Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 227 (2008) 9063–9090
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A fast 3D particle method for the simulation of buoyant flow

Fabrice Schlegel, Daehyun Wee, Ahmed F. Ghoniem *

Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, MA 02139, United States

Received 28 February 2007; received in revised form 22 January 2008; accepted 23 March 2008
Available online 10 April 2008

Abstract

This paper describes progress in several areas related to three-dimensional vortex methods and their application to multiphysics problems. The first is the solution of a generic scalar transport equation by advecting and diffusing the scalar gradient along a particle trajectory and onto a mesh, respectively, and recovering the scalar values using a Biot–Savart-like summation. The second is the accurate, high-resolution calculation of the velocity gradient using a fast treecode, which avoids using kinematic relations between the evolution of the gradients and the distortion of the flow map. The same tree structure is used to compute all the variables of interest and those required during the integration of the governing equations. Next, we apply our modified interpolation kernel algorithm for treating diffusion and remeshing to maintain long time accuracy. The coupling between vorticity transport and that of a dynamic scalar, in this case the temperature or density in a gravitational field, is manifested by the generation of vorticity. We demonstrate the performance of the multiphysics algorithm by solving a number of buoyant flow problems.

Keywords: Particle methods; Buoyant flow; Fast methods; Transport elements

1. Introduction

Lagrangian vortex methods [1,2] are tools for computing complex unsteady fluid flows at high Reynolds numbers. While they have other advantages, such as the relaxation of the CFL condition and the suppression of numerical diffusion, one of their most interesting features is the fact that they are based on the discretization of vorticity. Especially in unconfined and semi-confined flows, a typical computational domain must extend to a size that incorporates regions where the primary variables, i.e., velocity and pressure, may deviate very slightly from their uniform distribution. This can result in an unmanageable computational effort in 3D or would require complex non-uniform grids that cluster around zones of high gradients and transition to coarser meshes closer to uniform zones. Vorticity, on the other hand, is derived from the curl of the velocity field, and can be described by computational elements contained in a smaller fraction of the total volume of the flow field. As the result, the computational elements are utilized more efficiently. Lagrangian transport of vorticity guarantees that its evolution in space and time is well resolved.

* Corresponding author. Tel.: +1 617 253 2295; fax: +1 617 253 5981.
E-mail address: ghoniem@mit.edu (A.F. Ghoniem).

0021-9991/$ - see front matter © 2008 Published by Elsevier Inc.
doi:10.1016/j.jcp.2008.03.036

The extension of this idea to general transport problems has been suggested and implemented in several contexts. In this methodology, when solving for the transport of a scalar variable, one discretizes the gradients of the scalar field, instead of the scalar field itself. The evolution of the scalar field is hence determined by solving the corresponding transport equation for its gradients. The advantage of this approach is identical to that described in the previous paragraph for vortex methods. Since the gradients can be described by computational elements confined to a small fraction of the total volume of the domain, one can utilize the discrete elements more efficiently.

These ideas were first described for 1D problem in Ghoniem and Oppeneim for modelling diffusion processes [3]. Further developments were suggested and implemented more generically in Ghoniem and Sherman [4]. Anderson [5] extended the concept of gradient transport to convection in 2D, and to buoyant flows. A conservative formulation of that construction was suggested by Ghoniem et al. [6], called the transport element method, and used it for the simulation of mixing in shear flows. Krishnan and Ghoniem [7] extended the transport element method to nearly inviscid buoyant flows in 2D. They studied a two-dimensional Rayleigh–Taylor flow evolving under the action of gravity across a large temperature gradient, i.e., without the Boussinesq approximation. A reacting flow version of transport element methods was also proposed by Soteriou and Ghoniem [8,9] to investigate the dynamics of two-dimensional reacting shear layers. Soteriou et al. [10] applied transport element methods to planar buoyant plumes simulations.

Three-dimensional transport element method was proposed by Knio and Ghoniem, and was used to simulate the evolution of a periodic shear layer [11,12]. The construction of this method was, however, based on a rather complicated internal coordinate system inside each computational element, which made the implementation difficult. Essentially, the construction was based on a kinematic relation between the evolution of the local gradients and that of the distortion of the material elements. To implement this kinematic relation, one needs to evolve the underlying flow map, that is, both the location of the field particles and their connectivity.

In this paper, we resurrect the basic concepts of the transport element methods in the context of three-dimensional multi-physics problems, where the vorticity field and the scalar field are coupled by baroclinicity, but simplify its implementation using a number of new ideas. The scheme is equipped with a multi-purpose adaptive treecode, which enables fast and accurate evaluation of various quantities required for the simulation, i.e., velocity, velocity gradients, and scalar distribution. Accurate and fast evaluation of velocity gradients enables us to solve the scalar gradient evolution without complex internal coordinate systems, with negligible loss of conservation properties. The capability of the scheme is demonstrated in various three-dimensional flows driven by buoyancy.

This paper is organized as follows. The governing equations are given in Section 2 and the numerical algorithm is presented in Section 3. Section 4 is dedicated to the simulation of a vortex ring. Buoyancy-driven flows are studied in Section 5, and conclusions are given in Section 6. Finally, our multi-purpose adaptive treecode is presented in detail in Appendix.

2. Governing equations

To demonstrate the capability of our transport element method, we study buoyancy-driven flows in $\mathbf{R}^3$. Using the Boussinesq approximation, the Navier–Stokes equation is given as follows:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} = v\Delta\mathbf{u} - \frac{1}{\rho}\nabla p - \mathbf{g_r}\beta(T - T_\infty), \tag{1}$$

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

where $\mathbf{u}$ is velocity, p is pressure, β is the volumetric thermal expansion coefficient of the fluid, and $\mathbf{g_r}$ is the gravitational acceleration. T_∞ is the temperature of the environment, ∇ and Δ are the gradient and Laplacian operators. The temperature field,, is governed by the following advection–diffusion equation:

$$\frac{\partial T}{\partial t} + \mathbf{u}\cdot\nabla T = \alpha\Delta T. \tag{3}$$

Here, v is the kinematic viscosity, and α is the thermal diffusivity.

We normalize Eqs. (1)–(3) by choosing a reference length L, which can be defined by a characteristic geometric length scale of the problem. The corresponding reference flow speed is given by $U = \sqrt{g_r L}$, where $g_r^2 = \mathbf{g_r} \cdot \mathbf{g_r}$. We also choose a reference temperature, T_0, which is different from T_∞. Then, with the following normalization: $\tilde{\mathbf{x}} = \mathbf{x}/L,\ \tilde{\mathbf{u}} = \mathbf{u}/U,\ \tilde{t} = t/(L/U),\ \tilde{\theta} = (T - T_\infty) - (T_0 - T_\infty),\ \tilde{p} = p/(\rho U^2)$, and $\tilde{\mathbf{g}}_\mathbf{r} = \mathbf{g_r}/g_r$, we obtain

$$\frac{\partial \tilde{\mathbf{u}}}{\partial \tilde{t}} + \tilde{\mathbf{u}} \cdot \tilde{\nabla}\tilde{\mathbf{u}} = \frac{1}{Re}\tilde{\Delta}\tilde{\mathbf{u}} - \tilde{\nabla}\tilde{p} - \frac{Gr}{Re^2}\tilde{\mathbf{g}}_\mathbf{r}\tilde{\theta}, \tag{4}$$

$$\tilde{\nabla} \cdot \tilde{\mathbf{u}} = 0, \tag{5}$$

$$\frac{\partial \tilde{\theta}}{\partial \tilde{t}} + \tilde{\mathbf{u}} \cdot \tilde{\nabla}\tilde{\theta} = \frac{1}{PrRe}\tilde{\Delta}\tilde{\theta}. \tag{6}$$

Naturally, $\tilde{\nabla} = \frac{1}{L}\nabla$ and $\tilde{\Delta} = \frac{1}{L^2}\Delta$. The system is governed by three dimensionless parameters, i.e., the Reynolds number, $Re = UL/v$, the Prandtl number, $Pr = v/\alpha$, and the Grashof number, $Gr = g_r\,\beta(T_0 - T_\infty)L^3/v^2$. In the following, all the variables are understood as being normalized in this form, and the tilde over each dimensionless variable is omitted.

