

This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

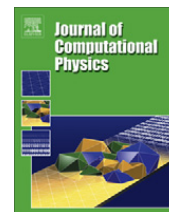
In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

<http://www.elsevier.com/copyright>



Contents lists available at ScienceDirect

## Journal of Computational Physics

journal homepage: [www.elsevier.com/locate/jcp](http://www.elsevier.com/locate/jcp)

## A multiresolution remeshed Vortex-In-Cell algorithm using patches

Johannes Tophøj Rasmussen<sup>a</sup>, Georges-Henri Cottet<sup>b</sup>, Jens Honoré Walther<sup>a,c,\*</sup><sup>a</sup>Department of Mechanical Engineering, Technical University of Denmark, Building 403, DK-2800 Kgs. Lyngby, Denmark<sup>b</sup>Laboratoire Jean Kuntzmann, Université Joseph Fourier, 51 rue des Mathématiques, 38041 Grenoble Cedex 9, France<sup>c</sup>Computational Science and Engineering Laboratory, ETH Zürich, Universitätsstrasse 6, CH-8092 Zürich, Switzerland

## ARTICLE INFO

## Article history:

Received 20 December 2010

Received in revised form 6 April 2011

Accepted 5 May 2011

Available online 14 May 2011

## Keywords:

Vortex-In-Cell algorithm

Multiresolution

Patches

Brinkman penalization

Fixed and rotating cylinder

Bridge aerodynamics

## ABSTRACT

We present a novel multiresolution Vortex-In-Cell algorithm using patches of varying resolution. The Poisson equation relating the fluid vorticity and velocity is solved using Fast Fourier Transforms subject to free space boundary conditions. Solid boundaries are implemented using the semi-implicit formulation of Brinkman penalization and we show that the penalization can be carried out as a simple interpolation. We validate the implementation against the analytic solution to the Perlman test case and by free-space simulations of the onset flow around fixed and rotating circular cylinders and bluff body flows around bridge sections.

© 2011 Elsevier Inc. All rights reserved.

## 1. Introduction

The Vortex-In-Cell (VIC) method [1,2] is a highly efficient hybrid particle-mesh algorithm for the simulation of incompressible viscous flow. The alternative, the fast multipole method (FMM) [3–5] scales optimally as  $\mathcal{O}(N)$ , where  $N$  denotes the number of particles, but involves a bigger prefactor than the VIC algorithm, and parallel implementations of the FMM are generally less efficient than VIC algorithms [6]. Another efficient hybrid scheme [7–10] decomposes space into smaller VIC problems with boundary conditions provided by an FMM. The VIC algorithm has previously been implemented in parallel [11] to simulate large flow problems, e.g. trailing vortices with billions of particles [12]. By using Brinkman penalization straightforward fluid-solid interaction can be included without the need of complex meshing [13]. Particle velocities are computed on meshes, using fast Fourier transforms (FFT) with a  $\mathcal{O}(N \log N)$  scaling. Non-periodic [14–16] or semi-periodic [17] flows can be simulated in spite of the periodic nature of the FFT algorithm. However, the FFTs require Cartesian uniform meshes. While these meshes allow efficient computation of vortical diffusion and stretching they do not facilitate local refinement.

Non uniform resolution in particle methods has been achieved with conform mappings [6,18], Dirichlet boundary conditions [19,20] and FMM multilevel particle remeshing algorithms [21]. The latter reference, which introduces overlaid meshes and overlapping buffers, serves as the starting point for the present multiresolution mesh based velocity algorithm. In the present work the meshes are staggered and the buffers are adapted accordingly, thereby alleviating the discrete sub-sectioning. The local refinement is determined a priori in the current work, but the presented algorithm does not prevent the formation or repositioning of the refined areas during the simulation. The present work is limited to two dimensions but can

\* Corresponding author at: Department of Mechanical Engineering, Technical University of Denmark, Building 403, DK-2800 Kgs. Lyngby, Denmark. Tel.: +45 45254327; fax: +45 4588 4325.

E-mail addresses: [jhw@mek.dtu.dk](mailto:jhw@mek.dtu.dk), [walther@inf.ethz.ch](mailto:walther@inf.ethz.ch) (J.H. Walther).

easily be extended to three dimensions and is relevant for simulating physics governed by the Poisson equation, e.g. celestial mechanics, electro dynamics.

The remaining part of the paper is structured as follows: in Section 2 we present the flow model and the penalization technique. In Section 3 we describe our multiresolution algorithm. In Section 4 we validate the implementation of the algorithm against quantitative benchmarks. In Section 5 we conclude the article.

## 2. The vortex in cell algorithm

The Vortex-In-Cell algorithm [1,2] evolves the incompressible flow described by the vorticity-transport equation

$$\frac{D\omega}{Dt} = \nu \nabla^2 \omega, \quad (1)$$

with constant kinematic viscosity  $\nu$ . Eq. (1) describes the rate of change of vorticity

$$\omega = \nabla \times \mathbf{u}, \quad (2)$$

of a particle moving with the velocity  $\mathbf{u}$ . The vorticity in two-dimensional analysis is a scalar  $\omega$  equal to the out of plane component ( $\mathbf{e}_z$ ) of the curl of the velocity field,  $\omega = \omega \mathbf{e}_z$ . The velocity  $\mathbf{u}$  is defined as

$$\mathbf{u} = \nabla \times \Psi + \mathbf{U}_\infty, \quad (3)$$

where  $\Psi$  is a stream function and  $\mathbf{U}_\infty$  the irrotational free stream velocity. Combining Eq. (3) with the definition of vorticity (Eq. (2)) yields the Poisson equation

$$\nabla^2 \Psi = -\omega, \quad (4)$$

linking vorticity to the stream function and in turn to the velocity, cf. Eq. (3). The stream function is calculated on the mesh from the Poisson equation, Eq. (4), using a Green's function solution [15]

$$\Psi(\mathbf{x}) = \int_V G(\mathbf{y} - \mathbf{x}) \omega(\mathbf{x}) d\mathbf{y}, \quad (5)$$

$$\Psi = G * \omega, \quad (6)$$

where

$$G(\mathbf{y} - \mathbf{x}) = -\frac{1}{2\pi} \log |\mathbf{y} - \mathbf{x}|$$

is the free-space Green's function to the Poisson equation in two dimensions [22]. The convolution is performed in Fourier space according to the convolution theorem. The Fourier transformation is carried out numerically using the fast Fourier transform which entails a periodicity of  $G$ ,  $\omega$  and in turn  $\Psi$ . To circumvent the periodicity the domain is doubled in each physical direction by padding the vorticity field  $\omega$  by zeros [14–16].  $G$  is defined in the full extent of the new domain. Due to the zero-vorticity region the Green's function does not add periodic contributions through the convolution.

The vorticity field  $\omega$  is approximated by  $N$  discrete vorticity-carrying particles. The particle vorticity is interpolated to the mesh points  $\mathbf{x}_m$  of an underlying equally spaced Cartesian mesh, through

$$\omega_m = \sum_p^N \omega_p W\left(\frac{\mathbf{x}_p - \mathbf{x}_m}{\Delta x}\right), \quad (7)$$

using the third order  $M'_4$  [23] interpolation kernel

$$W(x) = \begin{cases} 0, & \text{for } |x| > 2, \\ \frac{1}{2}(2 - |x|)^2(1 + |x|), & \text{for } 2 \geq |x| > 1, \\ 1 - \frac{5}{2}x^2 + \frac{3}{2}|x|^3, & \text{for } 1 \geq |x|, \end{cases} \quad (8)$$

in each coordinate direction, conserving 0, 1st and 2nd order moments.  $\Delta x$  is the mesh spacing.

From the velocity  $\mathbf{u}$  and vorticity  $\omega$  on the mesh the right hand side of Eq. (1) is computed and interpolated back to the particles along with the velocity. The particle position  $\mathbf{x}_p$  and vorticity  $\omega_p$  is updated using Runge–Kutta second order time integration.

