mathbf{x} - h\mathbf{e}^i) - 2\phi(\mathbf{x})}{h^2} \tag{25}$$

The fluid mesh and the elastic boundary mesh defined below are connected by a smoothed approximation to the Dirac delta function. It is denoted δ_h and is of the following form:

$$\delta_h(\mathbf{x}) = h^{-3}\phi(\frac{x_1}{h})\phi(\frac{x_2}{h})\phi(\frac{x_3}{h}), \tag{26}$$

where $\mathbf{x} = (x_1, x_2, x_3)$, and the function ϕ is given by

$$\phi(r) = \begin{cases} \frac{3-2|r|+\sqrt{1+4|r|-4r^2}}{8} & , \quad \text{if} \quad |r|<1 \\ \frac{5-2|r|-\sqrt{-7+12|r|-4r^2}}{8} & , \quad \text{if} \quad 1\leq|r|<2 \\ 0 & , \quad \text{if} \quad 2\leq|r| \end{cases} \tag{27}$$

The motivation and derivation for this particular choice is discussed in [21,23].

We are now ready to describe a typical timestep of the numerical scheme. The preliminary substep which goes from time level n to $n+\frac{1}{2}$ proceeds as follows:

First, update the position of the massless boundary $\mathbf{X}^{n+\frac{1}{2}}(r,s)$.

$$\frac{\mathbf{X}^{n+\frac{1}{2}} - \mathbf{X}^n}{\Delta t/2} = \sum_{\mathbf{x}} \mathbf{u}^n(\mathbf{x})\delta_h(\mathbf{x} - \mathbf{X}^n(r,s))h^3 \tag{28}$$

Here and throughout the paper $\sum_{\mathbf{x}}$ denotes the sum over the cubic lattice in physical space on which the fluid variables are defined. Similarly, $\sum_{r,s}$ will denote the sum over rectangular lattice in (r,s) space on which the two boundary positions $\mathbf{X}$, $\mathbf{Y}$, the force density $\mathbf{F}$ and the mass density M are defined.

The key to generalizing the formally second-order method to the massive case is to handle the massive boundary in a manner that closely parallels the numerical treatment of the immersed boundary itself. Thus, we update $\mathbf{Y}^{n+\frac{1}{2}}$ in a similar manner to $\mathbf{X}^{n+\frac{1}{2}}$:

$$\frac{\mathbf{Y}^{n+\frac{1}{2}} - \mathbf{Y}^n}{\Delta t/2} = \mathbf{V}^n, \tag{29}$$

where $\mathbf{V}^n$ is the velocity vector of the massive boundary (the known value at time $n\Delta t$, like $\mathbf{u}^n$).

Next, calculate the force density $\mathbf{F}^{n+\frac{1}{2}}$ which is the sum of two parts: one is elastic force and the other is from the spring linked between massless and massive boundaries.

$$\mathbf{F}_K^{n+\frac{1}{2}} = K(\mathbf{Y}^{n+\frac{1}{2}} - \mathbf{X}^{n+\frac{1}{2}}), \tag{30}$$

$$\mathbf{F}^{n+\frac{1}{2}} = -\frac{\partial E}{\partial \mathbf{X}}(\mathbf{X}^{n+\frac{1}{2}}) + \mathbf{F}_K^{n+\frac{1}{2}}. \tag{31}$$

These are discretization of Eqs. (20)-(22), respectively.

Now we have to change this elastic force density defined on Lagrangian grid points into the force at Eulerian spatial grid points to be applied in the Navier-Stokes equations. This is done by a discretization of Eq. (18).

$$\mathbf{f}^{n+\frac{1}{2}}(\mathbf{x}) = \sum_{r,s} \mathbf{F}^{n+\frac{1}{2}}(r,s)\delta_h(\mathbf{x} - \mathbf{X}^{n+\frac{1}{2}}(r,s))\Delta s \Delta r. \tag{32}$$

With $\mathbf{f}^{n+\frac{1}{2}}$ in hand, we can turn to solving the Navier-Stokes equations:

$$\rho(\frac{u_i^{n+\frac{1}{2}} - u_i^n}{\Delta t/2} + \frac{1}{2}(\mathbf{u}\cdot\mathbf{D}u_i + \mathbf{D}\cdot(\mathbf{u}u_i))^n) + D_i\tilde{p}^{n+\frac{1}{2}} = \mu L u_i^{n+\frac{1}{2}} + f_i^{n+\frac{1}{2}}, \tag{33}$$

for $i = 1, 2, 3$, and

$$\mathbf{D}\cdot\mathbf{u}^{n+\frac{1}{2}} = 0. \tag{34}$$

The notation $\tilde{p}^{n+\frac{1}{2}}$ is used to distinguish this pressure from the one that is computed when solving Eqs (38)-(39), below. Note that the unknowns in Eqs (33)-(34) are $u_i^{n+\frac{1}{2}}$ and $\tilde{p}^{n+\frac{1}{2}}$ and that they enter into these equations linearly. Since all the coefficients of these equations are constants, the system of Eqs (33)-(34) can be solved by Fast Fourier Transform with the periodic boundary condition [21,23].

Again we have to calculate the velocity $\mathbf{V}^{n+\frac{1}{2}}$ of the massive boundary in the same fashion.

$$M(\frac{\mathbf{V}^{n+\frac{1}{2}}-\mathbf{V}^{n}}{\Delta t/2}) = -\mathbf{F}_{K}^{n+\frac{1}{2}} - Mg\mathbf{e}_3. \tag{35}$$

This is nothing but Euler's method for Eq (23) and completes the preliminary substep.

The final substep is the use of $\mathbf{u}^{n+\frac{1}{2}}$, $\mathbf{V}^{n+\frac{1}{2}}$, $\mathbf{X}^{n+\frac{1}{2}}$ and $\mathbf{Y}^{n+\frac{1}{2}}$ obtained in the preliminary substep to compute the corresponding quantities at time level n+1 by the second-order accurate midpoint rule.

First, using the fluid velocity $\mathbf{u}^{n+\frac{1}{2}}$ and massive boundary velocity $\mathbf{V}^{n+\frac{1}{2}}$, we can find the massless boundary configuration $\mathbf{X}^{n+1}$ and massive boundary position $\mathbf{Y}^{n+1}$.

$$\frac{\mathbf{X}^{n+1}-\mathbf{X}^{n}}{\Delta t} = \sum_{\mathbf{x}} \mathbf{u}^{n+\frac{1}{2}}(\mathbf{x})\delta_h(\mathbf{x}-\mathbf{X}^{n+\frac{1}{2}}(r,s))h^3, \tag{36}$$

$$\frac{\mathbf{Y}^{n+1}-\mathbf{Y}^{n}}{\Delta t} = \mathbf{V}^{n+\frac{1}{2}}. \tag{37}$$

The last thing that we have to do is to update the fluid velocity data:

$$\rho(\frac{u_i^{n+1}-u_i^{n}}{\Delta t} + \frac{1}{2}(\mathbf{u}\cdot\mathbf{D}u_i + \mathbf{D}\cdot(\mathbf{u}u_i))^{n+\frac{1}{2}}) + D_i p^{n+\frac{1}{2}} = \frac{1}{2}\mu L(u_i^{n+1}+u_i^{n}) + f_i^{n+\frac{1}{2}}, \tag{38}$$

for $i = 1, 2, 3$, and

$$\mathbf{D}\cdot\mathbf{u}^{n+1} = 0. \tag{39}$$

Just as in the case of Eqs. (33)-(34), Eqs (38)-(39) are a constant-coefficient linear system in the unknowns u_i^{n+1}, $p^{n+\frac{1}{2}}$. This linear system is solved by fast

Fourier transform. For the velocity of the massive boundary, we have

$$M(\frac{\mathbf{V}^{n+1}-\mathbf{V}^{n}}{\Delta t}) = -\mathbf{F}_{K}^{n+\frac{1}{2}} - Mg\mathbf{e}_3. \tag{40}$$

Since we have now computed $\mathbf{u}^{n+1}$, $\mathbf{V}^{n+1}$, $\mathbf{X}^{n+1}$ and $\mathbf{Y}^{n+1}$, the timestep is complete.

To complete the description of the numerical pIB method, we need to explain the boundary conditions imposed along the edges of our computational domain. In all cases, we use periodic boundary conditions which are very convenient for the solution of the linear systems of Eqs. (33)-(34) and Eqs. (38)-(39) by fast Fourier transform, but we also break the periodicity in various ways depending on the application.

In all of the applications considered here, we need to impose an "inflow velocity". This is done on two (or four) adjacent parallel grid planes (or grid lines in the 2D case). It is very important to use two adjacent planes rather just one, in order to avoid the spatial oscillations that would otherwise propagate throughout the domain via the central-difference structure of our numerical scheme. Although we typically choose either the first or last two grid planes in some coordinate direction, this is of no fundamental significance, since the underlying domain is periodic and all grid points are created equal.

The way of driving the "inflow velocity" in this paper is to apply an external force per unit volume equal to

$$\mathbf{f}_0(\mathbf{x},t) = \begin{cases} \alpha_0(\mathbf{u}_0(t) - \mathbf{u}(\mathbf{x},t)) & , \quad \mathbf{x} \in \Omega_0 \\ \\ 0 & , \quad \text{otherwise,} \end{cases} \tag{41}$$

where Ω_0 is the set of grid points comprised of the two grid planes on which we want to control the velocity, $\mathbf{u}_0(t)$ is the desired velocity on those planes, and α_0 is a constant. When α_0 is large, the grid velocity is driven rapidly towards $\mathbf{u}_0(t)$ within Ω_0. Note that, when we refine the grid by a factor of 2, since the region Ω_0 is fixed, the number of grid planes in Ω_0 becomes twice as large.

Another way to break the periodicity is to put no-slip walls on faces of the domain. The method that we shall describe for this purpose can indeed be used to put no-slip walls of any shape anywhere within the domain; planar walls along the faces are just a special case.

We create fixed no-slip boundaries by laying out an array of "target points" to mark their desired positions. To avoid leaks, the target points should be spaced about half a meshwidth apart (or closer). These target points are fixed in position and do not interact with fluid. Each of them is connected, however, by a stiff spring to a massless immersed boundary point that moves at the local fluid velocity and applies the force generated by the stiff spring locally to the fluid. This provides a feedback mechanism for computing the boundary force needed to enforce the no-slip condition.

Note the close analogy between the above construction and the pIB Method, the massive boundary of which is essentially a target position that moves according to Newton's law of motion instead of being fixed in space.

The various methods that we have discussed above for breaking the periodicity of the domain are special cases which hint at the following general remark: In the context of an immersed boundary method, it is not much of a restriction to use periodic boundary conditions for the domain occupied by the fluid. This is because one can immerse within that domain whatever dynamic or static

geometry the application demands. As a general setting for immersed boundary computations, the periodic domain has both philosophical and practical advantages in comparison to other choices one might make. The philosophical advantage is that the periodic domain preserves the translation invariance of free space. Related to this is the practical advantage that we can use the fast Fourier transform to solve the discretized fluid equations at each time step. When walls are wanted in immersed boundary computations, we put them in as immersed boundaries, and this means that a geometrically complicated wall is no harder to model than a planar one. But the boundaries of the fluid domain do not introduce unwanted wall effects, for the simple reason that a periodic domain has, indeed, no boundaries at all.