Taking the curl of Eq. (4), we obtain the vorticity–velocity formulation for buoyant flows:

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \mathbf{u} \cdot \nabla\boldsymbol{\omega} = \boldsymbol{\omega} \cdot \nabla\mathbf{u} + \frac{1}{Re}\Delta\boldsymbol{\omega} + \frac{Gr}{Re^2}\mathbf{g}_\mathbf{r} \times \nabla\theta, \tag{7}$$

where $\boldsymbol{\omega} = \nabla \times \mathbf{u}$, and $\frac{Gr}{Re^2}\mathbf{g}_\mathbf{r} \times \nabla\theta$ is the baroclinic source term for vorticity generation.

Using the Helmholtz decomposition, the velocity field can be separated as follows:

$$\mathbf{u} = \mathbf{u}_\omega + \mathbf{u}_\mathrm{p}, \tag{8}$$

where $\mathbf{u}_\omega$ is the vortical velocity field, and $\mathbf{u}_\mathrm{p}$ is the potential velocity field. The potential velocity is added to satisfy the normal velocity boundary conditions. In $\mathbf{R}^3$, where no apparent boundary exists, $\mathbf{u}_\mathrm{p}$ is set to 0, and $\mathbf{u} = \mathbf{u}_\omega$. On the other hand, given a distribution of vorticity within a domain D, the vortical velocity in $\mathbf{R}^3$ is determined using the Biot–Savart law:

$$\mathbf{u}(\mathbf{x}, t) = \mathbf{u}_\omega(\mathbf{x}, t) = -\frac{1}{4\pi}\int_D \frac{(\mathbf{x} - \mathbf{x}') \times \boldsymbol{\omega}(\mathbf{x}', t)}{|\mathbf{x} - \mathbf{x}'|^3}\,\mathrm{d}\mathbf{x}'. \tag{9}$$

The set of equations, Eqs. (6), (7) and (9), provides a complete description of the flow.

3. Numerical algorithm

Our approach uses Lagrangian particles as computational elements. The vorticity field is discretized into Lagrangian computational elements, or particles, with weights $\mathbf{W}_i(t)$, and locations $\boldsymbol{\chi}_i(t)$ such that

$$\boldsymbol{\omega}(\mathbf{x}, t) \approx \sum_i^{N_\omega} \mathbf{W}_i(t) f_\delta(\mathbf{x} - \boldsymbol{\chi}_i(t)), \tag{10}$$

where N_ω is the number of vortex elements. The core function $f_\delta(\mathbf{r})$ is obtained from a reference function $f(r)$ by $f_\delta(\mathbf{r}) = \delta^{-3} f(|\mathbf{r}|/\delta)$. In this work, the reference function f is the low-order algebraic core function [16]:

$$f(r) = \frac{3}{4\pi}\frac{1}{(1 + r^2)^{5/2}}. \tag{11}$$

To simulate buoyant flows, we need to additionally solve the transport equation of energy or temperature, Eq. (6). Since the baroclinic source term in Eq. (7) depends on the gradient of θ, instead of using the scalar values as weights and computing their gradients locally, we use the scalar gradients as the weights of the corresponding computational elements. The advantage of using the gradients as weights is that the support of the gradient is smaller than that of the scalar itself, and hence the computational elements can be distributed over a smaller fraction of the domain. For instance, a hot sphere can be represented by discretizing the spherical shell between the hot interior and the cold exterior, while no elements are used in the temperature domains inside or out.

Such a method is generally referred as a transport element scheme. In the context of the current problem, we discretize the gradient of θ as follows:

$$\mathbf{g}(\mathbf{x}) = \nabla\theta(\mathbf{x}) \approx \sum_{i}^{N_g} \mathbf{G}_i(t) f_\delta(\mathbf{x} - \boldsymbol{\zeta}_i(t)), \tag{12}$$

where N_g is the number of transport elements. By taking gradient of Eq. (6), we get the governing equation for $\mathbf{g}$,

$$\frac{\partial \mathbf{g}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{g} = -(\nabla\mathbf{u})^t\cdot\mathbf{g} + \frac{1}{PrRe}\Delta\mathbf{g} \tag{13}$$

The solution of the equations of motion is expressed in terms of the instantaneous locations, i.e., $\boldsymbol{\chi}_i$ and $\boldsymbol{\zeta}_i$, and weights, i.e., $\mathbf{W}_i$ and $\mathbf{G}_i$, of these elements.

The numerical solution of Eqs. (7) and (13) is obtained through an operator splitting. The computational time step is split into three substeps, i.e., convection substep, generation substep, and diffusion substep. During each substep, we solve the following equations:

$$\text{Convection substep:} \quad \frac{D\boldsymbol{\omega}}{Dt} = \frac{\partial\boldsymbol{\omega}}{\partial t} + \mathbf{u}\cdot\nabla\boldsymbol{\omega} = \boldsymbol{\omega}\cdot\nabla\mathbf{u}, \tag{14}$$

$$\frac{\mathrm{D}\mathbf{g}}{\mathrm{D}t} = \frac{\partial\mathbf{g}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{g} = -(\nabla\mathbf{u})^t\cdot\mathbf{g}, \tag{15}$$

$$\text{Generation substep:} \quad \frac{\partial\boldsymbol{\omega}}{\partial t} = \frac{Gr}{Re^2}\mathbf{g}_\mathbf{r}\times\nabla\theta, \tag{16}$$

$$\text{Diffusion substep:} \quad \frac{\partial\boldsymbol{\omega}}{\partial t} = \frac{1}{Re}\Delta\boldsymbol{\omega}, \tag{17}$$

$$\frac{\partial\mathbf{g}}{\partial t} = \frac{1}{PrRe}\Delta\mathbf{g}. \tag{18}$$

During the convection substep, the solution of Eqs. (14) and (15) is obtained by integrating the following equations:

$$\frac{\mathrm{d}\boldsymbol{\chi}_i}{\mathrm{d}t} = \mathbf{u}(\boldsymbol{\chi}_i, t), \tag{19}$$

$$\frac{\mathrm{d}\boldsymbol{\zeta}_i}{\mathrm{d}t} = \mathbf{u}(\boldsymbol{\zeta}_i, t), \tag{20}$$

$$\frac{\mathrm{d}\mathbf{W}_i}{\mathrm{d}t} = \mathbf{W}_i(t)\cdot\nabla\mathbf{u}(\boldsymbol{\chi}_i, t), \tag{21}$$

$$\frac{\mathrm{d}\mathbf{G}_i}{\mathrm{d}t} = -(\nabla\mathbf{u}(\boldsymbol{\zeta}_i, t))^t\cdot\mathbf{G}_i(t). \tag{22}$$

The integration of these equations is performed using a second-order predictor–corrector scheme.[1]

During the generation substep, Eq. (16) for the baroclinic generation of vorticity is integrated using a first-order scheme. We need to introduce additional vorticity, where non-trivial baroclinicity exists. This is achieved by generating one new vortex element at the location of each transport element, using the following expression for each ith transport element:

$$\mathbf{W}_i^g = \left(\frac{Gr}{Re^2}\mathbf{g}_\mathbf{r}\times\mathbf{G}_i(t)\right)\Delta t \quad \text{for } 1 \leqslant i \leqslant N_g. \tag{23}$$

[1] Since Eqs. (19) and (20) have an identical form, it is possible to use one single set of computational elements representing both $\boldsymbol{\omega}$ and $\mathbf{g}$, instead of two different sets of computational elements. In the case where the support of $\boldsymbol{\omega}$ and that of $\mathbf{g}$ do not overlap much, using two different sets is memory-efficient. On the other hand, if there exists much overlap, it is better to use one single set of computational elements. Though we have employed two different sets of elements to test the capability of our transport element method here, it should be noted that the latter is usually the case in practical situations, where baroclinic interaction is dominant.