For the simulation of advection problems, particle-mesh algorithms are not constrained with the usual CFL condition. Instead, the time step size ( $\Delta t$ ) is limited by the amount of shear, through a condition of the type

$$\Delta t < \frac{C_1}{|\omega|_\infty},$$

which does not directly depend on the spatial resolution.  $C_1$  is a constant depending on the integration scheme. In the cases we have dealt with in the present paper, the limiting factor is the explicit discretization of the diffusion and our time step is chosen to satisfy

$$\Delta t < \frac{C_2 \Delta x^2}{\nu}.$$

$C_2 = 1/4$  is a constant depending on the integration scheme.

As the particles are advected they may be subject to numerical errors imposing compression/dilation in certain regions whereas other regions become sparsely populated. To ensure a sufficient resolution in the entire domain and to avoid strong deformation of the particle volumes new particles are remeshed from the mesh. Remeshing at every  $n_r$  time step improves spatial convergence [24]. Presently  $n_r = 1$ .

### 2.1. Brinkmann penalization

The no-slip condition at solid surfaces may be imposed by using Brinkman penalization [25,26,13]. By adding the Brinkmann penalization term to the Navier–Stokes equation, Eq. (1) becomes

$$\frac{D\omega}{Dt} = \nu \nabla^2 \omega + \nabla \times [\lambda \chi (\mathbf{u}_s - \mathbf{u})], \quad (9)$$

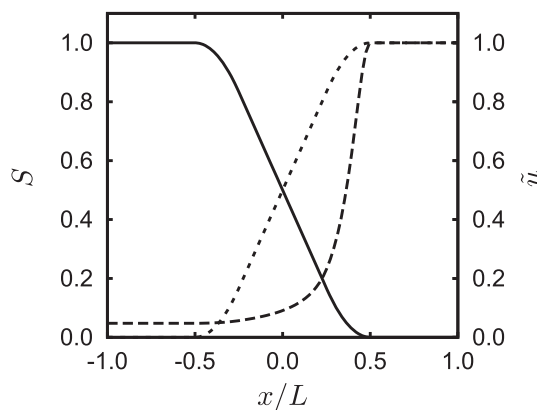
where  $\lambda$  is the penalization parameter ( $[1/s]$ ), equivalent of an inverse porosity and  $\mathbf{u}_s$  is the solid velocity.  $\chi$  is the solid mask that is 1 inside the solid and 0 in the fluid. In the present work  $\chi$  is defined solely on the meshes. To improve the numerical accuracy of the curl,  $\chi$  is varied smoothly from 0 to 1 over the mollification interval of width  $L$  normal to the surface, corresponding to a fixed number of mesh cells.  $\chi$  is constructed via a mollified step function  $S$ , a function of the signed distance to the surface of the solid. In the present work  $S$  is determined such that it is continuous and differentiable and consists of second order polynomials in the first and last  $L/4$  of the mollification interval and first order polynomial in the intermediate region, see Fig. 1. It is desirable to choose  $\lambda T \gg 1$ , where  $T$  is a flow characteristic time, but, in the case of explicit Euler time discretization the penalization parameter must satisfy  $\lambda \Delta t < \mathcal{O}(1)$  to ensure stability. Instead of evaluating the penalization as a source term, cf. Eq. (9), it can be evaluated semi-implicitly [13] by using a split-step algorithm. That is, at time step  $n$ : update the vorticity field by evaluating the penalization term and subsequently evaluate the right hand side of Eq. (1). The penalization term can be evaluated implicitly by the expression

$$\tilde{\mathbf{u}}^{n+1} = \frac{\mathbf{u}^n + \lambda \Delta t \chi \mathbf{u}_s^n}{1 + \lambda \Delta t \chi}, \quad (10)$$

$$\tilde{\omega}^{n+1} = \nabla \times \tilde{\mathbf{u}}^{n+1}, \quad (11)$$

where the tilde denotes that the result is not the final solution of the time step. This facilitates the use of high  $\lambda$  values but with an increasingly discontinuous velocity field as a consequence when using a smooth step function  $S$ . Also, a residual of  $\mathbf{u}^n$  on the order of  $1/\lambda \Delta t$  is left in the solid interior, cf. Fig. 1. Instead the penalization term may be evaluated explicitly in a similar split-step algorithm with  $\lambda = 1/\Delta t$ . The resulting expression may be viewed as simple interpolation between the velocity fields  $\mathbf{u}^n$  and  $\mathbf{u}_s^n$

$$\tilde{\mathbf{u}}^{n+1} = (1 - \chi) \mathbf{u}^n + \chi \mathbf{u}_s^n, \quad (12)$$



**Fig. 1.** The polynomial step function  $S$  (—) as a function of the signed distance to the solid surface. Implicit penalization of a uniform velocity field  $U = 1$  (Eq. (10)) results in velocity fields  $\tilde{u}$  (----) with a residual of  $1/\lambda \Delta t$  of  $U$  in the solid. (-----) is the velocity by interpolation penalization Eq. (12). For the sake of clarity  $\lambda \Delta t = 20$ .

thereby eliminating the penalization parameter  $\lambda$ , removing the residual velocity and controlling the weight of the solid and fluid velocity explicitly. The *interpolating penalization* Eq. (12) also corresponds to the semi-implicit penalization (Eq. (10)) using a special mask

$$\tilde{\chi} = \min \left( \frac{\chi}{\lambda \Delta t (1 - \chi)}, 1 \right),$$

that is capped at 1 to avoid singularities, where  $\chi$  is the corresponding mask of Eq. (12).

Replacing the vorticity field by the finite difference evaluation of Eq. (11) introduces significant numerical diffusion of the vorticity field. Therefore the penalization step Eqs. (12) and (11) is formulated as a correction

$$\begin{aligned} \Delta \mathbf{u} &= \tilde{\mathbf{u}}^{n+1} - \mathbf{u}^n = (1 - \chi) \mathbf{u}^n + \chi \mathbf{u}_s^n - \mathbf{u}^n \\ &= \chi (\mathbf{u}_s^n - \mathbf{u}^n), \\ \Delta \omega &= \nabla \times \Delta \mathbf{u}, \end{aligned} \quad (13)$$

$$\tilde{\omega}^{n+1} = \omega^n + \Delta \omega, \quad (14)$$

such that  $\omega^n$  is not replaced but updated by the correction Eq. (13) evaluated by finite differences.

The vorticity correction (Eq. (14)) is local and therefore the velocity field computed from the corrected vorticity can in general not be expected to provide a solution that exactly satisfies the no-slip boundary condition. This can be improved by repeating the penalization during a time step thereby increasing the correspondence between velocity and vorticity. We have observed that the deviation from the penalized velocity to the solid velocity decreases exponentially. In the present simulations the solid velocities are constant and, thus the effect of performing more than one penalization per time step is insignificant. In the case of objects moving or deforming objects consecutive penalizations may be necessary.

## 2.2. Aerodynamic forces

The two-dimensional (2D) force per unit length applied by the solid to the fluid and vice versa can be computed by integrating the penalization term over the domain. For the explicit evaluation of the penalization term the force is given by

$$\mathbf{F}_s = -\rho \int_A \lambda \chi (\mathbf{u}_s - \mathbf{u}) dA, \quad (15)$$

or, in the case of split-step interpolation penalization, by integrating the time averaged rate of change of the interpolation penalization

$$\mathbf{F}_s = -\rho \int_A \chi \frac{\mathbf{u}_s^n - \mathbf{u}^n}{\Delta t} dA. \quad (16)$$

Alternatively the force on stationary objects can be calculated by the rate of change of the integral of vorticity moments [27] over the domain  $A$

$$\mathbf{F}_s = -\rho \frac{d}{dt} \int_A \mathbf{x}_m \times \omega_m \mathbf{e}_z dA, \quad (17)$$

where  $\rho$  is the fluid density. The expression can be expanded to domains with a flux of vorticity across the boundaries [28] and to allow general solid body motion of the objects [29].

## 3. Multiresolution

Inherently, the Vortex-In-Cell algorithm is limited by the required uniform mesh resolution. Particles of different size can be initialised to increase the spatial resolution where needed but to be able to use FFTs to efficiently solve the Poisson equation, the underlying mesh must be uniform thus loosing the added spatial resolution when computing particle velocities and when remeshing. By utilising the linearity of the Poisson equation (Eq. (4)) it is possible to increase the local resolution on patches, i.e. to have local meshes of higher resolution conforming to the requirements of the FFT.