4 Numerical convergence test of the pIB method

In this section, we verify the claim we made in Section 2 that, as K goes to infinity in Eq. (22), the two boundaries $\mathbf{X}$ and $\mathbf{Y}$ converge to each other and $\mathbf{F}_K{=}K(\mathbf{X}-\mathbf{Y})$ approaches the negative D'Alembert force $-\mathbf{F}_D$. To do that, we consider a unit square of [0 1]×[0 1] filled with an incompressible fluid in which an initially elliptic flexible elastic boundary is immersed. (Throughout this section we use dimensionless variables.) The density and viscosity of the fluid are 1.0 and 0.01, respectively.

The initial shape of the elastic boundary is an ellipse with major axis 0.62 and minor axis 0.5. This ellipse is parameterized by arclength $\mathbf{x}{=}\mathbf{X}_0(s)$, with s in the interval $[0,L_0]$, where L_0 is the perimeter of the chosen ellipse. (There is no simple formula for an ellipse parameterized by arclength, but the function $\mathbf{X}_0(s)$ is easily computed numerically.) Of course, $s{=}0$ is equivalent to $s{=}L_0$,

since an ellipse is a closed curve. At all times t, the immersed elastic boundary will be described in parametric form $\mathbf{x}=\mathbf{X}(s,t)$, with $\mathbf{X}(s,0)=\mathbf{X}_0(s)$. Here s is a Lagrangian parameter (a particular value of s corresponds to a particular material point), but s is not equal to arclength in general, since an elastic material by definition may stretch or shrink. It is important to note that the domain of s is fixed: It is always the interval $[0,L_0]$ despite any changes in length of the boundary that may occur. This is because the label s attached to a given material point does not change. The immersed boundary with mass density 0.25 interacts with the fluid and generates an elastic force by the following law:

$$\mathbf{F}(s) = c_s \frac{\partial^2 \mathbf{X}}{\partial s^2}(s), \tag{42}$$

where c_s=40 is the stiffness coefficient of the immersed boundary. It can be shown that the force law given by Eq. (42) describes a completely flexible boundary (no bending rigidity) in which the tension is equal to $c_s|\partial\mathbf{X}/\partial s|$. This is a special case of Hooke's law, the special feature being that the rest length of the immersed boundary is zero. Thus, the boundary tries to shrink to a point, but it is prevented from doing so by the incompressibility of the fluid. The equilibrium configuration of the immersed boundary is a circle containing the same area as the initial ellipse. The expected behavior is a damped vibration about this circular state, see Fig. 2.

By choosing s=arclength at the initial time, we create uniform tension (equal to c_s, since $|\partial\mathbf{X}/\partial s|$=1) at that time, and in this case the force given by Eq. (42) is normal to the immersed boundary at t=0. This choice of initial condition avoids a rapid transient (boundary layer in time) of tangential equilibration. Since s is not arclength in general, the force given by Eq. (42) may

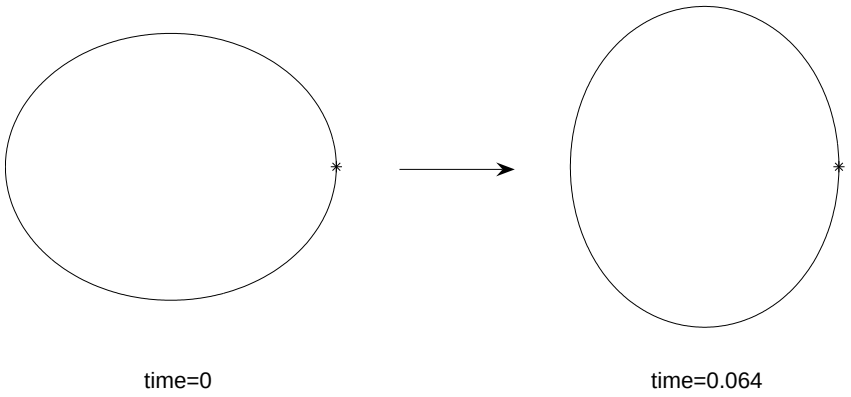

Fig. 2. Motion of a massive elastic boundary (closed curve).

have a nontrivial tangential component at all times other than t=0.

Now we choose the penalty spring constant K=200$\times N^2$ where N=32, 64, 128, 256, 512, and 1024. For each N, we also change the timestep Δt=3.2$\times 10^{-4}/N$, the space meshwidth Δx=$1/N$ and the boundary meshwidth Δs=$L_0/(4N)$. That is, when we refine the meshwidths Δx and Δs by a factor of 2, the timestep Δt is also reduced by the same factor. Although this is sufficient for stability within the range of parameters tested, the explicit computation of the boundary force may impose a more severe stability restriction once the parameters have been sufficiently refined, see [24]. Note that the penalty parameter K increases as the meshwidth and timestep are refined. The above manner of increasing it has the property that $K\Delta t^2$ is constant. In practice, this preserves the numerical stability of the scheme despite the increase in K, as one might expect since the frequency of oscillation of a spring-mass system with spring constant K is proportional to $\sqrt{K}$. It is fortunate for the overall effectiveness of the pIB method that one can increase K rapidly as the other numerical parameters are refined without inducing numerical instability.

The top panel of Fig. 3 shows the maximum distance between massless and

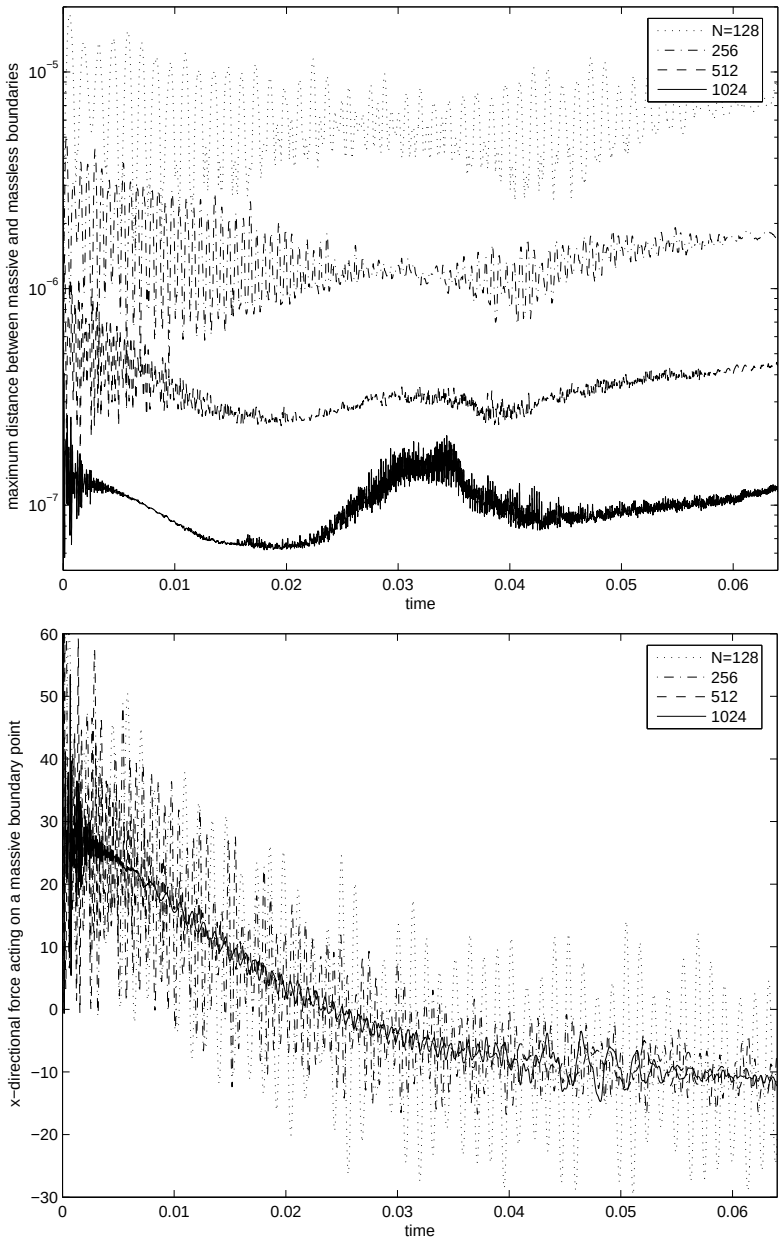

Fig. 3. The maximum amplitude $||\mathbf{X} - \mathbf{Y}||_\infty$ of the oscillation (distance between the two boundaries) and the x-directional force $-\mathbf{F}_K$ acting on a massive boundary point ('*' in Fig. 2) are plotted in time for each N. As N gets bigger, the computation gets refined and K increases. As the refinement goes on, the maximum distance of the two boundaries decreases (top panel), and the force acting on the chosen point converges (bottom panel).

massive boundaries for each N as a function of time. From the graph (note logarithmic scale, base 10) we can see that the maximum distance of the two boundaries goes to zero as the refinement proceeds, i.e., $||\mathbf{X} - \mathbf{Y}||_\infty \to 0$ as $K \to \infty$, which verifies our first claim. Note in particular that the amplitude of the rapid oscillation in the displacement between the two boundaries also tends to zero as the numerical parameters are refined and $K \to \infty$. The appearance of a constant amplitude of this rapid oscillation is a consequence of the logarithmic scale. The bottom panel of Fig. 3 compares the x-directional force acting on a point of the massive boundary for each N. The chosen point is marked with '*' in Fig. 2. The graphs verify our second claim that, $K \to \infty$, the force acting on the massive boundary $-\mathbf{F}_K$ converges (in fact, to $\mathbf{F}_D$, but we do not know the exact value $\mathbf{F}_D$). Note that the amplitude of the force oscillation appears to converge to zero as the numerical parameters are refined. This is remarkable when one considers that the stiffness parameter is approaching infinity.

Overall, Fig. 3 provides empirical evidence that the pIB method works as advertised. One certainly might worry about the high-frequency oscillation introduced into the system by the large value of the penalty stiffness K: In a nonlinear system, such a spurious high-frequency oscillation might interact with the frequency modes of interest in unpredictable ways. But the results in Fig. 3 indicate that this spurious oscillation is not only of high frequency; it is also of low amplitude, in fact, of an amplitude that approaches zero as the numerical parameters are refined. This suggests that the pIB method may indeed be convergent. We provide further empirical evidence for convergence in the remainder of the section.

Table 1 shows the convergence ratios of the velocity fields computed by the

Table 1

Convergence ratios in L_2 and L_∞ of the velocity field

N	$\frac{\|\|u_N-u_{2N}\|\|_2}{\|\|u_{2N}-u_{4N}\|\|_2}$	$\frac{\|\|u_N-u_{2N}\|\|_\infty}{\|\|u_{2N}-u_{4N}\|\|_\infty}$	$\frac{\|\|v_N-v_{2N}\|\|_2}{\|\|v_{2N}-v_{4N}\|\|_2}$	$\frac{\|\|v_N-v_{2N}\|\|_\infty}{\|\|v_{2N}-v_{4N}\|\|_\infty}$
32	1.9235	1.1200	2.0367	1.7614
64	2.1044	1.9523	2.0297	1.6366
128	2.0944	1.8466	2.0514	1.8517
256	2.0930	1.8125	2.0430	1.8451

pIB method. Since we do not know the exact solution of the problem, one convergence ratio needs three numerical solutions for three consecutive N's. Let (u_N, v_N) be the velocity field at the chosen time t=0.064, which is roughly half the longest period of the observed vibration of the immersed elastic boundary, and let $||\cdot||_2$ and $||\cdot||_\infty$ be L_2 and L_∞ norms, respectively. Table 1 shows that the pIB method has the convergence ratio roughly equal to 2 (at least in the L_2 norm; perhaps somewhat worse in the L_∞ norm), which implies that the scheme is first order accurate. Although this provides empirical evidence for the convergence of the pIB method, it also underscores the purely formal nature of its formal second order accuracy. In the formulation of the pIB method (Eqs. (16)-(23)) the appearance of the Dirac delta function induces jumps in certain quantities across the immersed boundary. Specifically, there is a jump in pressure (produced by the normal component of the boundary force) and a jump in the normal derivatives of the velocity components (produced by the tangential component of the boundary force). This lack of smoothness reduces the accuracy of the scheme. These same considerations apply to the original IB method for the massless case just as they do to the pIB method considered here.