The vorticity field is updated by adding these new vortex elements to the existing vortex elements,

$$\boldsymbol{\omega}(\mathbf{x}) = \sum_{i=1}^{N_\omega} \mathbf{W}_i f_\delta(\mathbf{x} - \boldsymbol{\chi}_i) + \sum_{i=1}^{N_g} \mathbf{W}_i^g f_\delta(\mathbf{x} - \boldsymbol{\zeta}_i). \tag{24}$$

After the update is finished, N_ω is augmented by N_g.

During the diffusion substep, Eqs. (17) and (18) are solved by using a modified interpolation kernel [13]. The existing field is interpolated over a new set of elements, whose location is selected to satisfy certain requirements. Through the interpolation process, the vorticity field is updated such that the strength of the new elements are

$$\mathbf{W}_j(t + \Delta t) = \sum_i f_{ij}^\omega \mathbf{W}_i(t), \tag{25}$$

where f_{ij}^ω is the redistribution fraction from the ith vortex element to the jth grid point, which in the current implementation is described by a uniform Cartesian grid with the grid size Δx as shown in Fig. 1. f_{ij}^ω is obtained by using the interpolation kernel Λ_3:

$$f_{ij}^\omega = \Lambda_3\left(\frac{x_j - x_i}{\Delta x}; c_\omega\right)\Lambda_3\left(\frac{y_j - y_i}{\Delta x}; c_\omega\right)\Lambda_3\left(\frac{z_j - z_i}{\Delta x}; c_\omega\right), \tag{26}$$

where

$$\Lambda_3(\xi; c) = \begin{cases} 1 - 2c^2 + |\xi|(3c^2 - 1/2) - \xi^2 + |\xi|^2/2, & |\xi| < 1, \\ (2 - |\xi|)\left(\frac{1}{6}(3 - |\xi|)(1 - |\xi|) + c^2\right), & 1 \leqslant |\xi| < 2, \\ 0, & 2 \leqslant |\xi|. \end{cases} \tag{27}$$

Here, $c_\omega = \sqrt{Re^{-1}\Delta t}/\Delta x$, which represents the ratio between the diffusion length scale and Δx. As shown in [11], in this modified interpolation kernel, during each interpolation step the second-order moments are increased by the amount required to simulate diffusion. In classical interpolation, the kernel preserves these second-order moments.

In a similar way, we update the gradient field,

$$\mathbf{G}_j(t + \Delta t) = \sum_i f_{ij}^g \mathbf{G}_i(t), \tag{28}$$

where f_{ij}^g is the redistribution fraction from the ith transport element to the jth grid point. f_{ij}^g is obtained by using Eq. (26), but with $c_g = \sqrt{\mathrm{Pr}^{-1}Re^{-1}\Delta t}/\Delta x$ in place of c_ω.

At the end of the interpolation, time is advanced to $t + \Delta t$, and that completes the entire step. By the end of the time step, the fields are again expressed by Eqs. (10) and (12), where i runs over all the grid points with

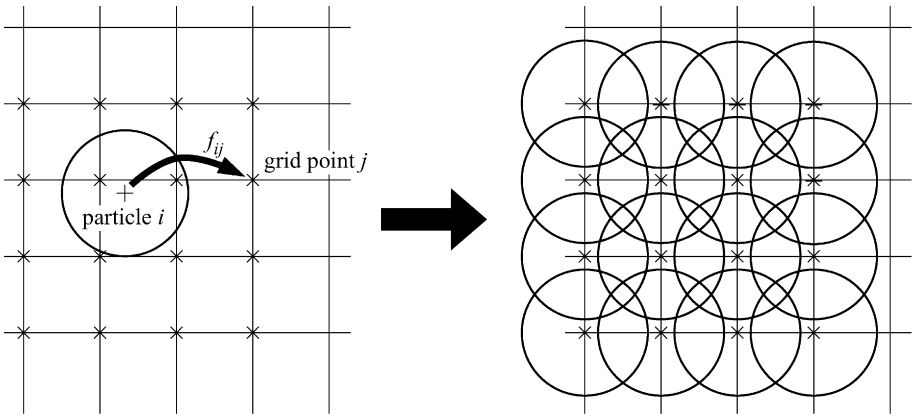

Fig. 1. Schematic illustration of a diffusion substep. Circles represent particles. The strength of each particle is interpolated onto its nearest neighboring grid points, denoted by 'x', which occupy a rectangular domain around the particle (left). After finishing interpolation for all the particles, each grid point containing non-trivial strength is converted into a particle (right).

non-trivial values of $\mathbf{W}_i$ and/or $\mathbf{G}_i$. Note that the problem of Lagrangian distortion is resolved within the diffusion substep, since the new particles are uniformly distributed at the end of the substep.

During the convection substep, we need to evaluate $\mathbf{u}$ and $\Delta\mathbf{u}$ at the location of each computational element. A naïve implementation of this process leads to an expensive operation, whose cost scales as $O(N^2)$. The recovery of θ from $\mathbf{g}$, which is necessary during post-processing, also requires similar set of operations. To reduce the computational cost of these tasks, we use a multi-purpose adaptive treecode, which is described in Appendix A.

Also we note that, for isothermal flows, Eq. (6) is redundant, and the source term for vorticity generation in Eq. (7) drops out. In this hydrodynamic limit, the computational algorithm reduces to the standard vortex element scheme, where only convection and diffusion, i.e., Eqs. (14) and (17), of vorticity is implemented. In the following sections, we first describe some results at this hydrodynamic limit, and then present the results of buoyant flow simulations.

4. Evolution of a vortex ring

In the following, we examine the accuracy and convergence of our algorithm. We first apply the algorithm at the hydrodynamic limit to perform the simulation of a vortex ring. The evolution of a viscous vortex ring was studied with an axisymmetric spectral calculation by Stanaway et al. [14], and the result was later reproduced in a vortex calculation by Wee and Ghoniem [13]. The initial vorticity distribution of the ring core is given by

$$\omega_\phi = \frac{K}{\pi}\frac{\Gamma}{a^2}\exp\left[-K\left(\frac{R^2}{a^2}+\frac{r^2}{a^2}-\frac{2Rr}{a^2}\sin\theta\right)\right] \tag{29}$$

with $r=\sqrt{x^2+y^2+z^2}$, $\tan\theta=\frac{\sqrt{x^2+y^2}}{y}$ and $K=\frac{(2.24182)^2}{4}$. The core radius is chosen to be $a/R=0.35$. The initial ring radius, $R(0)$, and its initial circulation, $\Gamma(0)$, are unity. Its evolution is computed for a Reynolds number $Re\equiv\Gamma/\nu=500$.

The numerical parameters are chosen as follows: the time step for the highest resolution simulation is $\Delta t=0.15$ and the grid size for the diffusion substep is $\Delta x=0.025$. Because of diffusion, the vorticity support expands and the number of particles grows in time. To control the number of particles, particles with its strength below a cutoff value are deleted after each diffusion substep. The cutoff value for deletion is chosen to be $|\omega \mathrm{d}V|_{\mathrm{del}}=10^{-11}$.

The simulation is initialized by computing the vorticity distribution on a Cartesian grid with a grid size $\Delta x=0.025$. The number of vortex elements at the beginning of the simulation is around 1,900,000.

The results are reported in terms of the following dimensionless variables. The dimensionless time is given by

$$\bar{t}=\frac{\nu^2}{I_0/\rho}, \tag{30}$$

where I_0 represents the linear impulse of the vortex ring. The dimensionless speed of the vortex ring centroid is defined by

$$\overline{U}=U_{\mathrm{c}}\frac{(I_0/\rho)^{\frac{1}{2}}}{\nu^{\frac{3}{2}}}, \tag{31}$$

where U_{c} is the ring centroid velocity measured in the computational units.

We present the vorticity contours at $\bar{t}=6.75\times10^{-5}$, 7.48×10^{-5}, 8.21×10^{-5}, 9.06×10^{-5}, 10.03×10^{-5} and 11.85×10^{-5}. The vorticity contours shown in Fig. 2 match well those previously reported [13,14]. Even the subtle tail structures at $\bar{t}=10.03\times10^{-5}$ are well captured.