By  $\bar{\omega}$  we define the complete vorticity field and by  $\bar{\Psi}$  and  $\bar{\mathbf{u}}$  the corresponding stream function and velocity field on the domain  $A$ . We compose  $A$  into  $A_1$  and  $A_2$  where

$$\begin{aligned} A &= A_1 \cup A_2, \\ A_1 \cap A_2 &= \emptyset. \end{aligned}$$

We similarly divide the complete vorticity  $\bar{\omega}$  into a part  $\omega_1$  and  $\omega_2$  contained in  $A_1$  and  $A_2$  respectively such that

$$\bar{\omega} = \omega_1 + \omega_2.$$

Due to the linearity of the Poisson equation Eq. (4) it follows that

$$\bar{\Psi} = \Psi_1 + \Psi_2, \quad (18)$$

where

$$\nabla^2 \Psi_1 = -\omega_1,$$

$$\nabla^2 \Psi_2 = -\omega_2,$$

are the stream functions on the domain  $A$  corresponding to the vorticity  $\omega_1$  and  $\omega_2$ . Due to the linearity of the curl operator it follows that

$$\bar{\mathbf{u}} = \mathbf{u}_1 + \mathbf{u}_2, \quad (19)$$

where

$$\mathbf{u}_1 = \nabla \times \Psi_1,$$

$$\mathbf{u}_2 = \nabla \times \Psi_2.$$

The numerical representation of  $\omega_1$  and  $\omega_2$  is contained on rectangular uniform Cartesian meshes  $M'_1$  and  $M'_2$  covering at least  $A_1$  and  $A_2$  respectively. The primes ' and ' ' indicate mesh spacings  $\Delta x_1$  and  $\Delta x_2$ . If  $\Delta x_1 \neq \Delta x_2$  Eqs. (18) and (19) are evaluated using mesh-to-mesh interpolation using Eq. (7), where  $(\omega_p, \mathbf{x}_p)$  is replaced by the mesh points and corresponding vorticity values. By  $\omega''$  we denote vorticity on a mesh with a resolution higher than that of a mesh containing  $\omega'$ . In the present work the resolution of  $\omega''$  is twice that of  $\omega'$ , i.e.  $\Delta x_2 = \Delta x_1/2$ . Similarly  $\mathbf{u}''$  denotes velocity at twice the resolution of  $\mathbf{u}'$ .

The points of the 2D meshes are interpreted as the centres of equally sized rectangular cells. We require that the boundaries of  $A_1$  and  $A_2$  are aligned with the cell sides to ensure a unique representation of the vorticity  $\omega_1$  and  $\omega_2$  and the conservation of the total vorticity, i.e. circulation. This implies that the mesh points of the two meshes are staggered in each coordinate-direction by  $\Delta x_2/2$ .  $A_1$  and  $A_2$  are shown as dark areas on Fig. 2.

It is possible to calculate the velocity field  $\mathbf{u}$  on both the coarse  $M'_1$  and the finer  $M'_2$  taking advantage of the increased resolution on  $M'_2$ . We refer to the ensemble of the mesh  $M_2$  and the set of particles created from it as a *patch* and refer to  $M_1$  and the set of particles created from  $M_1$  as the *parent*. The patch will be specified further in Section 3.1. By interpolating  $\omega'_2$  to  $M'_1$  the mesh contains a complete representation of the vorticity field

$$\bar{\omega}'_1 = \omega'_1 + \omega'_2 \quad (20)$$

and the velocity  $\bar{\mathbf{u}}'_1$  can be determined using the standard free-space Green's function approach on the uniform  $M'_1$ . The bar denotes that the velocity field corresponds to the complete vorticity and the lower index indicates the resolution level.

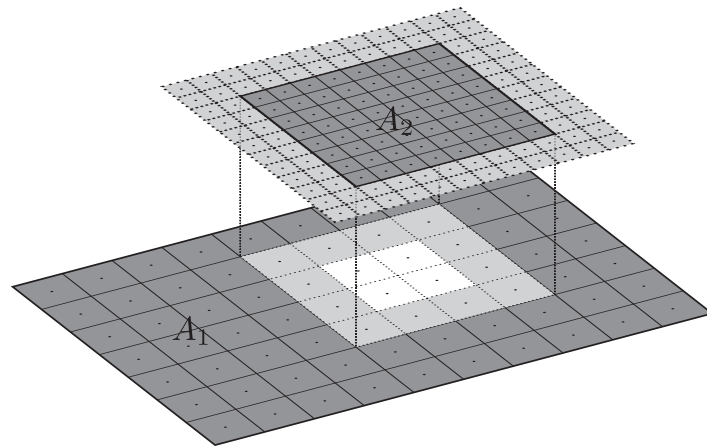
For  $M'_2$  the velocity contribution of both  $\omega_1$  and  $\omega_2$  must be included. To obtain the velocity contribution  $\mathbf{u}'_1$  from the exterior vorticity  $\omega_1$ , the velocity contribution  $\mathbf{u}'_2$  of the coarse patch vorticity  $\omega'_2$  is subtracted from  $\bar{\mathbf{u}}'_1$ . The result is then interpolated to  $M'_2$

$$(\mathbf{u}'_1)'' = (\bar{\mathbf{u}}'_1 - \mathbf{u}'_2)'' = (\bar{\mathbf{u}}'_1)'' - (\mathbf{u}'_2)'',$$

yielding the contribution of the exterior vorticity on  $M'_2$ . The final complete velocity is given by adding  $\mathbf{u}''_2$ , the velocity contribution of  $\omega'_2$ , to  $(\mathbf{u}'_1)''$

$$\bar{\mathbf{u}}''_2 = (\mathbf{u}'_1)'' + \mathbf{u}''_2 = (\bar{\mathbf{u}}'_1)'' - (\mathbf{u}'_2)'' + \mathbf{u}''_2. \quad (21)$$

Again the bar denotes that the velocity corresponds to the complete vorticity. If multiple patches share the same parent mesh  $M_1$  the velocity field  $\bar{\mathbf{u}}'_1$  can be reused. This is not the case if  $\mathbf{u}'_1$  is computed directly.



**Fig. 2.** Meshes: The dark areas are the patch interior mesh  $A_2$  and the parent mesh  $A_1$ , the light grey areas are the corresponding buffers. The cell centres have been marked by dots.

### 3.1. Patches and interpolation

The present approach builds on the work of [21] and thereby differs from other multiresolution particle methods as the set of all particles is divided into subsets, each subset being associated with a certain mesh. In the present work nodes of meshes of varying resolution are not aligned as [21] but staggered to ensure conservation of vorticity. The combination of a mesh and the corresponding set of particles we refer to as a patch. Each patch has a parent; a mesh and its associated set of particles, and the entire data structure consists of patches. The parent can be another patch of any resolution level or it may be the lowest level mesh and its associated particles. The mesh that covers the entire computational domain and its associated particles are level 1 and are referred to as the base patch. As the base patch does not have a parent patch, parts of the interpolation and velocity computation vary for the base patch. The resolution of the  $l$ th level patch is given relative to the base mesh spacing  $\Delta x_1$ . Thus

$$\Delta x_l = \Delta x_1 \left( \frac{1}{2} \right)^{l-1},$$

for any  $l > 0$ . In the present implementation a patch of level  $l > 1$  has a parent of level  $l - 1$ . By extending the interpolation, patches could have parents of even coarser resolution. For each level there may be several patches with the same or different parents and we refer to the  $p$ th patch of level  $l$  as  $P_l^p$ . To ensure spatial support for the patch particle-to-mesh interpolation (P2M) (Eq. (7)), particles are formed from both the interior patch as well as a surrounding support buffer [21].  $k$  denotes the width of the support; for the  $M_4'$  kernel  $k = 2$  cf. Eq. (8). These particles are created during the particle remeshing step to ensure support for the  $M_4'$  kernel for the interior mesh points close to the boundary. Fig. 3 shows the discrete support buffers for the staggered setup of the patch and the parent mesh