In this connection, note that Griffith and Peskin [10] have recently described an immersed boundary computation in which the immersed "boundary" has finite thickness (it is an immersed elastic shell), and in this computation *actual* second order accuracy is achieved by a formally second-order accurate IB scheme, despite the sudden changes in material properties where the shell meets the fluid. Thus, the case of an immersed elastic structure with finite thickness already has sufficient regularity for the formal accuracy of the IB scheme to manifest itself. Although we have not yet applied the pIB method to immersed elastic structures with finite thickness, it seems reasonable to conjecture that the pIB method of this paper would similarly exhibit actual second-order accuracy in such a case.

In the case of a thin immersed boundary, though, it is natural to ask what is the benefit of using a formally second-order accurate IB or pIB scheme, since only first-order accuracy will be realized in practice. We believe that the principal benefit is the reduction in numerical viscosity associated with the skew-symmetric differencing of the nonlinear terms in comparison to a first order upwind scheme. This benefit has in fact been demonstrated in [15], see also [20]. It is not clear, though, whether the skew-symmetric differencing of the nonlinear terms in any sense calls for the second order accurate time discretization that is also part of the formal second order accuracy of our scheme. In some applications, it seems possible to use the skew-symmetric differencing of the nonlinear terms together with first-order accurate time discretization, but in other applications this combination appears to be unstable.

To show that the reduction of accuracy comes from the spatial discretization and perhaps also from the increase of K, we have also conducted a convergence study involving time discretization only, with the spatial discretization

Table 2

Convergence ratios with the spatial discretization and penalty parameter fixed

M	$\frac{\|u_M - u_{2M}\|_2}{\|u_{2M} - u_{4M}\|_2}$	$\frac{\|u_M - u_{2M}\|_\infty}{\|u_{2M} - u_{4M}\|_\infty}$	$\frac{\|v_M - v_{2M}\|_2}{\|v_{2M} - v_{4M}\|_2}$	$\frac{\|v_M - v_{2M}\|_\infty}{\|v_{2M} - v_{4M}\|_\infty}$
1	4.0075	4.0112	4.0070	4.0070
2	4.0018	4.0028	4.0017	4.0017
4	4.0010	3.9991	4.0009	4.0009
8	3.9934	4.0124	3.9926	3.9926

and the penalty parameter fixed. Table 2 shows the convergence ratios obtained in this way. We fix N=256 and Δx=$1/N$, and change the timestep Δt=$1.25 \times 10^{-6}/M$ where M=1,2,4,8,16, and 32. The penalty stiffness K=10^7 is fixed like the spatial discretization. The simulation goes up to time t=0.064. Table 2 shows substantially better convergence ratios than Table 1, and indeed comes to second order accuracy, which would be a convergence ratio of 4. In this case, though, convergence is not to the true solution of the partial differential equations, but rather to the solution of the ordinary differential equation system defined by the spatially discretized pIB method, in which time is still continuous, and moreover with constant penalty parameter K. Thus the results shown in Table 2 only verify that our time-stepping scheme is second-order accurate, as claimed. The results in Table 2 have no bearing on the quality of the spatial discretization, nor on the effectiveness of the penalty approach

To illustrate the influence of mass on the behavior of an immersed elastic boundary, Figure 4 compares the massive boundary and massless boundary cases by plotting horizontal and vertical radii of the boundary curve in each

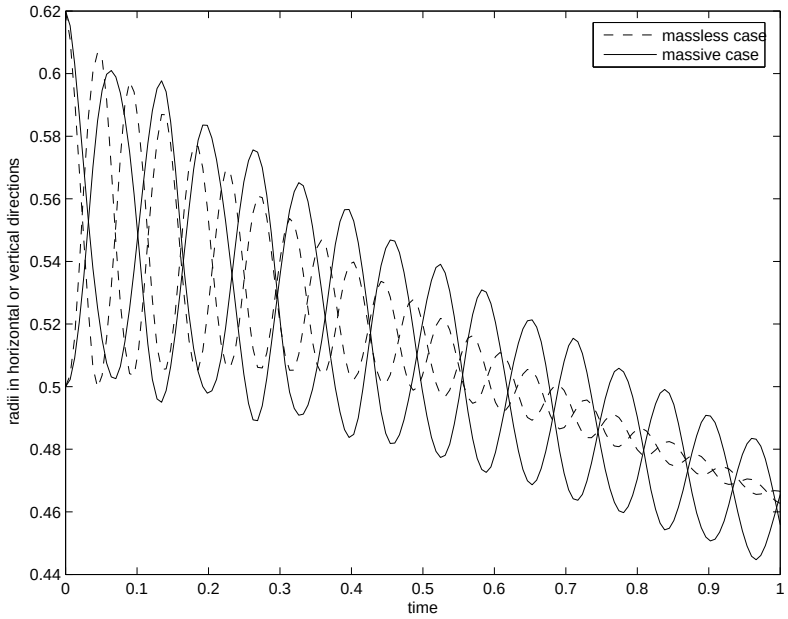

Fig. 4. The influence of boundary mass on the behavior of an immersed elastic boundary in the form of a simple closed curve in a two-dimensional fluid. The curve is initially elliptical, and the horizontal and vertical radii are plotted as functions of time. Both cases show a damped oscillation as the boundary approaches a circular shape, but the massive boundary oscillation has a lower frequency and is sustained longer than that of the massless boundary. The overall downward trend is a form of numerical error (see text) that is somewhat exaggerated here by the vertical scale of the plot.

case as functions of time. Of course, the fluid has mass in both cases. In each case, the two radii oscillate in a damped manner, with one radius having its maximum at about the same time as the other radius is at a minimum. As one would expect, the oscillation has a lower frequency and is sustained longer in the massive than in the massless case.

The overall downward trend of the plots in Figure 4 represents loss of volume (area), a form of numerical error that is typical of IB computations but can be virtually eliminated by a technique known as "improved volume conserva-

tion" [25], which is not implemented here, since most of our applications do not involve enclosed volumes. For other approaches that also lead to better conservation of volume, see [4] and [16].

5 Applications

5.1 Flapping Filament in a Flowing Soap Film

The first application is a flapping flexible filament in a flowing soap film. This replicates a simulation done by Luoding Zhu [27,28] by the method mentioned in the introduction, in which the mass of the immersed boundary is attributed instead to the nearby fluid. Our purpose here is to show that this problem can also be done by the pIB method.

The physical experiment that inspired both Zhu's simulation and ours was performed by Jun Zhang [26] in the Courant Institute WetLab. It involves a vertical soap film that flows downwards between parallel wires under the influence of gravity. A thread, or "filament", anchored at its upper end but otherwise free, is suspended in the flowing soap film. Two qualitatively different modes of filament behavior are seen in these experiments. In one, the filament settles into a straight steady configuration, hanging vertically in the flow. In the other, the filament settles into a pattern of sustained flapping, like a flag in the wind. For some choices of parameters, only one of these two behaviors is seen, but there are other choices of parameters for which both behaviors are possible. In the latter ("bistable") cases, the choice of which behavior occurs is determined by the initial conditions.

We have used the same parameters as in [27,28], see Table 3. For the most part, these are also the parameters of the experiment. The most important difference is that the film viscosity has been increased (in both computations) by a factor of 100. The experimental Reynolds number is about 20,000, but the computational Reynolds number is about 200.

Before we discuss the results of the filament simulation, we first consider two special aspects of the simulation. The first issue is about how big the spring coefficient K, which appears in Eq. (22), should be to keep two boundaries (massless and massive) close and not to make any computational instability. We choose a constant K by trial and error. As K gets bigger, a larger restoring

Table 3

Parameters of flapping filament simulations

Parameters	magnitude	units
Film Density	3×10^{-4}	g/cm^3
Film Viscosity	1.2×10^{-3}	g/cm·s
Film Inflow Velocity	$(-280) - (-200)$	cm/s
Filament Density	$1 \times 10^{-5} - 4 \times 10^{-4}$	g/cm
Filament rigidity	0.1	erg·cm
Filament Length	$2.0 - 3.0$	cm
Reynolds number	$100 - 210$	
Gravitational acceleration	980	cm/s^2
Domain (rectangle)	9×18	cm×cm

force is generated, and the computation becomes more unstable, in which case, we need to reduce the timestep Δt. Our strategy is to choose an allowed distance between the two boundaries, to adjust K until it is large enough that this allowed distance is not exceeded, and to reduce Δt, if necessary, as K is increased to avoid numerical instability. Although this process does involve trial and error, the tests for success are extremely simple, and it is easy to zero in on good choices for K and Δt, as follows: The test for whether K is large enough is simply to monitor the maximum distance between any massless immersed boundary point and its massive twin. Since this distance will be inversely proportional to K (for large K) it is easy to see from the results of a trial run how K needs to be adjusted for the next trial. Numerical instability resulting from too large a choice of Δt is unmistakable: the immersed boundary seems to explode. Also, in adjusting Δt when K changes, one can use the rule of thumb that (for a given spatial discretization) stability is determined by $K(\Delta t)^2$. This has the pleasant consequence that to make K four times larger (for example), one only needs to make Δt twice as small.

Fig. 5 shows that the maximum distance between the two boundaries is controlled within around 1.5×10^{-4} meshwidth. For this simulation, we use $K = 10^8 \mathrm{g/cm{\cdot}s^2}$ and $\Delta t = 5 \times 10^{-7}\mathrm{s}$. Despite the oscillations seen in the figure, the massless and massive boundaries stay close to each other as the simulation proceeds.