The predicted location of the vortex ring centroid velocity shown in Fig. 3 also compares well with the results of previous calculations. We note that the current calculation which was performed using the full three-dimensional representation of the ring structure matches more closely the two-dimensional spectral calculation results obtained by Stanaway et al. [14] than those reported in [13]. The ring maintains its two-

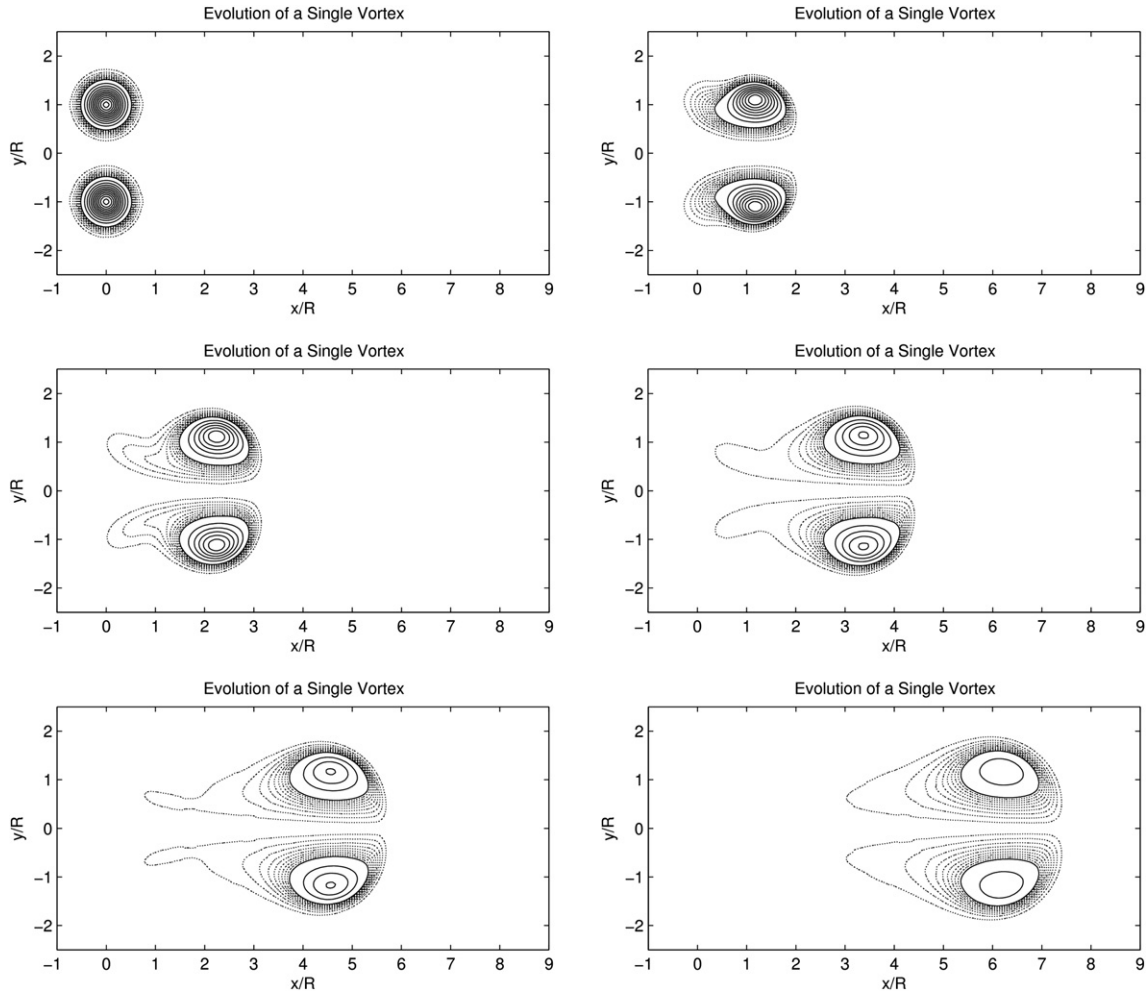

Fig. 2. Vorticity contours of the evolution of a single vortex ring for the times $\bar{t} = 6.75 \times 10^{-5}$, 7.48×10^{-5}, 8.21×10^{-5}, 9.06×10^{-5}, 10.03×10^{-5} and 11.85×10^{-5}. The vorticity difference between two solid lines is ten times higher as the vorticity difference between two dashed lines.

dimensionality during the simulation and hence it is possible to compare our three-dimensional results with the two-dimensional results. The results of Wee and Ghoniem [13] were obtained using a 20° section to reduce the number of computational elements (see also Ref. [15]). The current results were obtained for the full 360° ring representation. The number of vortex elements at the end of the simulation is around 3,500,000.

The use of our more efficient vortex-particle algorithm, instead of the vortex filament algorithm in [13], allows us to perform simulations with a smaller grid size. As a consequence, we have better accuracy and can accommodate up to 5 millions particles even with a serial implementation.

5. Buoyancy-driven flows

As mentioned earlier, the development of a fast treecode to compute the velocity and its gradient enable us to resurrect the idea of the transport element methods (TEM), in which the gradients of primitive variables are used as weights for computational particles, instead of the primitive variables. In the following, we demonstrate the capability of our combined strategy, i.e., vortex element/transport element scheme, by computing buoyancy-driven flows.

We consider the evolution of thermals. Single thermal spheres have been intensively studied and their behavior is now well known. An important theoretical and experimental work has been done by Scorer

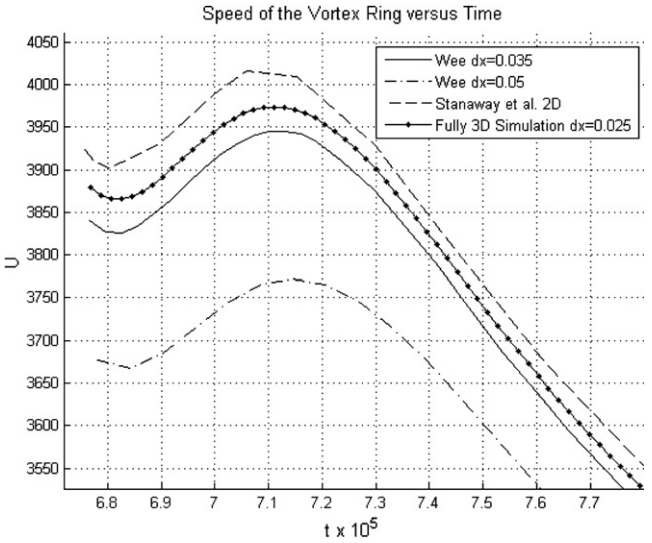

Fig. 3. The vortex ring velocity for the highest resolution fully 3-D simulation (solid curve) using the current vortex particle algorithm. The results obtained in 2-D by Stanaway et al. [14] are plotted in dashed lines, and those obtained by Wee and Ghoniem [13] for a 20° section simulation performed in parallel are plotted in solid lines.

[17,18], Wang [19], Lin [20], Turner [21,22], and Escudier and Maxworthy [23]. Numerical simulations on thermals have been initially performed by Andersdon [5] and Marcus and Bell [24] in 2D. More generally, two-dimensional studies of the Rayleigh-Taylor instability were conducted by Baker et al. [25], Kerr [26], Tryggvason [27] and Zufiria [28,29], using vortex methods. Three-dimensional simulations of buoyant bubbles were realized by Brecht and Ferrante [30,31] in the inviscid limit using a vortex-in-cell code. Walther and Koumoutsakos [32] extended the particle methods in 3D to the viscous case.

We first present the evolution of a single thermal sphere, and then nonlinear interactions between two thermal spheres will be shown.

5.1. Evolution of a thermal sphere

A sphere of hot air is placed in relatively cold ambient atmosphere, such that its center is initially at the origin. The radius of the sphere is used as the reference length scale for normalization, i.e., $R = 1$. Dynamically, the difference in temperature between the hot and cold air drives the sphere against gravity through buoyancy. This phenomenon can be kinematically described by the baroclinic generation of vorticity around the surface of the sphere.