$$l_{s1} = k\Delta x_1, \quad (22)$$

$$l_{s2} = k\Delta x_2. \quad (23)$$

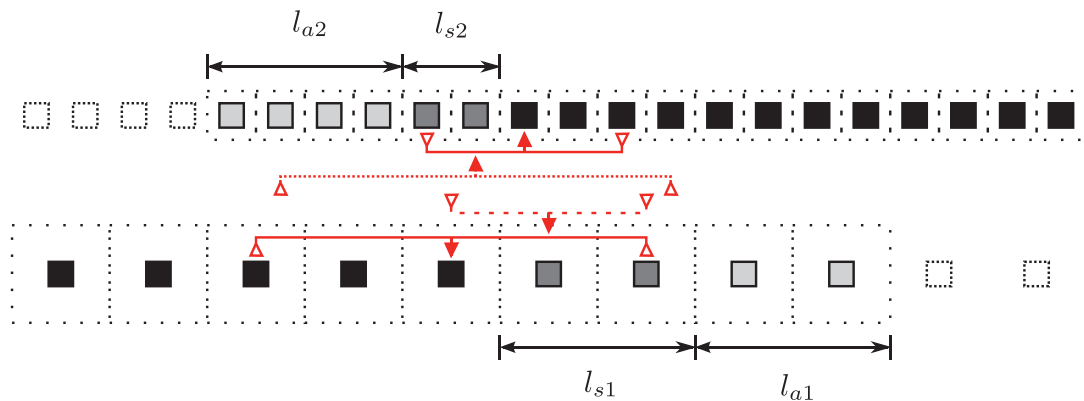
When the particles are advected after the remeshing at every  $n_r$  time steps, the interpolation support of the outermost mesh points is ensured by creating additional particles in the advection buffer [21] of width

$$l_{a1} = \sum_{n=1}^{n_r} \|u^n\|_{\infty} \Delta t, \quad (24)$$

$$l_{a2} = \sum_{n=1}^{n_r} \|u^n\|_{\infty} \Delta t. \quad (25)$$

The advection buffer is truncated to the lowest multiple of the cell length. To evaluate the right hand side of Eq. (1) with second order finite differences the buffer must be either expanded additionally by one cell for support of the finite difference scheme or, as in the present study, through one-sided finite differences for the outermost layer of cells.

The present mesh and buffer setup ensures support for the mesh values of the patch buffer such that values can be interpolated from the particles of the parent, and vice versa, as shown by the dashed and dotted arrows on Fig. 3. Thereby the meshes, including buffers, are populated entirely by particle-to-mesh interpolation. In two dimensions the buffers enclose the patch interior as shown on Fig. 2. We bring to mind that vorticity is only considered unique in the patch interior, i.e. excluding buffers, and the part of the parent mesh that is not covered by the patch interior.



**Fig. 3.** Sketch of the patch/parent staggered meshes at patch boundary. The dashed lines represent the volumes corresponding to the mesh points. The solid black squares are the unique patch interior mesh points and unique points of the parent mesh. Solid dark/light grey points are support/advection buffer points. Solid arrows indicate the required support of the interpolation kernel and dashed/dotted arrows indicate the required support for interpolating the patch/parent particles to the patch/parent buffers.



### 3.1.1. Mesh-to-mesh interpolation in the buffer

In the present work the value of the buffer mesh points are obtained by mesh-to-mesh (M2M) interpolation. By extending the patch support buffer both parent and patch buffer mesh values can be obtained entirely through mesh-to-mesh interpolation after particle values have been interpolated to the unique interior points. This reduces the computational overhead for the structured meshes. To ensure support for the  $M^4$  kernel the patch buffer  $l_{s2} + l_{a2}$  is increased by one mesh point as indicated by the dashed arrow on Fig. 3 and  $l_{a2}$  is truncated to nearest multiple of  $\Delta x_2$ . This extension of the buffer has a beneficial side effect when dealing with discontinuities at the patch boundary as described in Section 4.1. After performing particle-to-mesh interpolation to the unique vorticity regions of the patch and parent the patch vorticity is interpolated not just to the parent buffer. Instead the entire patch vorticity is interpolated to the parent mesh corresponding to Eq. (20). Finally the patch buffer is filled by mesh-to-mesh interpolation from the parent mesh. Alternatively the parent buffer can be similarly extended but resulting in severe limitations when adjacent patches exist on the same level.

### 3.1.2. Upgrading neighbour buffer

Adjacent patches have an overlap of the interior and the buffer of the neighbouring patch. After particle-to-mesh interpolation the buffer vorticity is *upgraded*, i.e. copied from the neighbouring patch to the buffers, where possible. Thereby the existing buffer vorticity, stemming from the lower resolution parent mesh, is upgraded by copying the same resolution neighbour vorticity. After the velocity has been computed on all patches and levels velocity is upgraded as well. This increases the direct connection between patches.

In case that the vorticity cannot be upgraded from neighbouring patches the coarser resolution vorticity interpolated from the parent is left in the buffer, which in turn is advected into the patch interior. The coarser incoming buffer vorticity is sufficient under the assumption that the complexity of the flow is formed on patch levels of higher resolution.

### 3.1.3. Multiresolution and penalization

Penalization is carried out entirely on the meshes. Since the width of the mollification interval  $L$  and in turn  $S$  and  $\chi$  depend on the mesh resolution, penalization is carried out only on patches of same resolution in the present implementation. The resulting vorticity is interpolated and copied to the other levels. Alternatively  $L$  could be geometrically fixed causing the mollification interval to either span a large number of cells at the highest level or being under-resolved on the lowest level. However, the error of the finite difference approximation introduces a discrepancy of the vorticity of levels of different resolution.

The force calculated from Eq. (16) is thus only integrated on the level where penalization is performed. When calculating the force from the vorticity moments Eq. (17) only unique vorticity is included. That is, vorticity is only included once; the version of highest resolution and not the lower resolution vorticity of parents.

## 3.2. Vortex-In-Cell algorithm using patches

The vorticity is initialised on the patch particles. In the time integration loop the particle vorticity is interpolated to the meshes. First particle values are interpolated to their patch mesh. The patches are traversed from the highest (finest) level down to level 2, copying and interpolating the patch mesh vorticity to the parent mesh. Then, traversing the patches from level 2 to the highest level, the parent mesh vorticity is interpolated and copied to the patch buffer. The velocity on level 1 is calculated by the standard free-space VIC method. Going from level 2 to the highest level the velocity is calculated as described in Section 3. Interpolation penalization is carried out on the levels where  $\chi \neq 0$ , vorticity is updated and the velocity is recalculated from the updated vorticity field. Particles are remeshed, i.e. new particles are formed at the mesh nodes, and subsequently advected and updated due to viscous diffusion. This is done by interpolating the mesh based velocities and  $\frac{D\omega}{Dt}$  to the particles. The advection and particle diffusion may be applied in one (Euler) step or by a higher order integration scheme. In the latter case the particle vorticity is interpolated to the mesh, velocities and  $\frac{D\omega}{Dt}$  are calculated and interpolated back to the particle positions in each sub-step. More explicitly:

### Initialisation

Create particles on all patches  $P_i^p$  carrying the initial vorticity field

### START TIME INTEGRATION LOOP:

#### Particle to mesh interpolation

For  $l$  from 1 to  $l_{\max}$

and  $p$  from 1 to  $p_{\max}(l)$

- Interpolate the particle vorticity to the corresponding mesh ( $\mathcal{O}(N_i^p)$ )

For  $l$  from  $l_{\max}$  to 2

For  $p$  from 1 to  $p_{\max}(l)$



- Interpolate and copy the mesh vorticity to the parent mesh (M2M) hereby also filling the buffer ( $\mathcal{O}(N_l^p)$ )

For  $l$  from 2 to  $l_{\max}$

For  $p$  from 1 to  $p_{\max}(l)$

- Interpolate and copy the parent mesh vorticity to the patch buffer (M2M) ( $\mathcal{O}(N_l^p)$ )

Upgrade the buffer vorticity for adjacent patches. The vorticity is completely defined on all meshes

### Velocity computation

On  $P_1^1$  calculate the velocity  $\bar{\mathbf{u}}_1^1$  by standard free-space VIC procedure ( $\mathcal{O}(N_1^p \log(N_1^p))$ )