The second question is about the way of driving a desired inflow. In section 3 we have suggested a method to drive an inflow which involves an external body force term $\alpha_0(\mathbf{u}_0(t) - \mathbf{u}(\mathbf{x}, t))\theta(\mathbf{x})$ on the right hand side of the Navier-Stokes (momentum) equation. Here, α_0 is a large constant, $\mathbf{u}_0(t) = (U_0(t), V_0(t))$ is the desired inflow velocity at time t, and $\theta(\mathbf{x})$ is the characteristic function

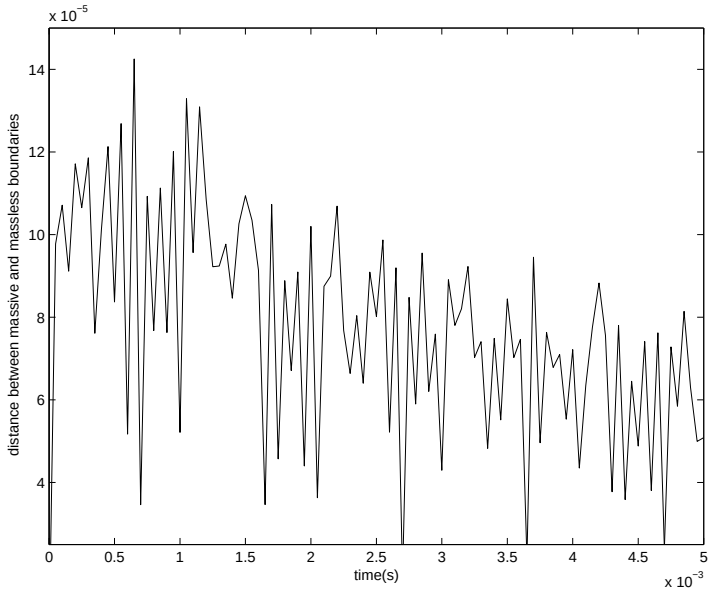

Fig. 5. The maximum distance in units of meshwidth between the massless and massive boundaries as a function of time. The maximum distance between two boundaries stays below 1.5×10^{-4} meshwidth here and even after the time shown in this graph.

of the chosen region in which we want to specify the velocity. If we use this method, the size of the constant α_0 should be taken carefully. In order to make the inflow velocity obtained by solving the Navier-Stokes equations close to the desired inflow in a reasonably short amount of time, the constant α_0 should be large. If it is too large, however, α_0 can be a source of numerical instability. If such an instability appears, however, it can be eliminated by reducing the time step Δt. In this way it is possible to control the inflow velocity closely enough for practical purposes.

Fig. 6 shows the desired inflow velocity $V_0(t)$ and the error of the calculated velocity $v(\mathbf{x}, t)$ at one point $\mathbf{x}$ in the specified region. The top graph represents the y-component $V_0(t)$ of the desired inflow, and the bottom graph is the absolute error of the y-component of the induced velocity $v(\mathbf{x}, t)$ from the

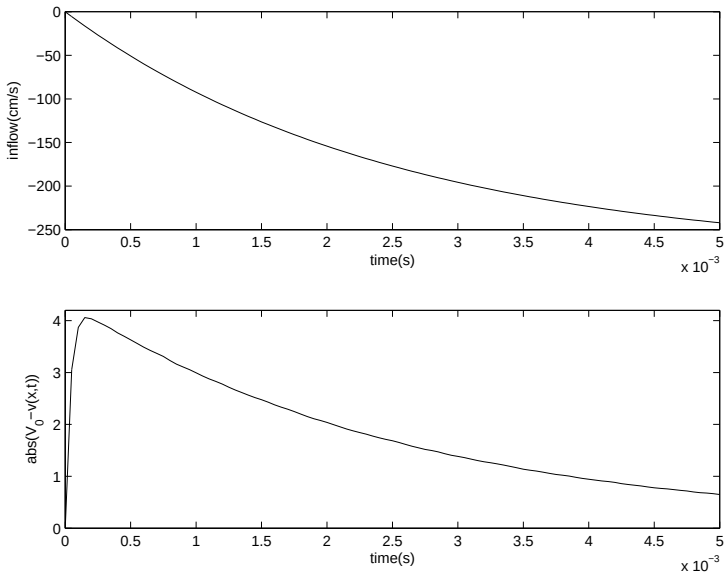

Fig. 6. The top panel represents the y-component $V_0(t)$ of the desired inflow, and the bottom panel is the absolute error of the y-component of the calculated velocity $v(\mathbf{x}, t)$ from the desired velocity $V_0(t)$ at one specified point $\mathbf{x}$, i.e., $|V_0(t) - v(\mathbf{x}, t)|$.

desired velocity. We can see that, except for the very beginning of the time, both velocities match well, i.e. our method of deriving the inflow velocity forces very quickly the velocity in the chosen region to approach the desired inflow. Note that, to avoid a sudden change of the inflow velocity, $V_0(t)$ is smoothly increasing in magnitude: $v_0(1 - \exp(-t/t_0))$, where v_0 is the final velocity (y-component) and t_0 is an arbitrary chosen constant. Also note that, even though this graph shows only a very short time result, the match of both velocities is getting better as the time proceeds. Control of the inflow velocity is accurate to within 1% by the time 3×10^{-3}s have elapsed and continues to improve thereafter.

An important result of [27,28] is that filament mass is needed for sustained flapping to occur. Our simulations by the pIB method give the same result. Fig. 7 compares two cases: one has an almost massless filament with density

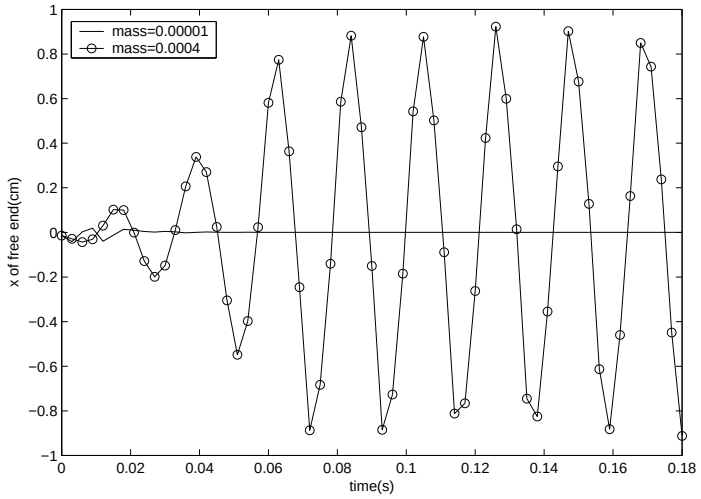

Fig. 7. The x-coordinate of the free ends of two filaments are plotted as functions of time. The sustained flapping motion is observed only in the case of a filament with enough mass.

1×10^{-5}g/cm and the other has a heavier filament with density 4×10^{-4}g/cm, which is the value used in [27,28]. This value is the same as in Zhang's experiments [26], allowing for a factor of 2 to account for a bulge of the film that forms around the filament as a result of surface tension. Other conditions are also the same: the inflow velocity is -280cm/s, the filament length is 3cm, and the initial configurations are same as in [27,28]. The figure plots the x-coordinate of the free end of the two filaments as functions of time. As time proceeds, while the small mass filament rapidly becomes straight and then remains straight like a rigid body, the larger mass filament settles into a flapping mode with the total excursion of the free end 1.8cm and the frequency 48/s.

Fig. 8 shows the vorticity contours in both cases at two fixed times. Note the concentrated vorticity on the two sides of the domain as well as around the filament and in the wake of the filament. The concentrated vorticity near the side walls represents the boundary layers that are generated there. (Recall

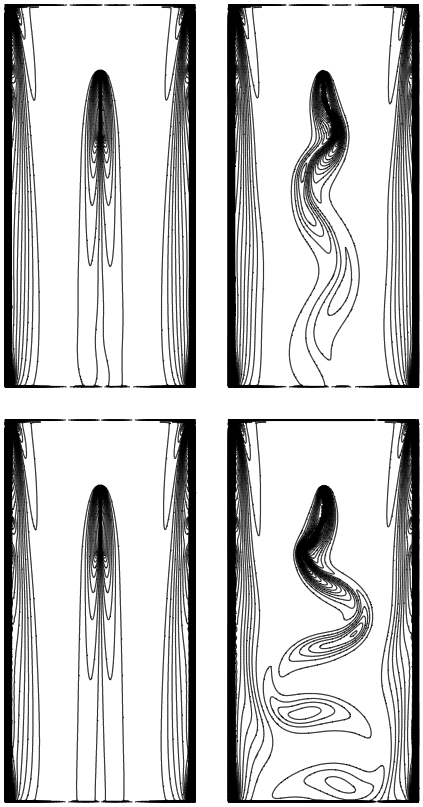

Fig. 8. Motion of two filaments with different mass densities: 10^{-5}g/cm (left panels) and 4×10^{-4}g/cm (right panels). The vorticity contours are drawn at two different times: 0.06s (top) and 0.18s (bottom).

the manner in which the no-slip condition has been imposed on the side walls through the use of tether points, as explained in Section 3.) The concentrated vorticity on the two sides of the filament similarly represents the boundary layers generated by the shearing effect of the (almost inextensible) filament on the flow. In the case of the more massive filament, discrete vortices of opposite sign are shed as the filament flaps.

In order to explore the issue of bistability, we choose a filament of length 2cm and mass density 4×10^{-4}g/cm. The steady inflow velocity v_0 is -200cm/s, which produces a Reynolds number of 100. Two different cases are created by two different initial perturbations from the straight configuration of the fila-

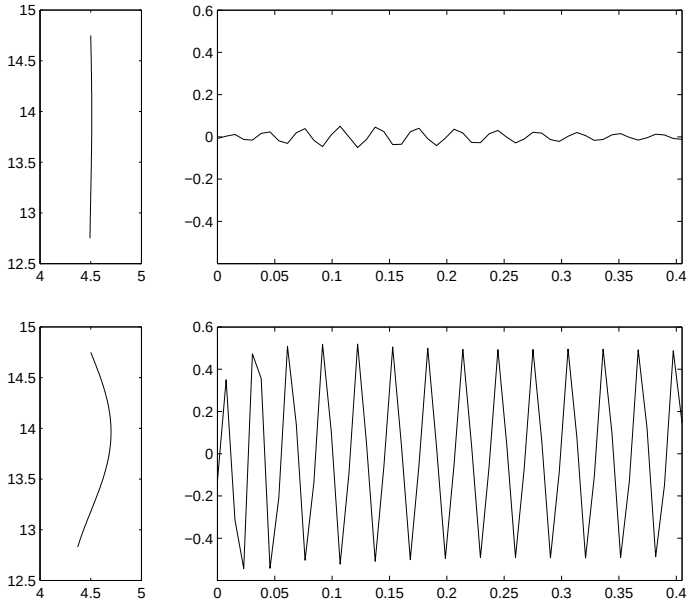

Fig. 9. The left two pictures show the initial configurations of a filament in the shape of a sine curve with two different amplitudes: 0.01cm (left-top) and 0.2cm (left-bottom). The x-coordinate of the free ends of the two filaments are plotted as functions of time in the right two panels. While the case with a small initial perturbation almost stops flapping (top), the large initial perturbation case quickly settles into a sustained oscillation with a larger amplitude than that prescribed initially (bottom).

ment, see Fig. 9. Since our model of the filament has as its initial configuration a sine curve, the initial perturbation can be expressed by the amplitude of the sinusoid. The left-top picture in the figure shows the initial configuration of the filament with $0.005cm$ amplitude initial perturbation. For the larger initial perturbation case, we choose 0.2cm amplitude (left-bottom). The right panels of the figure show the x-coordinates of the free ends as functions of time. We can see that, while the filament with a small initial perturbation almost stops flapping, the filament with the large initial perturbation sustains its flapping motion. Fig. 10 shows the vorticity contours in both cases at two fixed times.