The initial temperature profile is defined by the error function, i.e., $\theta(\mathbf{x}) = \frac{1}{2}\mathrm{erfc}\left(\frac{|\mathbf{x}|-R}{\delta_T}\right)$, where δ_T is the thickness of the temperature transition layer.The gradient profile is given by a Gaussian distribution, $\nabla\theta = -\frac{1}{\sqrt{\pi}\delta_T}\exp\left[-\left(\frac{|\mathbf{x}|-R}{\delta_T}\right)^2\right]\mathbf{e}_r$, where $\mathbf{e}_r$ represents the radial unit vector. The profiles are shown in Fig. 4. As described above, using gradients as weights for the particles, we only need to cover the support of the gradient. Gravity is pointed in the negative y-direction.

The parameters are chosen as follows:

$$Pr = v/\alpha = 1, \tag{32}$$

$$Re = \frac{U^* R}{v} = \frac{R\sqrt{gR}}{v} = 1000, \tag{33}$$

$$Gr = \frac{\rho_\infty g_r R^3 \Delta\rho}{\mu^2} \approx \frac{g_r R^3 \beta \Delta T}{v^2} = 5 \times 10^5, \tag{34}$$

$$Gr/Re^2 = \frac{1}{2}. \tag{35}$$

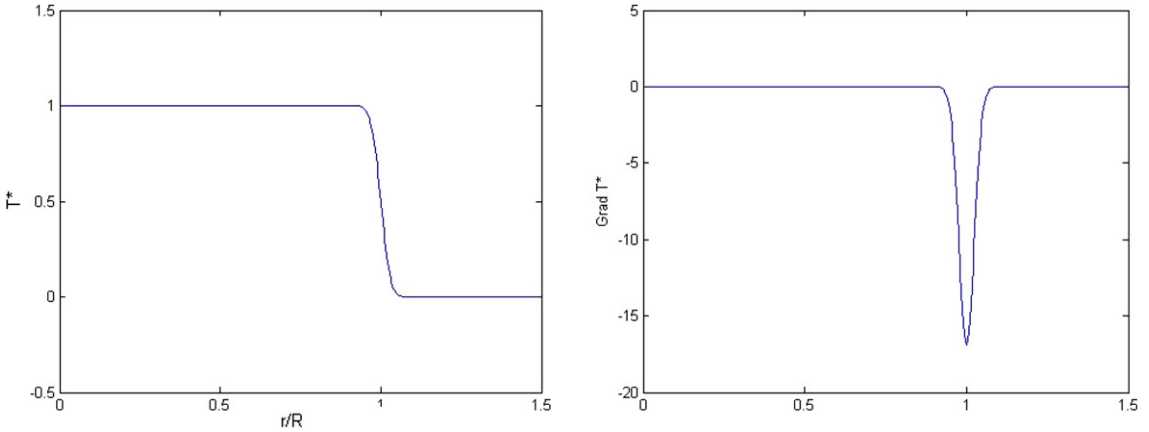

Fig. 4. View on a radial cut, of the value of the temperature distribution on the left (erfc), and the value of the temperature gradient distribution on the right (*Gaussian*), at $t = 0$ s.

The thickness of the temperature transition region is given by $\delta_T = 1/30$, i.e., the temperature difference is allowed to spread over 1/30 of the initial radius of the sphere before we start the simulations. The grid size for diffusion $\Delta x = 0.05$ is and the time step size is $\Delta t = 0.125$. Fig. 5 shows the evolution of both the temperature and the vorticity on the left hand side and the right hand side respectively. The temperature is recovered from the gradient elements, using the method described in Appendix for fast summation over gradient elements:

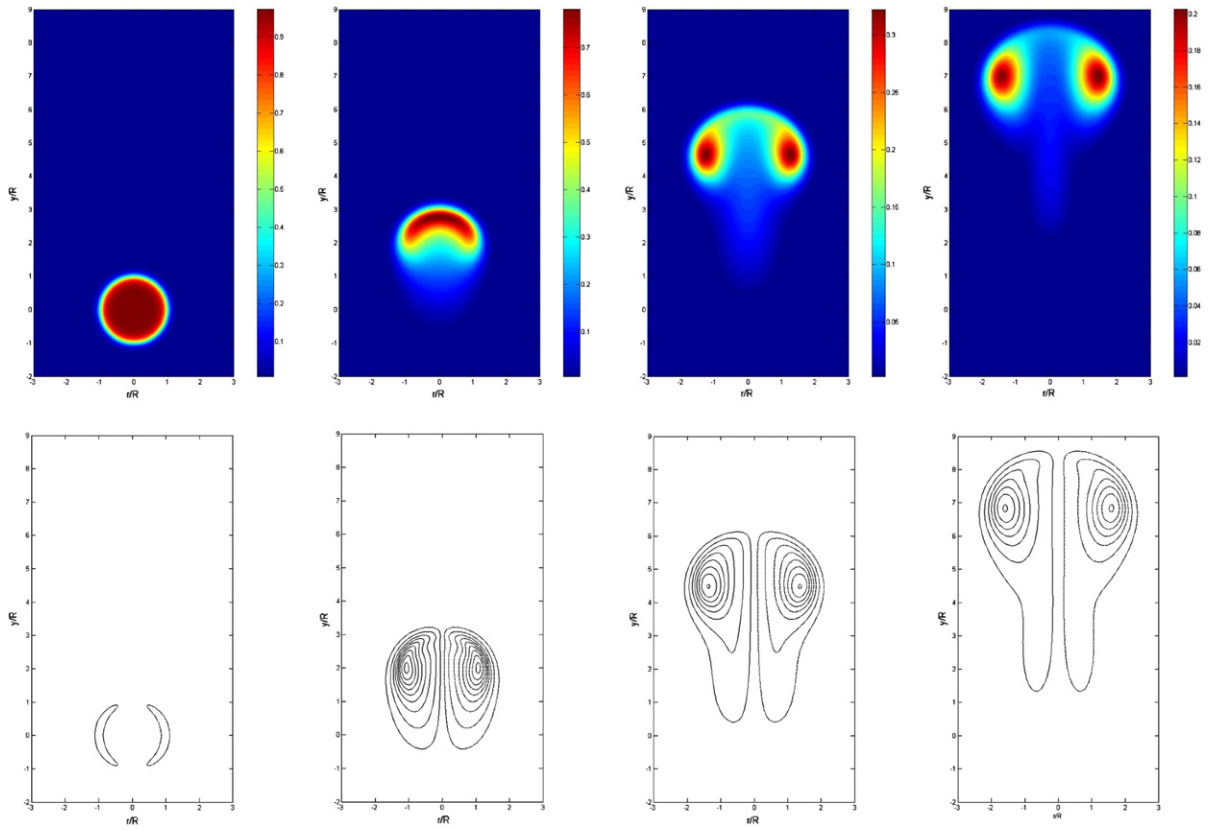

Fig. 5. Evolution of the buoyant sphere, 2D-cut (3D simulation); temperature contours on the top and vorticity contours on the bottom, for the times $\bar{t} = 0.125$, 2.4, 4.8 and 7.2. The two first contour values are 0.1 and 0.33, then their values vary linearly with an increment of 0.33.

$$\theta(\mathbf{x}) = -\sum_j \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\zeta}_j) \cdot \mathbf{G}_j \tag{36}$$

As expected, the sphere is driven against gravity through buoyancy. The vorticity generated on both sides of the sphere rolls up forming a complex ring structure.