For  $l$  from 2 to  $l_{\max}$

For  $p$  from 1 to  $p_{\max}(l)$

- Interpolate the coarser velocity  $\bar{\mathbf{u}}_{l-1}^d$  to  $M_l^p$  where  $P_{l-1}^d$  is the parent of patch  $P_l^p$  ( $\mathcal{O}(N_l^p)$ )
- Interpolate  $\omega_l^{p''}$  to  $\omega_l^{p'}$  ( $\mathcal{O}(N_l^p)$ )
- Compute  $\mathbf{u}_l^{p'}$  and  $\mathbf{u}_l^{p''}$  ( $\mathcal{O}(N_l^p \log(N_l^p))$ )
- Compute  $\bar{\mathbf{u}}_l^p$  cf. Eq. (21) ( $\mathcal{O}(N_l^p)$ )

Upgrade the buffer velocity for adjacent patches

### Penalization

For  $l$  from 1 to  $l_{\max}$

For  $p$  from 1 to  $p_{\max}(l)$

- Impose interpolation penalization cf. Eqs. (12) and (13) on the relevant levels ( $\mathcal{O}(N_l^p)$ )

For  $l$  from  $l_{\max}$  to 1

For  $p$  from 1 to  $p_{\max}(l)$

- Interpolate and copy the updated vorticity to the parent meshes ( $\mathcal{O}(N_l^p)$ )
- Create new remeshed particles from the penalized vorticity ( $\mathcal{O}(N_l^p)$ )

Recalculate the velocity from vorticity ( $\mathcal{O}(N_l^p \log(N_l^p))$ )

### Vorticity equation

For  $l$  from 1 to  $l_{\max}$

For  $p$  from 1 to  $p_{\max}(l)$

- Compute  $\frac{D\omega}{Dt}$  on the mesh cf. Eq. (1) ( $\mathcal{O}(N_l^p)$ )

### Mesh to particle interpolation

For  $l$  from 1 to  $l_{\max}$

For  $p$  from 1 to  $p_{\max}(l)$

- Interpolate the velocity and  $\frac{D\omega}{Dt}$  to the particles ( $\mathcal{O}(N_l^p)$ )

### Particle position and vorticity update

For  $l$  from 1 to  $l_{\max}$

For  $p$  from 1 to  $p_{\max}(l)$

- Update the particle positions and vorticity values ( $\mathcal{O}(N_l^p)$ )

### END OF TIME INTEGRATION LOOP

If the time integration of the particle positions and vorticity is carried out using a higher order integration method the particle-to-mesh interpolation, velocity computation, vorticity equation and mesh-to-particle interpolation steps are carried out for each sub-step of the scheme.

## 4. Validation and results

The presented multiresolution velocity algorithm is validated against the Perlman vorticity patch [30]. To validate the interpolation penalization, the impulsively started flow around a circular cylinder is simulated. The drag and lifting forces are investigated for both static and rotating cylinders. Also the flow around the Great Belt East suspension bridge deck is simulated and compared to solutions of other numerical methods.

#### 4.1. Perlman vorticity patch

To test the calculation of the fluid velocity separately we consider the radially symmetric patch of vorticity as defined by [30].

$$\omega(r) = \begin{cases} (1 - r^2)^7, & r \leq 1, \\ 0, & r > 1, \end{cases} \quad (26)$$

where  $r = \sqrt{x^2 + y^2}$  is the radial distance from the centre of the domain, and

$$\mathbf{u}(x, y) = \begin{cases} -\frac{1}{16r^2} (1 - (1 - r^2)^8) \begin{pmatrix} y \\ -x \end{pmatrix}, & r \leq 1, \\ -\frac{1}{16r^2} \begin{pmatrix} y \\ -x \end{pmatrix}, & r > 1 \end{cases} \quad (27)$$

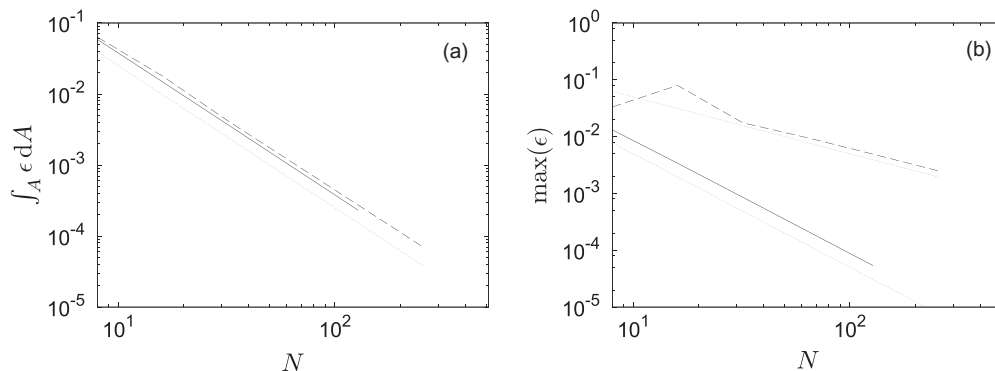
is the corresponding velocity field. The extend of the computational domain is  $3 \times 3$  and the  $1 \times 1$  centre of the domain is divided into four equally sized quadratic patches with side length  $l_2 = 1/2$ . Thus the patch vorticity field is discontinuous at the patch boundary which may lead to Gibb's phenomenon when performing the Fourier transform. The Gibb's oscillations can be suppressed by windowing [31] and spectral convergence can be achieved [32] by careful selection of windowing functions and by extending the size of the buffer. However, as this requires a 360% increase of the domain size in each physical direction [32] and since  $\mathcal{O}(2)$  finite differences are applied to the result, we have found that a convergent solution can be obtained without windowing. Thus, by extending the vorticity field into the buffer the discontinuity and associated errors are moved away from the region of interest; the patch interior. Velocity values in the buffer are then filled from the parent or neighbouring patches.

Fig. 4a shows the integral absolute error of the numerical result to Eq. (27) over the patch interior as function of the number mesh cells  $N$  along a side of the quadratic patches. For buffer widths of  $1.8l_2$  the integral error converges according to the  $\mathcal{O}(\Delta x^2)$  finite difference scheme used. For a buffer width of  $0.18l_2$ , the integral error is increased by 20–40% but converges  $\mathcal{O}(\Delta x^2)$ . Fig. 4b shows the maximum error. For the buffer width of  $1.8l_2$  the maximum error is aligned with the  $N^{-2}$  trend line, the expected convergence of the  $\mathcal{O}(\Delta x^2)$  the finite difference approximation. However, by decreasing the buffer width to  $0.18l_2$  the error converges  $\mathcal{O}(\Delta x)$ , indicating that errors due to discontinuities do not dominate the  $1.8l_2$  errors.

On the coarsest mesh it may be necessary to mollify the vorticity field at the domain boundary by modifying and reducing the vorticity field. As proposed in [33] the boundary vorticity is ramped sinusoidally to zero in the present work. Compared to the vorticity being discarded as it is advected across the domain boundary this is an insignificant modification of the vorticity field.

#### 4.2. Impulsively started flow past a circular cylinder at $Re = 550$

We simulate the impulsively started flow around a circular cylinder which has been thoroughly studied in the literature c.f. e.g. [24,33]. We simulate the flow at Reynolds number  $Re = U2R/\nu = 550$ , where  $U$  is the far field velocity and  $R$  is the radius of the cylinder. Initially we resolve the domain by one uniform patch, i.e. by the standard VIC algorithm. The mesh spacing is  $\Delta x_1 = R/256$  and the cylinder is centred at  $[2R, 2.5R]$  from the bottom left corner of the  $8R \times 5R$  domain. The results are presented in non-dimensional time  $tU/R$  and the dimensionless time step size is  $\Delta tU/R = 2.5 \cdot 10^{-4}$ . Simulation is carried out until  $tU/R = 6$ . The circular cylinder is imposed by the interpolation-penalization method and is defined by  $\chi$  and  $S$  as