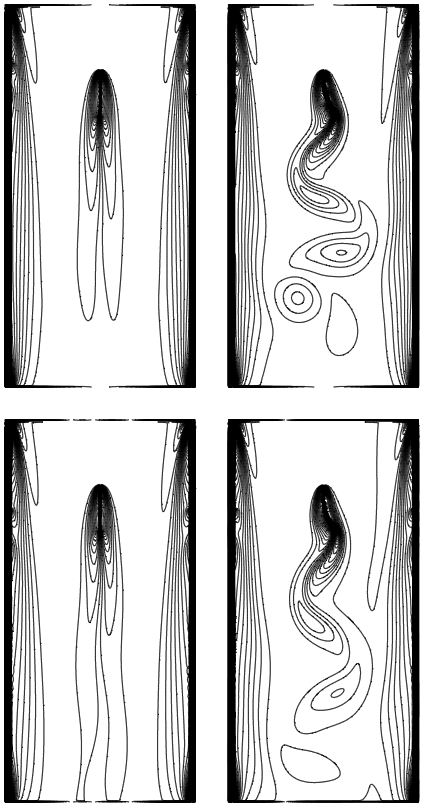

Fig. 10. Motion of a filament with all the same circumstances except the initial conditions (see figure 8): a small perturbation (left) and a large one (right). The vorticity contours are drawn at two different times: 0.06s (top) and 0.18s (bottom).

Despite the qualitative similarity with Zhu's results, there are some quantitative differences: With Reynolds number 210 (the inflow is -280cm/s and the filament length is 3cm) and 4×10^{-4}g/cm filament mass density, both simulations yield about 50 Hz for the flapping frequency of the filament. The total excursion of the filament free end of our experiment, however, is 1.8 cm which is somewhat different from Zhu's result 2.1cm, but is close to the real experimental data 1.5 cm [26]. For the case of the flapping filament with -200cm/s inflow velocity and 2cm filament length, Zhu's and ours have a similar frequency 36 Hz, but again the total excursions are different: Zhu's is 1.5cm and ours 1.0cm. These differences between the two methods are presumably ex-

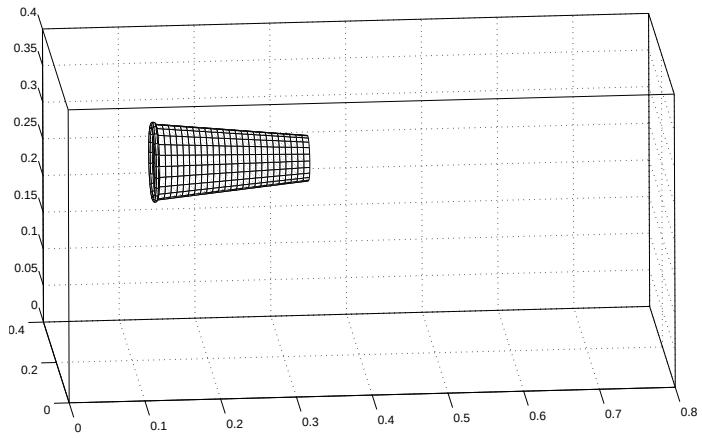

Fig. 11. Initial configuration of windsock and computational domain which is a box $0.8 \times 0.4 \times 0.4$ m^3. The left end of the windsock is attached to a ring which is fixed in the domain. Wind is driven from left to right by prescribing inflow velocity at the left face of the domain.

pressions of the numerical error of one method or the other, or most likely of both, and as such would become smaller if the resolution of both computations were refined. We are not trying to claim here the superiority of one method over the other, and the fact that the pIB method comes closer to the experiment in the amplitude of the motion may well be fortuitous, since the computations are being performed at a much lower Reynolds number than the experiment.

5.2 Windsock

A windsock is a device used at airports to make the speed and direction of the wind visible to pilots. The magnitude of the wind and the weight of the windsock oppose each other to determine the extent to which the windsock stands out horizontally or hangs down vertically. Thus the mass of windsock is

Table 4

Parameters of the 3-D simulations

Parameters	Windsock	Flag	units
Fluid Density	1.2	1.2	kg/m^3
Viscosity	3.0×10^{-3}	5.0×10^{-3}	kg/m·s
Inflow Velocity	$0.0 - 5.0$	$0.0 - 4.6$	m/s
Density(mass)	0.08	0.09	kg/m^2
Dimensions	$0.2 \times 0.08\pi$	0.333×0.25	m
Gravitational acceleration	9.8	9.8	m/s^2
Reynolds number	up to 400	up to 370	
Domain(Box)	$0.8 \times 0.4 \times 0.4$	$1.28 \times 0.64 \times 0.64$	m×m× m

essential for the windsock simulations. In filament simulations, we need mass of the filament because the flapping motion requires the inertia of the filament. In this case, however, the mass is needed for the gravitational force.

For the simulation, we make the windsock with the shape of a part of a slightly tapered cone, of which the circle at the upstream end is only a little bigger than that at the downstream end. The upstream end of the windsock is attached to a fixed ring, see Fig. 11. The fixed ring is created by the method explained in Section 3 generating a fixed no-slip boundary which can be of any shape. Then we make a wind blow in one direction from left to right.

Table 4 contains the parameters used here (and also in the simulations of a flag considered below). The size of the windsock, which is represented by its

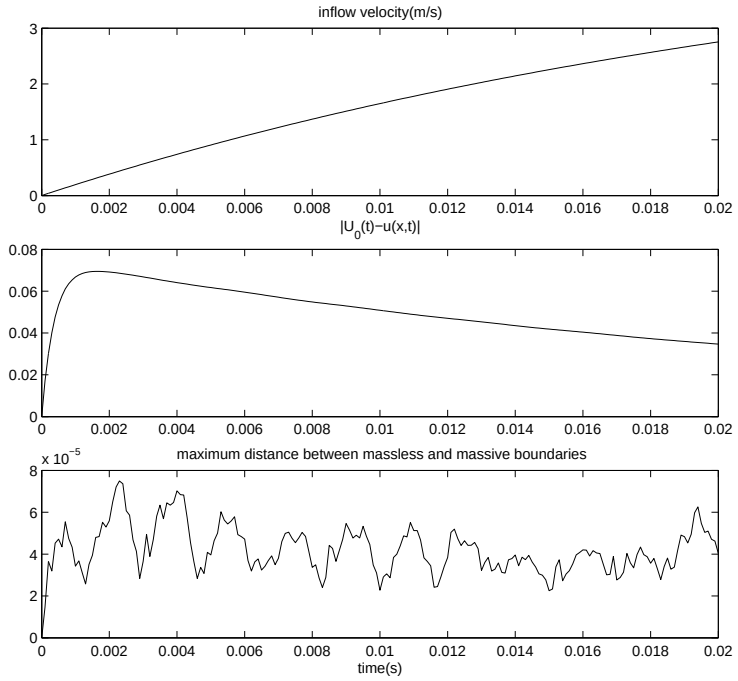

Fig. 12. Errors in the windsock simulation. The top graph represents the x-component $U_0(t)$ of the desired inflow, and the middle graph is the absolute error of x-component of the induced velocity $u(\mathbf{x}, t)$ from the desired one $U_0(t)$ at one specified point $\mathbf{x}$,i.e. $|U_0(t) - u(\mathbf{x}, t)|$. The maximum distance (in units of meshwidth) between two boundaries is shown in the bottom graph. Note that the maximum distance between the massive and massless boundaries is always less than 8×10^{-5} meshwidths.

length times the circumference of the fixed circle to which it is attached, and the surface mass density of windsock material are chosen arbitrarily. The other parameters are all realistic except viscosity. Air has viscosity 2×10^{-5}kg/m·s which is 150-250 times smaller than the viscosities of the simulations. The reason for this modification is that we want to reduce Reynolds number (Re) in the simulations. We believe that, with computational resolution affordable so far, we can properly compute up to a Reynolds number of several hundreds. With the modified viscosity, the windsock has maximum Re=396 based on its

length, and the flag considered below has maximum Re=370 based on its width. Note that, in all applications considered in this paper, we modify the viscosities usually existing in real experiments to make the Reynolds number between 100 and 400.

In order to see if the method for driving an inflow and the pIB method for keeping the massless and massive boundaries close work well in the 3-D simulation, we draw Fig. 12 as in the previous section. The top graph is the x-component $U_0(t)$ of the desired inflow velocity which we want to drive in time. The desired velocity is gradually increasing up to 5.0m/s (not seen in the graph). The comparison of $U_0(t)$ with the induced velocity $\mathbf{u}(\mathbf{x}, t)$, which is nothing but the solution of the fluid equations at a specified point $\mathbf{x}$, is shown in the middle graph. For the windsock simulation, we use the constant $\alpha = 10^5/\text{s}$ and the timestep $\Delta t = 2 \times 10^{-5}\text{s}$. The middle graph shows that our inflow driving method works well even in 3-D simulation.

The bottom graph shows the maximum distance (in units of meshwidth) between the massless boundary $\mathbf{X}(r, s, t)$ and its massive counterpart $\mathbf{Y}(r, s, t)$. With the spring coefficient $K = 10^7\text{kg}/(\text{ms})^2$ and the timestep $\Delta t = 2\times 10^{-5}\text{s}$, we can keep these two boundaries close without any computational instability. Note that the error of the driven inflow from the desired inflow and the distance between the two immersed boundaries are kept within 0.07m/s and 8×10^{-5} meshwidths, respectively.

Fig. 13 compares the movements of the windsock with different wind speeds. For a while, there is no wind (top) and, after that time, the wind slowly increases up to 2.5m/s (middle) and 5.0m/s (bottom). In the figure, we choose a vertical plane that is a plane of symmetry for the windsock, record only the

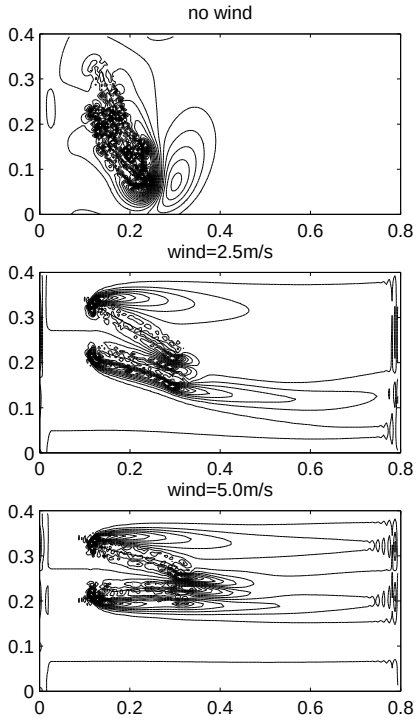

Fig. 13. Three different cases of a windsock are compared. Vorticity contours in a vertical plane that bisects the windsock are plotted. The top frame has no wind and the windsock falls down. In the case of a small wind speed (middle), the windsock finds an equilibrium angle that depends on the wind speed. The bottom frame has a large wind speed which straightens the windsock.

velocity components parallel to that plane, and plot contour lines of vorticity of that two-dimensional velocity field. The top windsock has no wind and naturally falls down. The bottom one has enough wind to make the windsock stand out almost straight in the direction of the wind. The middle windsock has half the wind speed of the bottom one and slopes downwards at about 30^o below the horizontal. Thus we see that each windsock finds a different angle (with respect to the wind direction) depending on the wind speed.