A convergence study is performed by repeating the same simulations for different grid sizes, $\Delta x = 0.025$, 0.035, 0.05 and 0.1. The corresponding time step size is determined by $c_\omega = c_g = 2^{-1/2}$, which represents the ratio between the diffusion length scale and the grid size. The temperature is computed on a two-dimensional Cartesian grid, $\Delta x = 0.025$, by performing our fast summation over all the computational elements. The temperature centroid is defined as

$$y_T = \frac{\int \theta y \mathrm{d}V}{\int \theta \mathrm{d}V}. \tag{37}$$

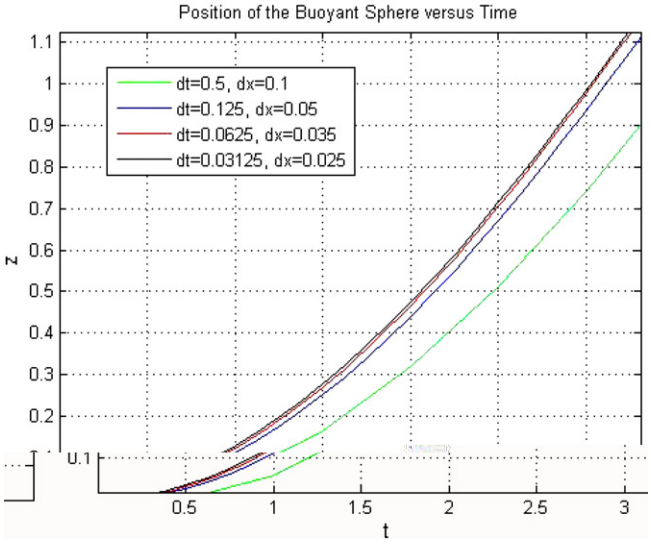

Fig. 6. Position of the temperature center of the buoyant sphere for different resolutions.

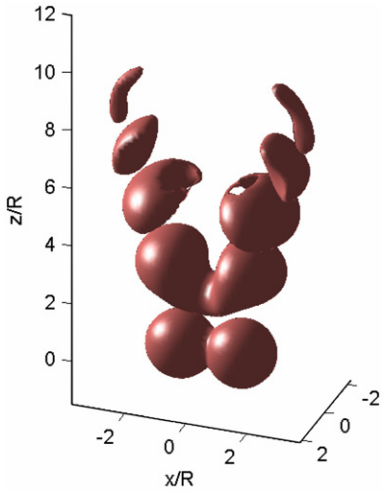

Fig. 7. Temperature isosurface, $\theta = 0.3$, of the evolution in time of two thermal spheres of initial radius $R = 1$, for $\bar{t} = 0$, 6.5, 10.5, 14.5 and 19.

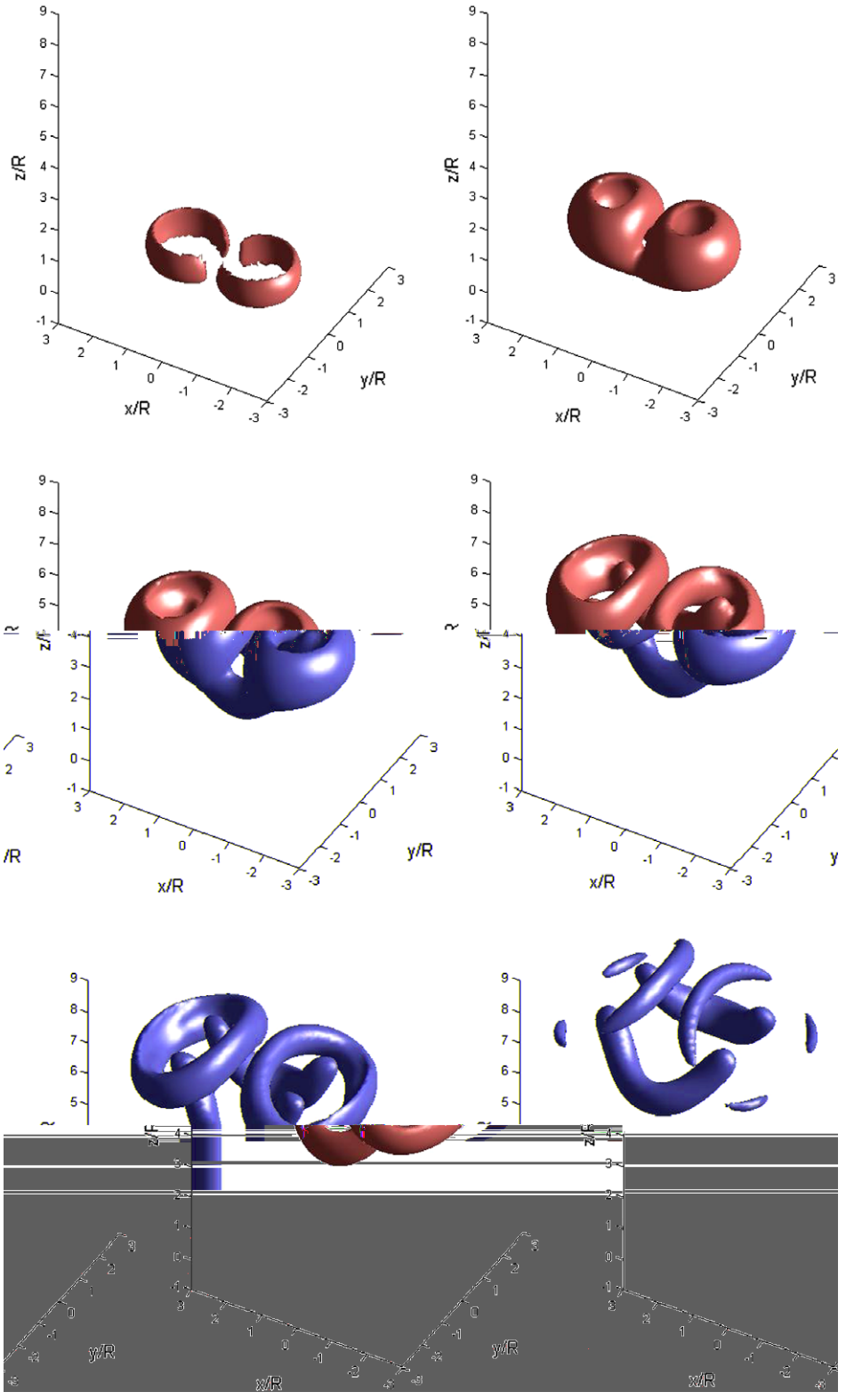

Fig. 8. Vorticity isosurface, $|\omega| = 1.2$, of the evolution in time of two thermal spheres of initial radius $R = 1$, for $\bar{t} = 1.3$, 3.3, 6.9, 8.9, 11.9 and 17.1.

The position of the buoyant sphere temperature centroid is plotted in Fig. 6 for different resolutions. The error is defined as

$$\|Error\|_{L_2} = \sqrt{\frac{1}{N^2}\sum_{t=0}^{N\Delta t}(y_T(t,\Delta x) - y_T(t,\Delta x = 0.025))^2}, \tag{38}$$

where N is the number of time steps. The highest resolution simulation, was compared with the three other simulations obtained using coarser grids. The order of convergence is 1.45. Note that we use a second-order scheme for the convection step, while the diffusion and the baroclinic generation of vorticity are first-order. In all cases the flows remains essentially two-dimensional.