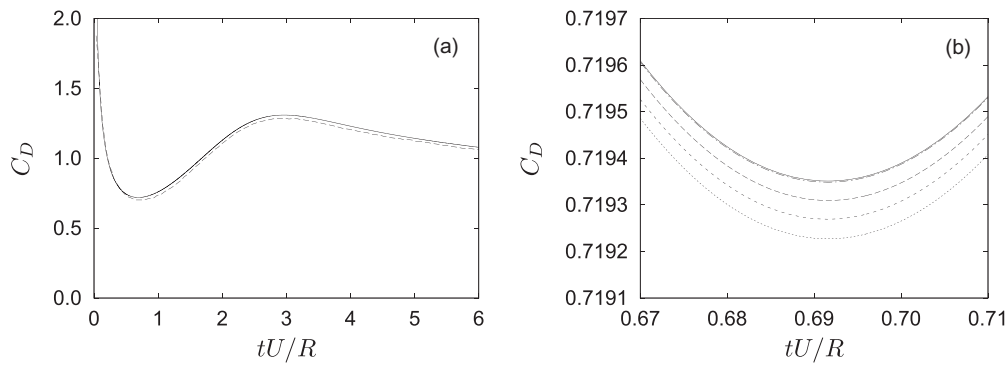
**Fig. 4.** Spatial convergence study for the Perlman vorticity patch Eq. (26) while extending the vorticity field to a  $1.8l_2$  buffer (—) or a  $0.18l_2$  buffer (---): (a) Integral of the magnitude of the deviation of the computed velocity to Eq. (27) as a function of the number of mesh cells  $N$  along the side of the quadratic patch.  $N^{-1}$  is shown as reference (.....). (b) The maximum patch interior error as a function of  $N$ .  $N^{-1}$  and  $N^{-2}$  are shown as reference (.....).

described in Section 2.1 with  $L = 3\sqrt{2}\Delta x_1$ . The force acting on the cylinder is calculated by integrating the time averaged change of the penalization Eq. (16) and from the vorticity moments Eq. (17).

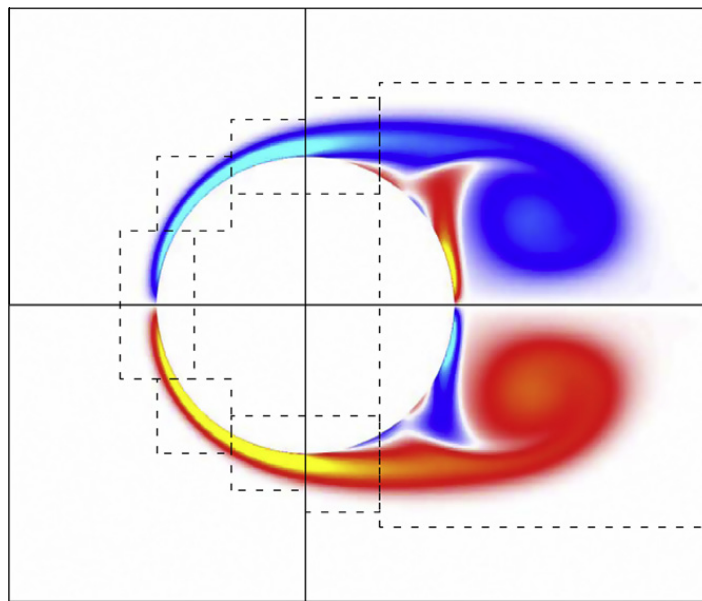
Fig. 5a shows the time evolution of drag coefficient  $C_D = F_D / (\frac{1}{2}\rho U^2 D)$ , where  $F_D$  is the drag force; the streamwise component of the aerodynamic force vector. The agreement with the results of Koumoutsakos and Leonard [24] is excellent until  $tU/R = 0.5$  where the present results deviate by assuming a slightly higher value of  $C_D$ . This offset remains of the same order of magnitude and within 3% of the reference. It is conjectured that the offset may be due to the proximate definition of the solid mask  $\chi$ .

To investigate the effect of the present multiresolution algorithm we reduce the resolution of the base patch to  $\Delta x_1 = R/128$  and fill the base patch entirely by patches of resolution  $\Delta x_2 = R/256$ . The patch buffers are  $18\Delta x_2$  wide. This *quartered* layout of patches can be seen on Fig. 6. We only penalize the flow at level 2, thus the solution is comparable to that of the uniformly resolved domain. Fig. 5b shows a close up of the local minimum of  $C_D$  at  $tU/R \approx 0.7$ . The simulation is carried out until  $tU/R = 1.25$ . The coefficient of drag is 0.02% higher for the quartered patch layout than for the uniform reference solution. For the uniform domain and the quartered patch layout  $C_D$  calculated with vorticity moments is 0.011% and 0.016% lower than when integrating the penalization term.

By adding patches at the boundary of the cylinder and in the immediate wake, see Fig. 6, the computational resources are focused more efficiently at the high vorticity regions. As for the quartered layout the patch buffers are  $18\Delta x_2$  wide. The varying spatial resolution, in particular in the regions of lower resolution, could potentially alter the simulation. However, the most significant impact of this setup is the absence of patches in the solid interior. Because the velocity by which the aerodynamic forces are calculated (Eq. (16)) is found from the vorticity field diffused by Eq. (1) some finite deviation to the solid velocity field must be expected in the interior of the solid. Due to the numerical evaluation of the curl (Eq. (11)) at different



**Fig. 5.** (a) Coefficient of drag for onset flow around circular cylinder. The present results (—) agree with the results of Koumoutsakos and Leonard [24] (---) within 3% for  $Re = 550$ . (b) Close-up of  $C_D$  for uniform resolution by penalization term (—) and vorticity moments (---).  $C_D$  for the quartered patch layout by penalization term (— · —) and vorticity moments (·····).  $C_D$  for a boundary and wake patched domain by vorticity moments (— · · —) coincides with the solid line.



**Fig. 6.** Snapshot of vorticity for the impulsively started flow around a cylinder at  $Re = 550$  and  $tU/R = 4$ . The domain is overlaid by the quartered layout of patches (—) and the 8 boundary and wake patches (-----).

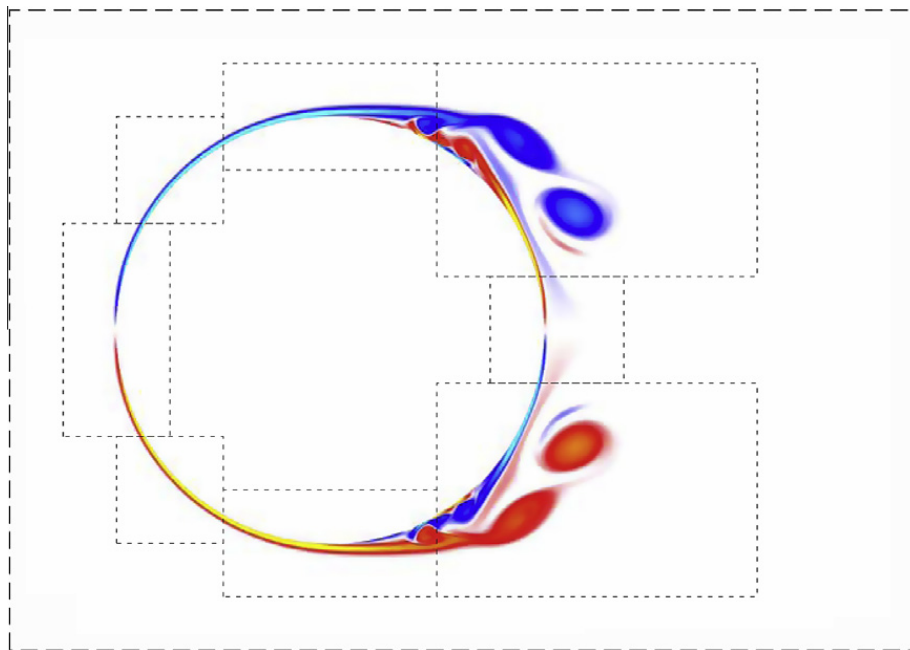
resolutions, the resulting vorticity and in turn the corresponding velocity will vary when evaluating Eq. (16). When the force is partially evaluated on the coarser mesh in the solid interior this results in a 13% deviation for the current setup. The vorticity moments depend only on the evolution of the vorticity and give correct results as shown in Fig. 5(b). Therefore for more complex arrangements of patches the vorticity moments should be used to compute the body forces.