Figs. 14 and 15 show perspective views of the windsock with the two different magnitudes of the wind considered above: 5.0m/s on the top and 2.5m/s on

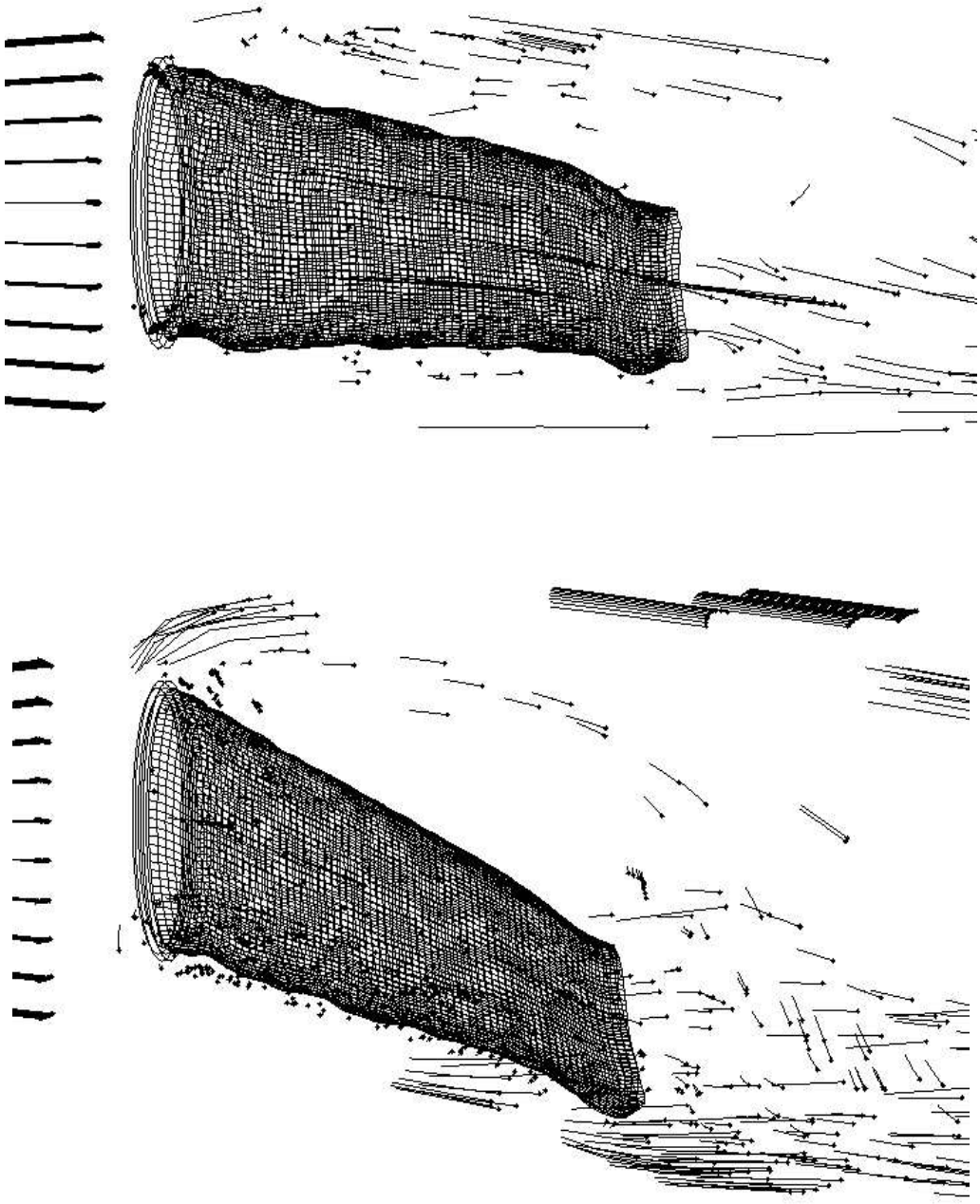

Fig. 14. Sideview of windsocks with different wind speeds: 5.0m/s (top) and 2.5m/s (bottom). The larger the wind speed is, the less tilt angle windsock has. The lines around the windsocks show the motion of some fluid markers. Each marker has left a trail showing its recent trajectory.

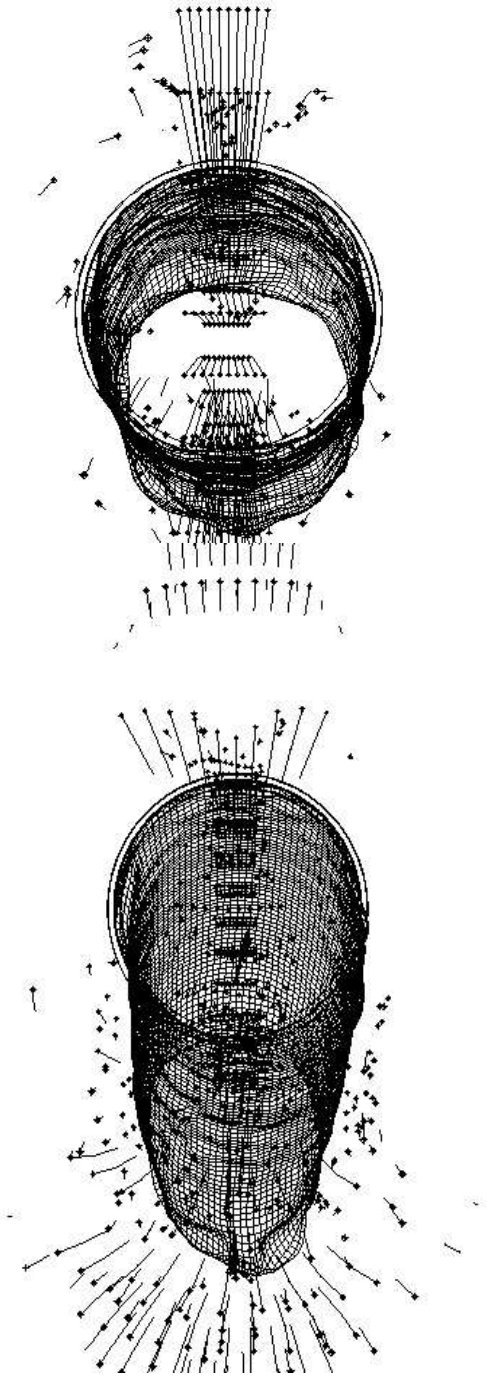

Fig. 15. The same situation as in Fig. 14 with the view looking into the wind.

the bottom of the figures. A few fluid markers are also shown. Each marker has left a trail showing its recent trajectory (streak line).

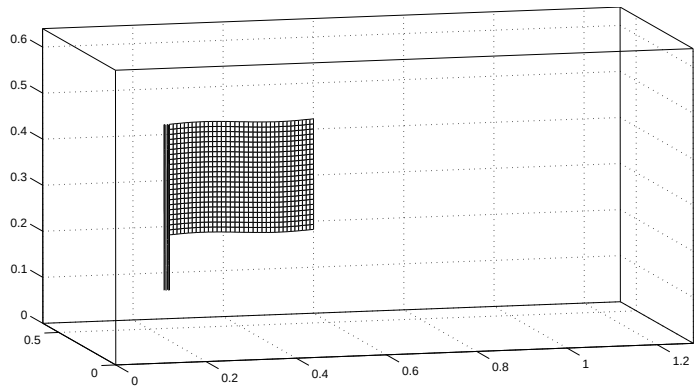

Fig. 16. Initial configuration of a flag and the computational domain which is a box $1.28\times0.64\times0.64\text{m}^3$. The left end part of the flag is attached with the flagpole which is fixed in the domain. The flow of the wind is from left to right.

5.3 Flag in Wind with Gravity

The third application is a flapping flag in the wind. This application looks like the 3-D generalization of the 2-D filament simulation. In the flag problem, however, the wind flow is from the side and the gravity force acts downward while both in the 2-D filament case are in the downward direction. Thus the importance of the mass of the flag turns out to combine those of the filament and the windsock. Because of gravity, the flag will sag, and, because of the flag's inertia, we can also expect to see it flap. For related work on modeling cloth in the context of computer graphics, see the excellent survey [17], and the references therein.

The parameters are shown in Table 4 and the initial configuration of the flag is drawn in Fig 16. The initial configuration of the flag is that of a sine curve from the top view. A vertical flagpole in the form of a cylinder is provided. It is fixed in place in the same way as the ring of points that anchors the

windsock, see Section 5.2. The left edge of the flag is attached directly to the flagpole, and wind blows from left to right in these simulations.

Fig. 17 shows the motions of the flag in two different situations. As we can expect, when there is no wind, the flag falls down and stops moving, see the bottom frame in Fig. 17. But if the wind is strong enough, the flag overcomes gravity and straightens up in the wind direction. With this sufficiently large wind speed, we can see the flag flapping (top of Fig. 17).

In order to observe more clearly the flapping and sagging motion of the flag in the top of Fig. 17, we can plot the trajectories of several points on the downstream edge of the flag as functions of time. In Fig. 18, choosing three points (top, middle, and bottom) of the downstream end of the flag, we plot the three coordinates of their motion in each case. The top frame shows the x-coordinate of the motion which represents the movement in the direction of wind. Since the y-axis is perpendicular to both the flag and the wind direction, the middle frame clarifies the flapping motion of the flag. The bottom and middle points of the flag go through a relatively large oscillation but the top undergoes only a small oscillation. One more interesting observation is that the three points are moving out of phase, i.e. the bottom part is moving ahead of the top part. These two phenomena (out of phase and different amplitudes) can be easily seen when we watch the motion of a real flag. The bottom panel shows the z-coordinate (height) of the flag. The flag settles down little bit at the beginning of the computation and then stops sagging because of the drag force of the wind.

By analogy to the results of filament simulation (Section 5.1), we expect that, for the sustained flapping motion, a flag should have several conditions satis-

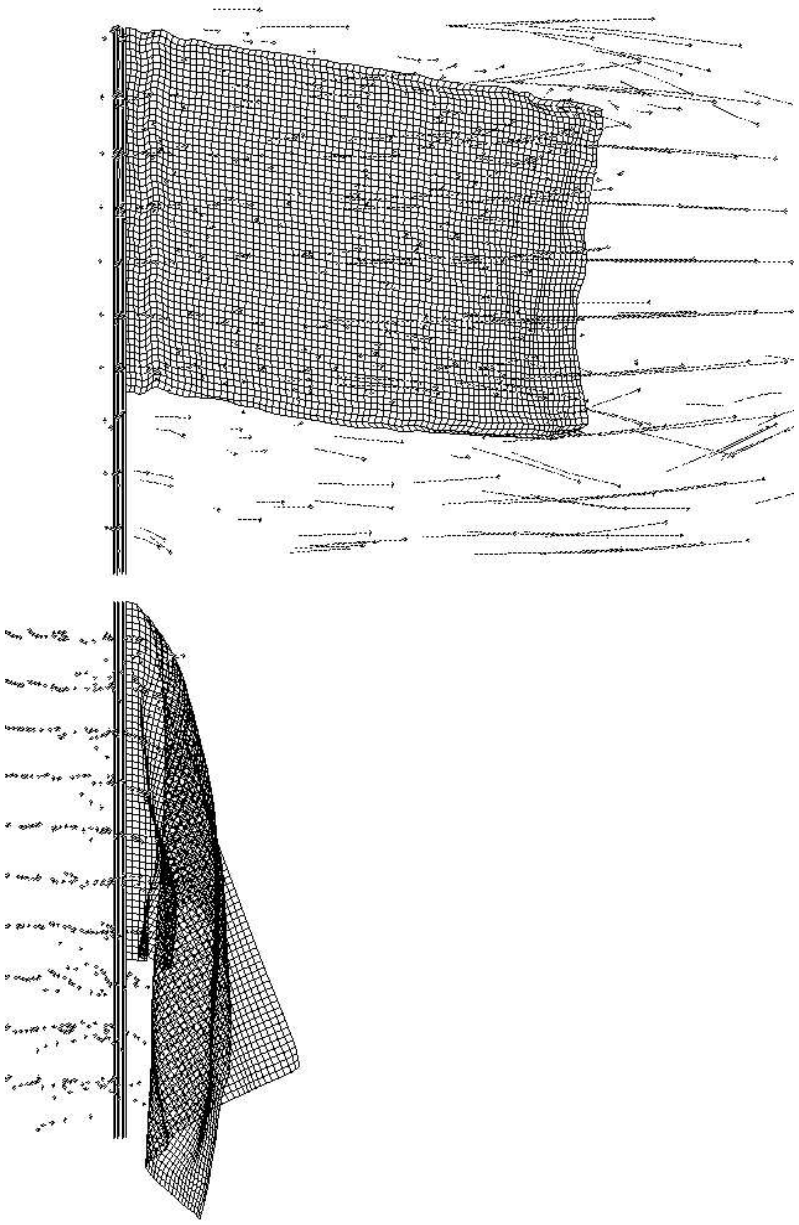

Fig. 17. A large enough windspeed (top) helps the flag overcome the gravity and sustain a flapping motion. We can see the flag still sagging slightly. In the case of no wind (bottom), the flag falls down and stops moving. Note the folds which are a consequence of the (small) bending rigidity of the flag. (time=1.2s)