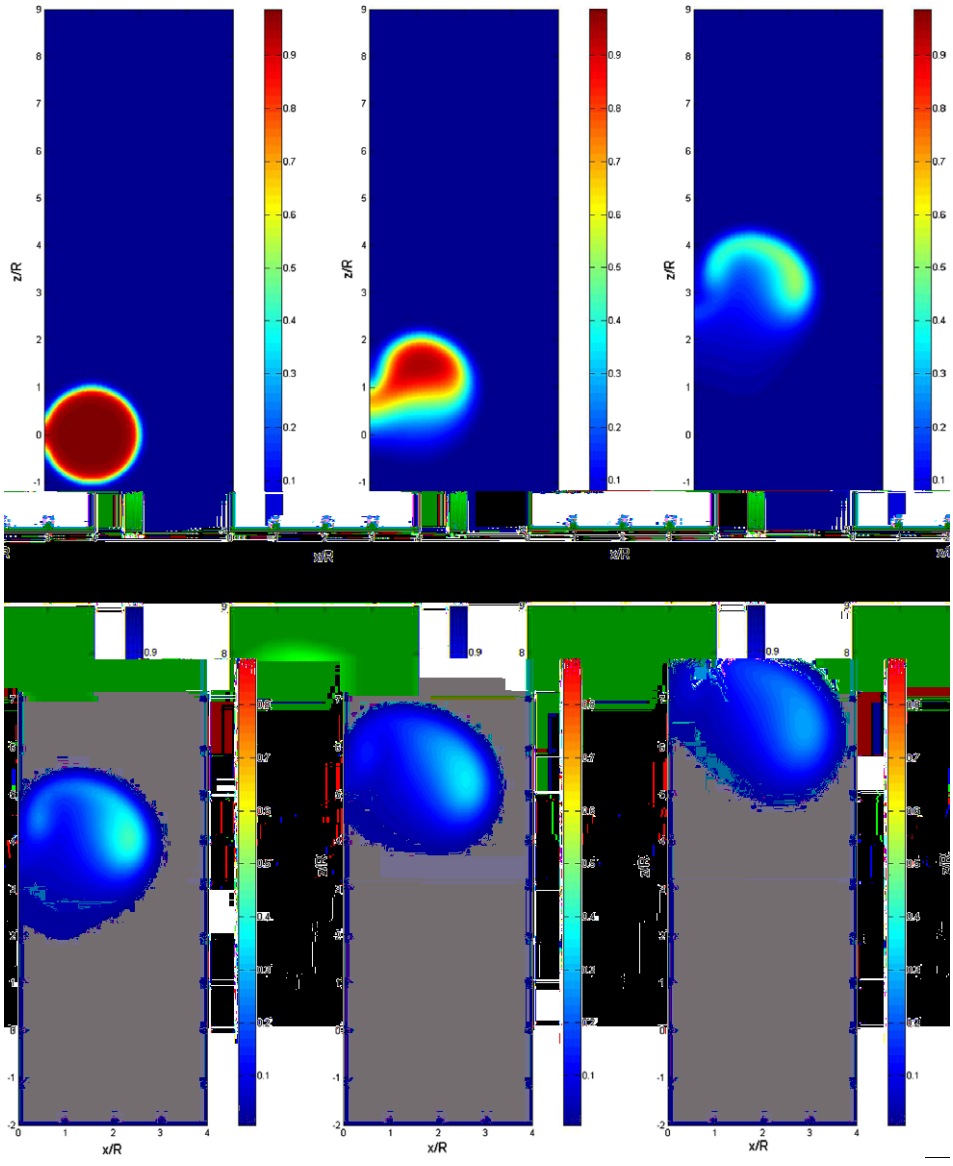

Fig. 9. Temperature contours of the evolution in time of two thermal spheres of initial radius $R = 1$, for $\bar{t} = 0$, 3.3, 6.9, 8.9, 11.9 and 15.1.

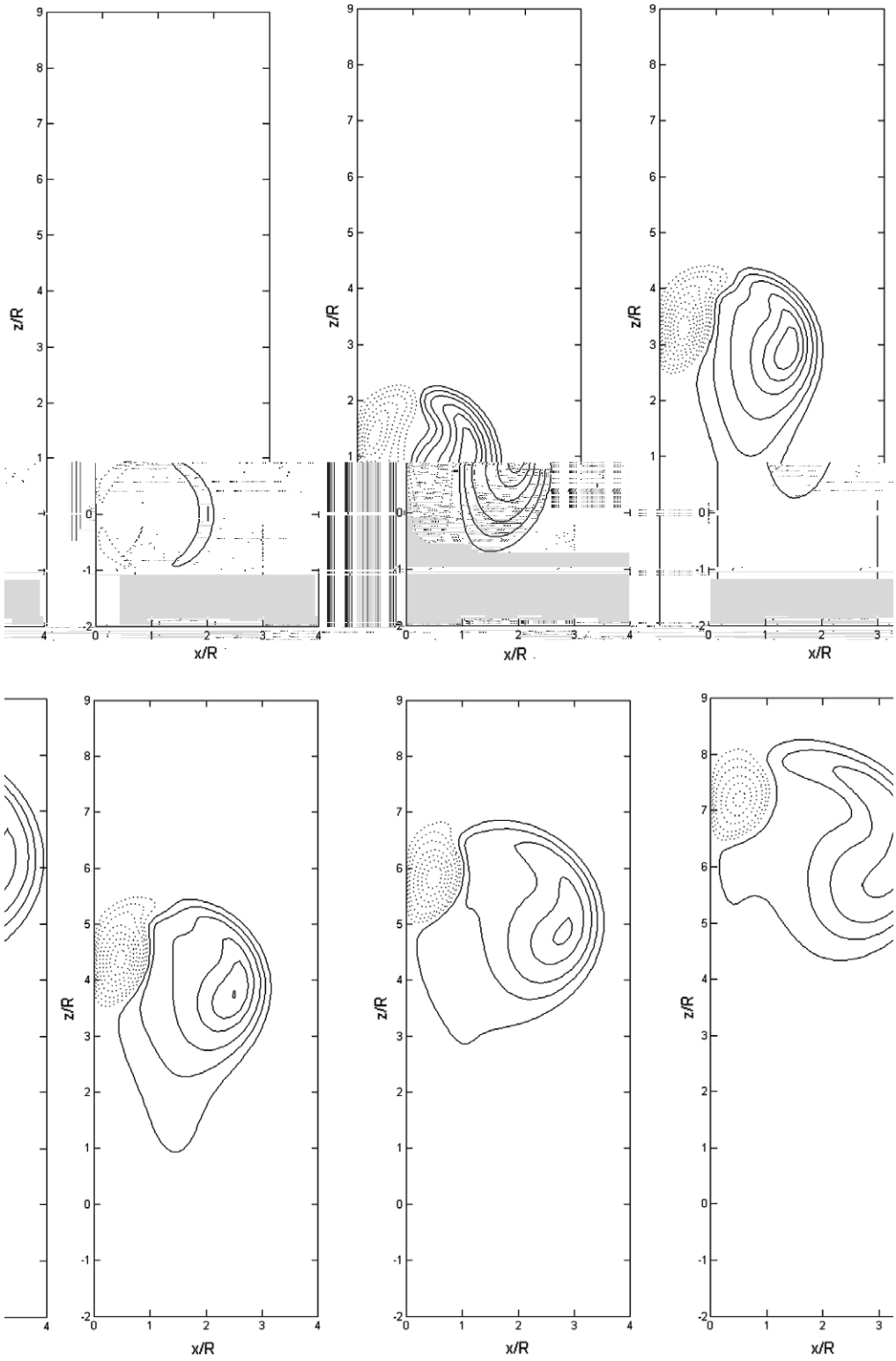

Fig. 10. Vorticity contours of the evolution in time of two thermal spheres of initial radius $R = 1$, for $\bar{t} = 0.125$, 3.3, 6.9, 8.9, 11.9 and 15.1. The three first contour values are 0.1, 0.25 and 0.5, then their values vary linearly with an increment of 0.5.

5.2. Interactions of two thermal spheres

5.2.1. Side-by-side interaction

Fig. 7 shows another sample calculation used to demonstrate the three-dimensional capabilities of the code: two hot-air spheres are evolving under a gravity field in the z-direction. In this case the two spheres are initially placed side-by-side. We again use the same parameters that we used in the single buoyant sphere simulation but allow the spatial grid size to grow in time such that $\Delta x = 0.035$ for $t \in [0, 3.75]$, $\Delta x = 0.05$ for $t \in [3.75, 12.37]$ and $\Delta x = 0.07$ for $t > 12.37$. The corresponding time step size is determined by $c_\omega = c_g = 2^{-1/2}$. Figs. 7 and 8 show 3D plots of the temperature contour, $\theta = 0.3$, and the vorticity isosurface, $|\omega| = 1.2$. These figures show that, due to diffusion, the distinction between the two spheres is lost and a continuous complex structure is formed a short distance into their vertical rise. The distortion of the temperature