#### 4.2.1. Impulsively started flow past a circular cylinder at $Re = 9500$

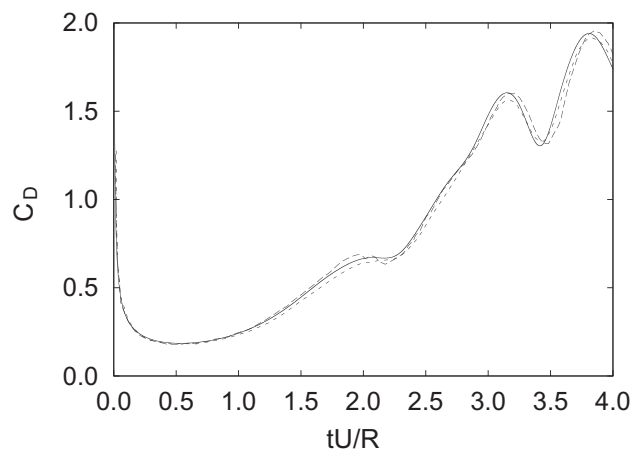
Increasing the Reynolds number to 9500 leads to a more complex flow requiring a finer resolution and time integration, i.e.  $\Delta tU/R = 10^{-4}$ . Fig. 7 shows the vorticity field at  $tU/R = 3.0$  overlaid by patches of varying resolution. On the base mesh one patch of twice the base mesh resolution ( $\Delta x_2 = R/256$ ) is positioned around the cylinder and its wake. This patch has a buffer of width  $8\Delta x_2$ . On this patch several patches of four times the base mesh resolution ( $\Delta x_3 = R/512$ ) is positioned on the cylinder boundary and the immediate wake. All level 3 patches have buffers  $18\Delta x_3$  wide. The drag force shown in Fig. 8 is computed using vorticity moments and matches previous results cf. [24,33].

#### 4.2.2. Impulsively started flow around a rotating circular cylinder at $Re = 1000$

We centre a cylinder in the  $y$ -direction in a  $16R \times 10R$  domain at a  $\Delta x_1 = R/64$  resolution. Two nested patches cover the cylinder and its immediate wake with a resolution  $\Delta x_3 = R/256$  at the highest level (not shown). The buffers are of width  $6\Delta$



**Fig. 7.** The vorticity around a cylinder in an onset flow for  $Re = 9500$  at  $tU/R = 3$ . The lines (-----) and (-----) show the patches of resolution  $\Delta x_2 = R/256$  and  $\Delta x_3 = R/512$ . The base patch (level 1) is not shown.

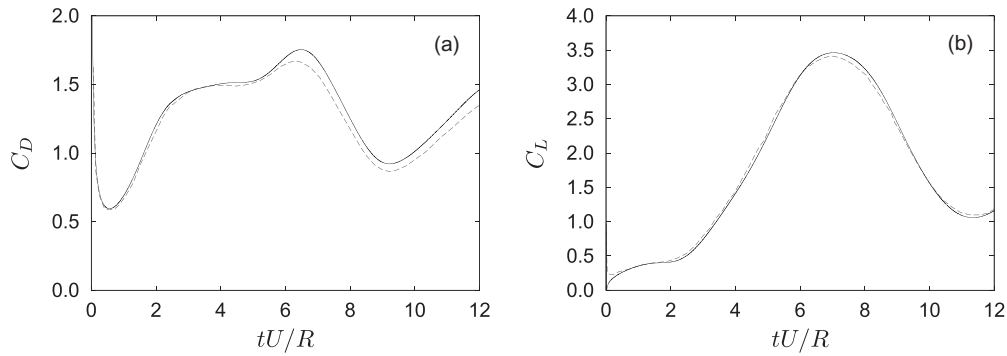


**Fig. 8.** The calculated  $C_D$  (—) for the onset flow around a circular cylinder at  $Re = 9500$  shows good agreement with the results of Koumoutsakos and Leonard [24] (-----) and Rossinelli et al. [33] (-----).

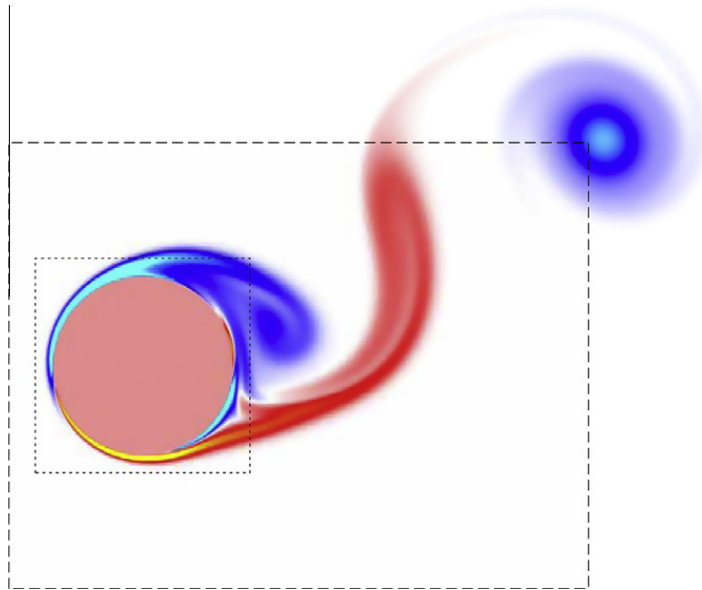
$x_2$  and  $10\Delta x_3$  respectively. We initialise the solid body rotation velocity  $\mathbf{u}_s$  with the ratio of the counter-clockwise circumferential velocity to the far field velocity  $R\Omega_s/U = 1$ , where  $\Omega_s$  is the angular velocity of the cylinder. Fig. 9 shows the computed coefficient of drag and coefficient of lift  $C_L = F_L / (\frac{1}{2}\rho U^2 D)$ , where  $F_L$  is the component of the aerodynamic force normal to the direction of the flow, i.e. the lifting force. The drag deviates by 6.5% to the results of Chou [34] and apart from the initial development  $C_L$  deviates by 2.5% from this reference solution. The simulations has run at two different resolutions with insignificant variation of the results. (See Fig. 10)

#### 4.2.3. Flow around Great Belt East suspension bridge deck

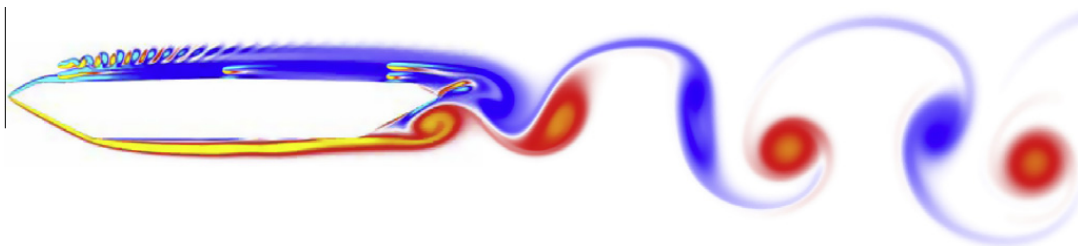
To demonstrate the versatility of the method the flow around the Great Belt East bridge is simulated at  $Re = 10000$  based on the chord  $C$  and the free-stream velocity. The geometry includes railing and the crash barrier cf. Fig. 11. The measured



**Fig. 9.** (a)  $C_D$  and (b)  $C_L$  for the impulsively started flow around a cylinder rotating with a non-dimensional speed of  $R\Omega_s/U = 1$ . Present values (—) and the reference by Chou [34] (-----).



**Fig. 10.** The vorticity around a rotating cylinder at  $tU/R = 11$  with  $R\Omega_s/U = 1$  and  $Re=1000$ . The lines (-----) and (.....) show the patches of resolution  $\Delta x_2 = R/128$  and  $\Delta x_3 = R/256$ . The base patch (level 1) is not shown.



**Fig. 11.** The vorticity field around the Great Belt East suspension bridge deck at  $tU/C = 6.9$ .

time average of the lift and drag,  $C_L = 0.0445$  and  $C_D = -0.0533$ , deviates by 11.6% and 3.08% respectively compared to the results of the flow solver DVMFLOW [29,35]. The vorticity field (Fig. 11) shows large scale vortex shedding in the wake, but also higher frequency vortex shedding from the upstream railings.