fied: its mass, wind speed (or Reynolds number), and the initial perturbation must all be sufficiently large. Here the initial perturbation means the amplitude of the sine curve which is the image of flag viewed in the plane perpendic-

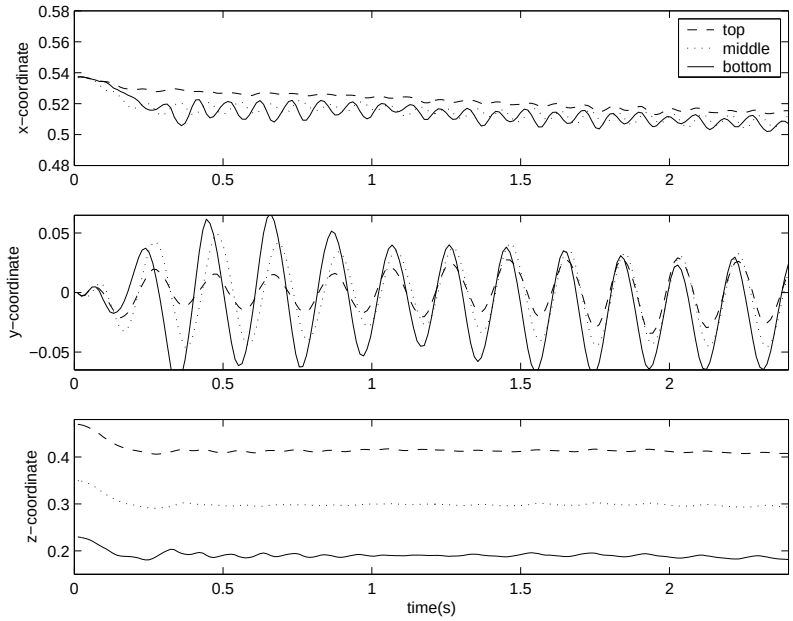

Fig. 18. Cartesian coordinates of the positions of three different material points on the downstream edge of the flag. Three points are chosen at the top, middle, and bottom of the downstream edge. The x-axis is in the direction of the wind, the y-axis is perpendicular to the flag and the wind direction, and the z-axis is vertical. We can see from the middle panel that the flag is flapping, and, from the bottom panel, that the flag sags a little at the beginning of the computation and then stops sagging. Note (middle panel) that the flapping motions of top and bottom of the flag are out of phase, i.e. the bottom part runs ahead of the top part. Also the amplitude of the flapping motion is much greater at the bottom of the free edge than at the top.

ular to z-axis (see Fig. 16). In order to check whether the above expectation is correct, we can study the case of a flag with a deficit in each condition and compare the result with that of a flapping flag. As the conditions suitable for a sustained flapping, for example, we have the flag density 0.09kg/m^2, wind speed 4.6m/s (Reynolds number 370), and the initial perturbation 0.01m. This case produces the Figs. 17 (top) and 18.

The first deficit case is a massless flag with all other parameters the same as that of the flag considered above. From the result in the filament case, we can easily guess what will happen. Without enough inertia force to kick the flag from side to side, the flag will not flap. Because there is also no gravity force on the massless flag, there will be no sagging motion either. Fig. 19 shows the motion of the massless flag computed by the IB method. Comparing with the flag in the top panel of Fig. 17, here we cannot see any sagging of the flag. To make the comparison more clear, we plot the (y, z)-coordinates of the lower downstream corners of both flags as functions of time, see Fig. 20. As we mentioned before, the y-coordinate shows the flapping motion best, and the z-coordinate shows how much a flag has sagged. From the figure, we can see that only the massive flag is flapping (top panel) as well as sagging (bottom panel). Notice that the massless flag actually rises a little instead of going down. This must be a subtle effect of the flagpole, since the only thing that breaks the up-down symmetry of the problem is that the flagpole extends below but not above the flag.

Now, to see the case with a small wind speed, keeping all other conditions of the flapping flag case, we take wind speed 4.0m/s which reduces the Reynolds number to 320. Fig. 21 shows the flag at a fixed time, and Fig. 22 compares (y, z)-coordinate trajectories in the same manner as before. From the comparison of the y-coordinates in Fig. 22, we can see that, with a small wind speed, the flag does not flap. As we can see from Figs. 19 (no mass) and 21 (slow wind), however, although these two flags both do not flap, their detailed motions are different. The massless flag in Fig. 19 is straightened like a rigid body, which reminds us of the small mass filament case. On the other hand, the massive flag in a slow wind (Fig. 21) keeps a particular bent shape such

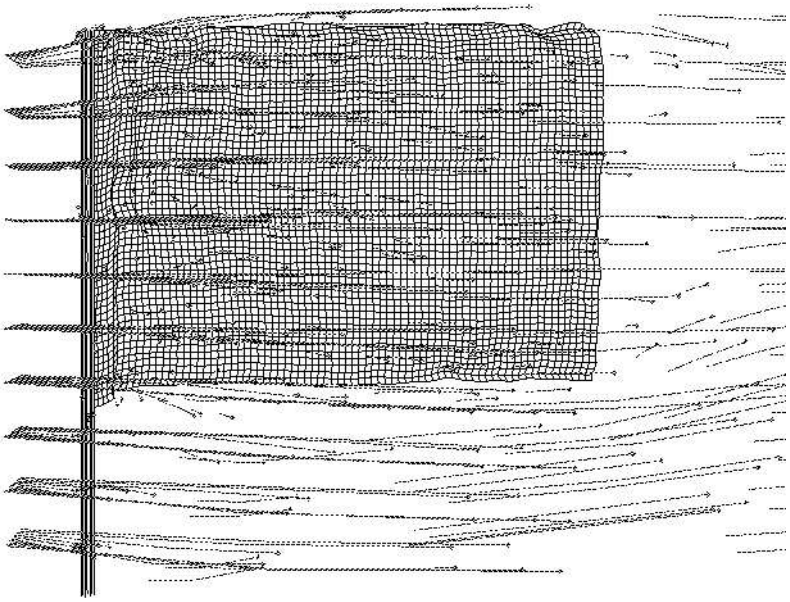

Fig. 19. Motion of the massless flag. Even though all other conditions are same as in Fig. 17 top, the massless flag neither flaps nor sags. See also Fig. 20.

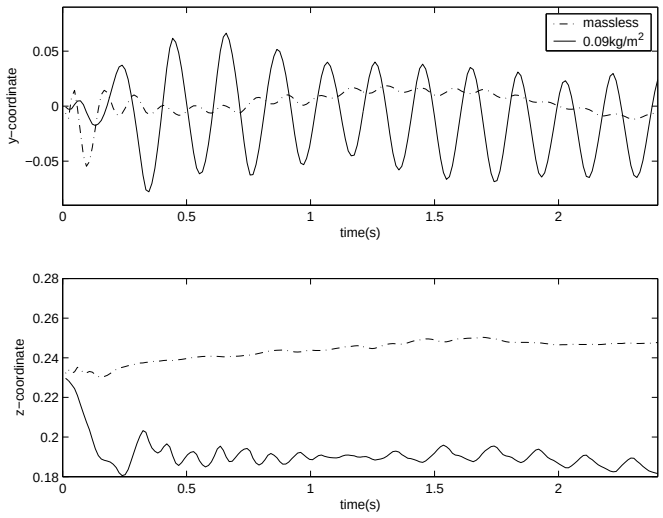

Fig. 20. Comparison of the motions of two flags with and without mass. Trajectories of the (y, z) coordinates of the lower downstream corners of the two flags are plotted, with the solid lines corresponding to the flag with mass, and the broken lines corresponding to the flag without mass. The massive flag flaps (top panel) and sags (bottom panel) but the massless flag does neither.

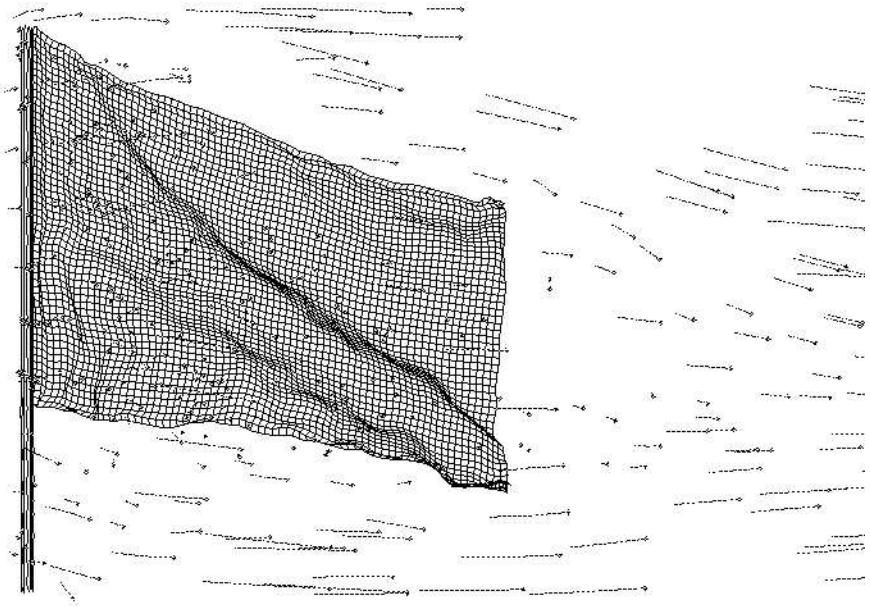

Fig. 21. Motion of a flag with a small wind speed. (time=2.4s)

that the downstream edge of the flag is somewhat S-shaped when viewed looking into the wind. See also the bottom graph of Fig. 22, which shows that the wind speed has essentially no effect on the sagging of the bottom of the flag even though the top of the flag sags more in the slow wind (non-flapping) case than in the fast wind case.