## 5. Conclusions

We introduce a novel multiresolution Vortex-In-Cell (VIC) algorithm using patches of varying resolution. The algorithm is based on free-space FFTs for the solution of the Poisson equation on the patches. Superposition is used to compute the contribution from vorticity of separate meshes. The present multiresolution algorithm is implemented in two dimensions but is, like the general VIC algorithm [12], not restricted to two dimensions. We are currently working on extending the implementation to three dimensions through the open source parallel particle mesh (PPM) library [11]. We demonstrate that penalization can be carried out as an interpolation between the solid and fluid velocities in a manner similar to the semi-implicit penalization method. This interpolating penalization eliminates the penalization parameter. The algorithm is validated by considering the impulsively started flow past fixed and rotating circular cylinders. The flow past a suspension bridge including railings and crash barriers demonstrate the ability of proposed multiresolution VIC algorithms to handle complex geometries.

## Acknowledgement

We would like to acknowledge the helpful discussions with Diego Rossinelli, Wim van Rees and Petros Koumoutsakos. The research has been supported by the Danish Research Council of Independent Research (Grant. No. 274-08-0258).

## References

- [1] C.K. Birdsall, D. Fuss, Clouds-in-clouds, clouds-in-cells physics for many-body plasma simulation, *J. Comput. Phys.* 3 (1969) 494–511.
- [2] J.P. Christiansen, Numerical simulation of hydrodynamics by the method of point vortices, *J. Comput. Phys.* 13 (1973) 363–379.
- [3] L. Greengard, V. Rokhlin, A fast algorithm for particle simulations, *J. Comput. Phys.* 73 (1987) 325–348.
- [4] J. Carrier, L. Greengard, V. Rokhlin, A fast adaptive multipole algorithm for particle simulations, *SIAM J. Sci. Stat. Comput.* 9 (4) (1988) 669–686.
- [5] H. Cheng, L. Greengard, V. Rokhlin, A fast adaptive multipole algorithm in three dimensions, *J. Comput. Phys.* 155 (1999) 468–498.
- [6] P. Ploumhans, G.S. Winckelmans, J.K. Salmon, A. Leonard, M.S. Warren, Vortex methods for direct numerical simulation of three-dimensional bluff body flows: Applications to the sphere at  $Re = 300$ , 500 and 1000, *J. Comput. Phys.* 178 (2002) 427–463.
- [7] G.-H. Cottet, Particle-grid domain decomposition methods for the Navier–Stokes equations in exterior domains, *Lect. Appl. Math.* 28 (1991) 103–117.
- [8] M.L. Ould-Salihi, G.-H. Cottet, M. El Hamraoui, Blending finite-difference and vortex methods for incompressible flow computations, *SIAM J. Sci. Comput.* 22 (5) (2000) 1655–1674.
- [9] G. Winckelmans, R. Cogle, L. Dufresne, R. Capart, Vortex methods and their application to trailing wake vortex simulations, *C.R. Physique* 6 (2005) 467–486.
- [10] R. Cogle, G. Winckelmans, G. Daeninck, Combining the vortex-in-cell and parallel fast multipole methods for efficient domain decomposition simulations, *J. Comput. Phys.* 227 (2008) 9091–9120.
- [11] I.F. Sbalzarini, J.H. Walther, M. Bergdorf, S.E. Hieber, E.M. Kotsalis, P. Koumoutsakos, PPM – a highly efficient parallel particle-mesh library for the simulation of continuum systems, *J. Comput. Phys.* 215 (2006) 566–588.
- [12] P. Chatelain, A. Curioni, M. Bergdorf, D. Rossinelli, W. Andreoni, P. Koumoutsakos, Billion vortex particle direct numerical simulations of aircraft wakes, *Comp. Methods Appl. Mech. Eng.* 197 (2008) 1296–1304.
- [13] M. Coquerelle, G.-H. Cottet, A vortex level set method for the two-way coupling of an incompressible fluid with colliding rigid bodies, *J. Comput. Phys.* 227 (21) (2008) 9121–9137.
- [14] R.W. Hockney, The potential calculation and some applications, *Methods Comput. Phys.* 9 (1970) 136–210.
- [15] R.W. Hockney, J.W. Eastwood, in: *Computer Simulation Using Particles*, second ed., Institute of Physics Publishing, Bristol, PA, USA, 1988.
- [16] G. Morgenthal, J.H. Walther, An immersed interface method for the vortex-in-cell algorithm, *Comput. Struct.* 85 (2007) 712–726.
- [17] P. Chatelain, P. Koumoutsakos, A Fourier-based elliptic solver for vortical flows with periodic and unbounded directions, *J. Comput. Phys.* 229 (2010) 2425–2431.
- [18] G.-H. Cottet, P. Koumoutsakos, M.L.O. Salihi, Vortex methods with spatially varying cores, *J. Comput. Phys.* 162 (1) (2000) 164–185.
- [19] P.K. Stansby, A. Slaouti, Simulation of vortex shedding including blockage by the random-vortex and other methods, *Int. J. Numer. Methods Fluids* 7 (1993) 1003–1013.
- [20] M. El Ossmani, P. Poncet, Efficiency of multiscale hybrid grid-particle vortex methods, *Multiscale Model. Simul.* 8 (5) (2010) 1671–1690.
- [21] M. Bergdorf, G.-H. Cottet, P. Koumoutsakos, Multilevel adaptive particle methods for convection-diffusion equations, *Multiscale Model. Simul.* 4 (1) (2005) 328–357.
- [22] G. Barton, *Elements of Green's functions and propagation*, in: *Potentials, Diffusion and Waves*, Oxford University Press, 1989.
- [23] J.J. Monaghan, Extrapolating B splines for interpolation, *J. Comput. Phys.* 60 (2) (1985) 253–262.
- [24] P. Koumoutsakos, A. Leonard, High-resolution simulation of the flow around an impulsively started cylinder using vortex methods, *J. Fluid Mech.* 296 (1995) 1–38.
- [25] P. Angot, C.-H. Bruneau, P. Fabrie, A penalization method to take into account obstacles in incompressible viscous flows, *Numer. Math.* 81 (1999) 497–520.
- [26] N.K.-R. Kevlahan, J.-M. Ghidaglia, Computation of turbulent flow past an array of cylinders using a spectral method with Brinkman penalization, *Eur. J. Mech. B* 20 (2001) 333–350.
- [27] J.C. Wu, N.L. Sankar, Aerodynamic force and moment in steady and time-dependent viscous flows, in: *18th AIAA Aerospace Sciences Meeting*, Pasadena, California, 1980.
- [28] F. Noca, D. Shiels, D. Jeon, Measuring instantaneous fluid dynamic forces on bodies, using only velocity fields and their derivatives, *J. Fluids Struct.* 11 (1997) 345–350. special brief note.
- [29] J.H. Walther, A. Larsen, Discrete vortex method for application to bluff body aerodynamics, *J. Wind Eng. Ind. Aerodyn.* 67–68 (1997) 183–193.
- [30] M. Perlman, On the accuracy of vortex methods, *J. Comput. Phys.* 59 (1985) 200–223.
- [31] F.J. Harris, On the use of windows for harmonic analysis with the discrete Fourier transform, *Proc. IEEE* 66 (1) (1978) 51–83.
- [32] P. Schlatter, N.A. Adams, L. Kleiser, A windowing method for periodic inflow/outflow boundary treatment of non-periodic flows, *J. Comput. Phys.* 206 (2005) 505–535.



- [33] D. Rossinelli, M. Bergdorf, G.-H. Cottet, P. Koumoutsakos, GPU accelerated simulations of bluff body flows using vortex particle methods, *J. Comput. Phys.* 229 (89) (2010) 3316–3333.
- [34] M.-H. Chou, Numerical study of vortex shedding from a rotating cylinder immersed in a uniform flow field, *Int. J. Numer. Methods Fluids* 32 (5) (2000) 545–567.
- [35] A. Larsen, J.H. Walther, Discrete vortex simulation of flow around five generic bridge deck sections, *J. Wind Eng. Ind. Aerodyn.* 77–78 (1998) 591–602.