The final deficit case is created by reducing the initial perturbation to be 0.0025m. To demonstrate bistability, we choose the inflow velocity U_0=4.3m/s rather than 4.6m/s used so far. we can see from Figs. 18 and 23 that the flags with 0.005m initial perturbation both show sustained flapping motion even though they have different wind speeds of 4.6m/s and 4.3m/s, respectively. On the contrary, the case with the small initial perturbation (0.0025m amplitude) does not show the sustained flapping motion of the flag. This result is a reminiscent of the bistable motion of the filament in the Section 5.1, for which it was also the case that flapping or not depends on the amplitude of the initial perturbation. Note that the bottom graph in Fig. 23 is very similar

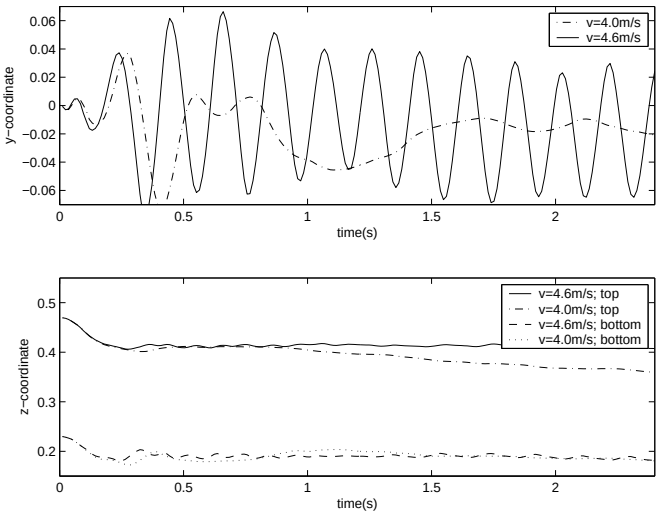

Fig. 22. Comparison of flag motions with low and high wind speed. The figure shows (y, z)-coordinates of the positions of selected points of the downstream edge of the two flags as functions of time: the inflow velocities are 4.0m/s and 4.6m/s. From the upper panel it is clear that the flag in the slower wind is not flapping although its position is not perfectly steady either. The heights of the downstream-bottom point of both flags is almost same, but the downstream-top point of the slower wind case is lower that of the faster wind case. This implies that the downstream edge of the flag in the slower wind is bent, which is a typical phenomenon when a flag does not flap, see also Fig. 21.

to that of Fig. 22. In fact, the detailed motion of the flag, when not flapping, is S-shaped when viewed looking into the wind, and it is almost same as in Fig. 21.

6 Summary and Conclusions

We have introduced a new version of the IB method which can handle a massive elastic boundary and have shown that the new method can be well applied to many problems in which mass of the immersed boundary is important. This

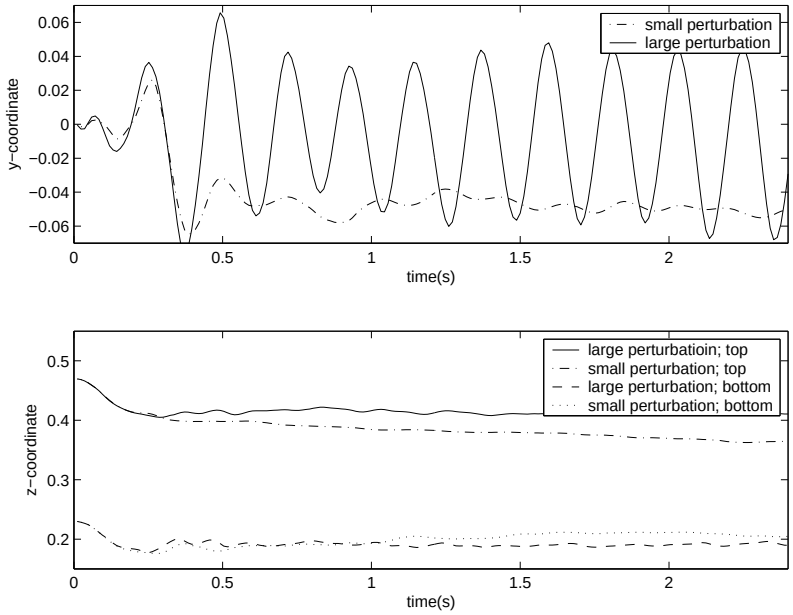

Fig. 23. Comparison of two flags with different initial perturbations. The figure shows (y, z)-coordinates of selected points of the downstream edge of two flags as functions of time. Everything about these two flags is identical except that the amplitudes of the initial perturbations are 0.0025m and 0.005m. For the sustained flapping motion, the flag needs a large enough initial perturbation. The bottom graph is very similar to that of Fig. 22. In fact, the specific motion of the flag, when not flapping, is S-shaped when viewed looking into the wind, and it is almost same as in Fig. 21.

method has the virtue of simplicity: one can easily implement it in the context of an existing IB method code for the massless case. The new method contains a penalty parameter, which is the stiffness coefficient of the springs that connect the massless immersed boundary points to their massive twins. In principle, this stiffness should be infinite, and one might think that it would have to be so large as to require a reduced time step for numerical stability, but in practice we have been pleased to find that this is *not* the case.

Since the purpose of this paper has been to introduce and illustrate the new

method, we have not yet pursued the applications begun here in as much detail as they deserve. The flapping flag, in particular, seems to have many aspects that are worthy of further study. For example, we can see from the flag simulation that whether a flag is flapping or not depends on several parameters: mass density of the flag itself, speed of the incident wind, and even the amplitude of the initial perturbation of the flag from a planar configuration. A detailed exploration of the behavior of the flag as a function of these parameters is needed.

Another subject for future research is a comparison of the three different methods that have now been proposed for the simulation of massive immersed boundaries within the framework of the IB method. These are the mass-spreading method [7,27,28], the direct use of the D'Alembert force [14], and the pIB method of the present paper. A preliminary comparison is presented in [14], but further work is needed to sort out the advantages and disadvantages of the different approaches.

Acknowledgement

We especially thank D.M.McQueen for kindly sharing experience and knowledge of various aspects of the IB method. We have made extensive use of his 3-D visualization software in carrying out this work. We also thank Luoding Zhu for helpful discussion about the flapping filament problem.

This research was supported by the National Science Foundation under KDI research grant DMS-9980069.

References

[1] S.S.Antman. *Nonlinear problems of elasticity.* Springer-Verlag 1995

[2] K.M.Arthurs, L.C.Moore, C.S.Peskin, E.B.Pitman, and H.E.Layton. *Modeling arteriolar flow and mass transport using the immersed boundary method.* J. Comput. Phys. 147:402-440, 1998

[3] R.P.Beyer. *A computational model of the cochlea using the immersed boundary method.* J. Comput. Phys. 98:145-162, 1992

[4] R.Cortez and M.Minion. *The Blob Projection Method for Immersed Boundary Problems.* J. Comput. Phys. 161:428-453, 2000

[5] A.L.Fogelson. *A mathematical model and numerical method for studying platelet adhesion and aggregation during blood clotting.* J. Comput. Phys. 56:111-134, 1984

[6] A.L.Fogelson and C.S.Peskin. *A fast numerical method for solving the three-dimensional Stoke's equations in the presence of suspended particles.* J. Comput. Phys. 79:50-69, 1988

[7] A.L.Fogelson and J.Jhu. *Implementation of variable-density immersed boundary method.* unpublished, http:/www.math.utah.edu/ fogelson

[8] L.J.Fauci and C.S.Peskin. *A computational model of aquatic animal locomotion.* J. Comput. Phys. 77:85-108, 1988

[9] E.Givelberg. *Modeling elastic shells immersed in fluid.* Ph.D.Thesis, Mathematics, New York University, 1997.

[10] B.E.Griffith and C.S.Peskin. *On the order of accuracy of the immersed boundary method: Higher order convergence rates for sufficiently smooth problems.* J. Comput. Phys. 208:75-105, 2005

[11] E.Jung and C.S.Peskin. *Two-dimensional simulations of valveless pumping using the immersed boundary method.* SIAM J.Sci.Comput. 23:19-45, 2001

[12] Y.Kim. *The Penalty Immersed Boundary Method and its Applications to Aerodynamics.* Ph.D.Thesis, Mathematics, New York University, 2003.

[13] Y.Kim and C.S.Peskin. *2-D parachute simulation by the Immersed Boundary Method.* preprint. submitted to SIAM journal on Scientific Computing.

[14] Y.Kim, L.Zhu, X.Wang, and C.S.Peskin. *On various techniques for computer simulation of boundaries with mass.* In: Computational Fluid and Solid Mechanics: Proceedings Second M.I.T. Conference on Computational Fluid and Solid Mechanics, June 17-20 (Bathe KJ, ed.), Elsevier, 2003, pp. 1746-1750.

[15] M.C.Lai and C.S.Peskin. *An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity.* J. Comput. Phys. 160:705-719, 2000

[16] L.Lee and R.J.Leveque. *An Immersed Interface Method for Incompressible Navier-Stokes Equations.* SIAM J.Sci.Comput. 25(3):832-856, 2003

[17] H.N.Ng and R.L.Grimsdale. *Computer Graphics Techniques for Modeling Cloth.* IEEE Computer Graphics and Applications 16(5):28-41, 1996

[18] C.S.Peskin. *Numerical analysis of blood flow in the heart.* J. Comput. Phys. 25:220-252, 1977

[19] C.S.Peskin. *The Immersed Boundary Method.* Acta Numerica, 11:479-517, 2002

[20] C.S.Peskin and D.M.McQueen. *Heart Simulation by an Immersed Boundary Method with Formal Second-order Accuracy and Reduced Numerical Viscosity.* In: Mechanics for a New Millennium, Proceedings of the International Conference on Theoretical and Applied Mechanics(ICTAM) 2000, (H.Aref and J.W.Phillips,eds.)Kluwer Academic Publishers,2001

[21] C.S.Peskin and D.M.McQueen. *Three dimensional computational method for flow in the heart :Immersed elastic fibers in a viscous incompressible fluid.* J. Comput. Phys. 81:372-405, 1989

[22] C.S.Peskin and D.M.McQueen. *A general method for the computer simulation of biological systems interacting with fluids.* Symposia of the society for Experimental Biology. 49:265-276, 1995

[23] C.S.Peskin and D.M.McQueen. *Fluid dynamics of the heart and its valves. In: Case studies in Mathematical Modeling: Ecology, Physiology, and Cell Biology.* Prentice Hall, Englewood Cliffs NJ, 1996, pp. 309-337

[24] C.Tu and C.S.Peskin. *Stability and instability in the computation of flows with moving immersed boundaries: a comparison of three methods.* SIAM J.SCI.STAT.COMPUT, vol 13, 6:1361-1376, 1992

[25] C.S.Peskin and B.F.Printz. *Improved volume conservation in the computation of flows with immersed elastic boundaries.* J.Comput. Phys. 105:33-46, 1993

[26] J.Zhang, S.Childress, A.Libchaber, and M.Shelley. *Flexible filaments in a flowing soap film as a model for one-dimensional flags in a two-dimensional wind.* Nature 408, 835, 2000

[27] L.Zhu. *Simulation of a flapping flexible filament in a flowing soap film by the Immersed Boundary method.* Ph.D.Thesis, Mathematics, New York University, 2001.

[28] L.Zhu and C.S.Peskin. *Simulation of a flapping flexible filament in a flowing soap film by the Immersed Boundary method.* J.Comput. Phys. 179:452-468, 2002