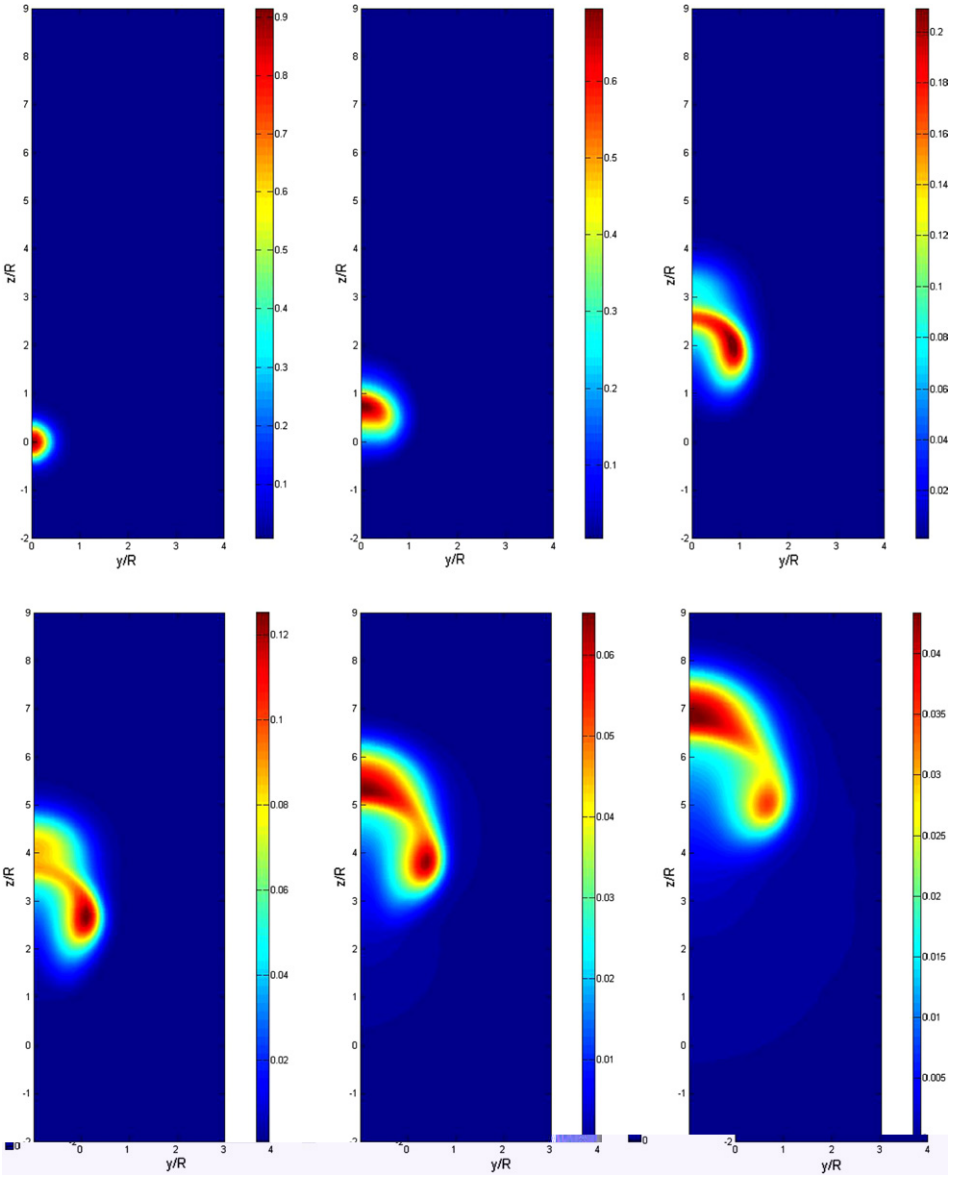

Fig. 11. Temperature contours of the evolution in time of two thermal spheres in the y–z plane at $x = 0$, for $\bar{t} = 0$, 3.3, 6.9, 8.9, 11.9 and 15.1.

isosurfaces because of the mutual interactions between the two sphere leads to the formation of a complex tangle of vorticity structures. The results are plotted in two-dimensional cuts across the vertical plane of the initial centers of the two spheres in Figs. 9 and 10 respectively, to show the strong distortion of the temperature contours, followed by the inter-diffusion between the neighbors spheres and the corresponding vorticity field in that plane.

The vortical structures formed due to the interaction between generation, convection and diffusion are well illustrated in Fig. 8. Here, we observed two distinct rings whose individual structures resemble those observed in the single sphere simulations only at the very early stages. As these rings evolve, a mutual distortion of the overall structure is observed starting at $t = 1.3$ in Fig. 8. This distortion is most pronounced in the contortion of the initial two rings upwards, where they intersect close to the anti-symmetry plane, and the formation of two new crescent shaped structures between these rings. Although each hot sphere leads to the formation of a single vortex ring, as seen before, and the two spheres initially form two side-by-side vortex rings, the similarity ends here, as explained next.

It is interesting to notice that the underlying physics here is much different from what is observed in the case of the interaction of two vortex rings. The general behavior of two colliding side-by-side vortex rings is well known and can be seen in [13]. In the case of the side-by-side ring propagation, the following features are observed. At the early stages, before the inner vorticity cores inter-diffuse and values with opposite signs annihilate each other, or before the inner cores connect, the downward motion near the anti-symmetry plane pushes the inner cores downward with respect to the outer cores. At the later stages, however, after the two inner cores connect and their vorticity dissipates by inter-diffusion, the strength of the inner cores becomes weaker than those associated with the outer cores. The motion is now reversed and the inner cores move upwards with respect to those of the outer cores.

In the case of the side-by-side hot spheres, the two rings that form early in the evolution are contorted upwards near the anti-symmetry plane from the very early stages, as seen at $t = 3.3$. In this case, the

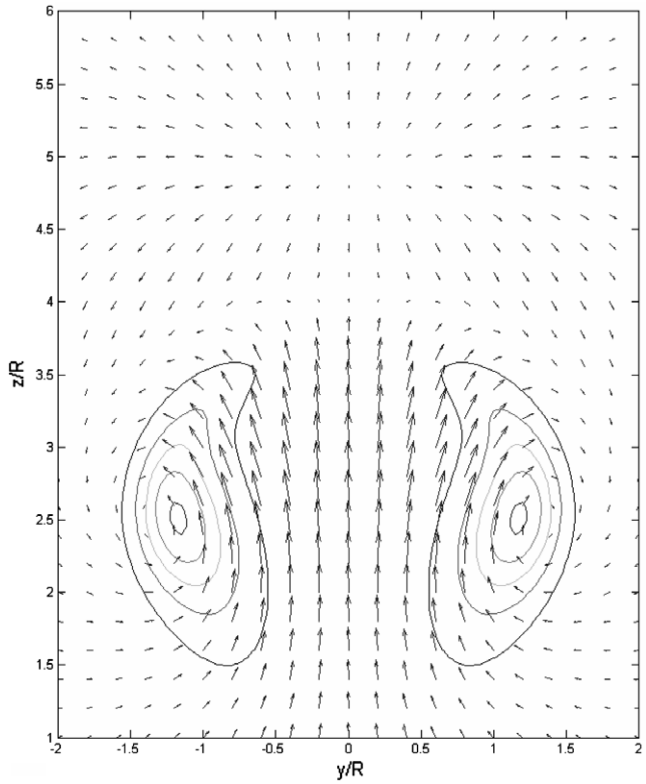

Fig. 12. Velocity field induced by the vortical structures in the yz-plane at $x = 0$ at time $\bar{t} = 8.9$. The contours correspond to the vorticity norm, their values vary linearly from 0.5 to 2.5, with an increment of 0.5.

reconnection between the two hot spheres reduces the vorticity generation rate in the inner cores below that at the outer cores. This is shown by the empty zone at $t = 1.3$. Moreover, the diffusion of the opposite signs vorticity generated within the inner cores further weakens their impact with respect to that of outer cores. The formation of relatively uniform temperature zones near the anti-symmetry plane is shown in Fig. 9 as the two spheres connect/diffuse. Meanwhile, higher temperature zones persist at the outer cores even at the later stages. The impact of this distortion on the vorticity field is seen in Fig. 10, where from the early stages of the simulation, the outer vorticity cores are larger and the inner cores are driven upwards from $t \geqslant 0$.

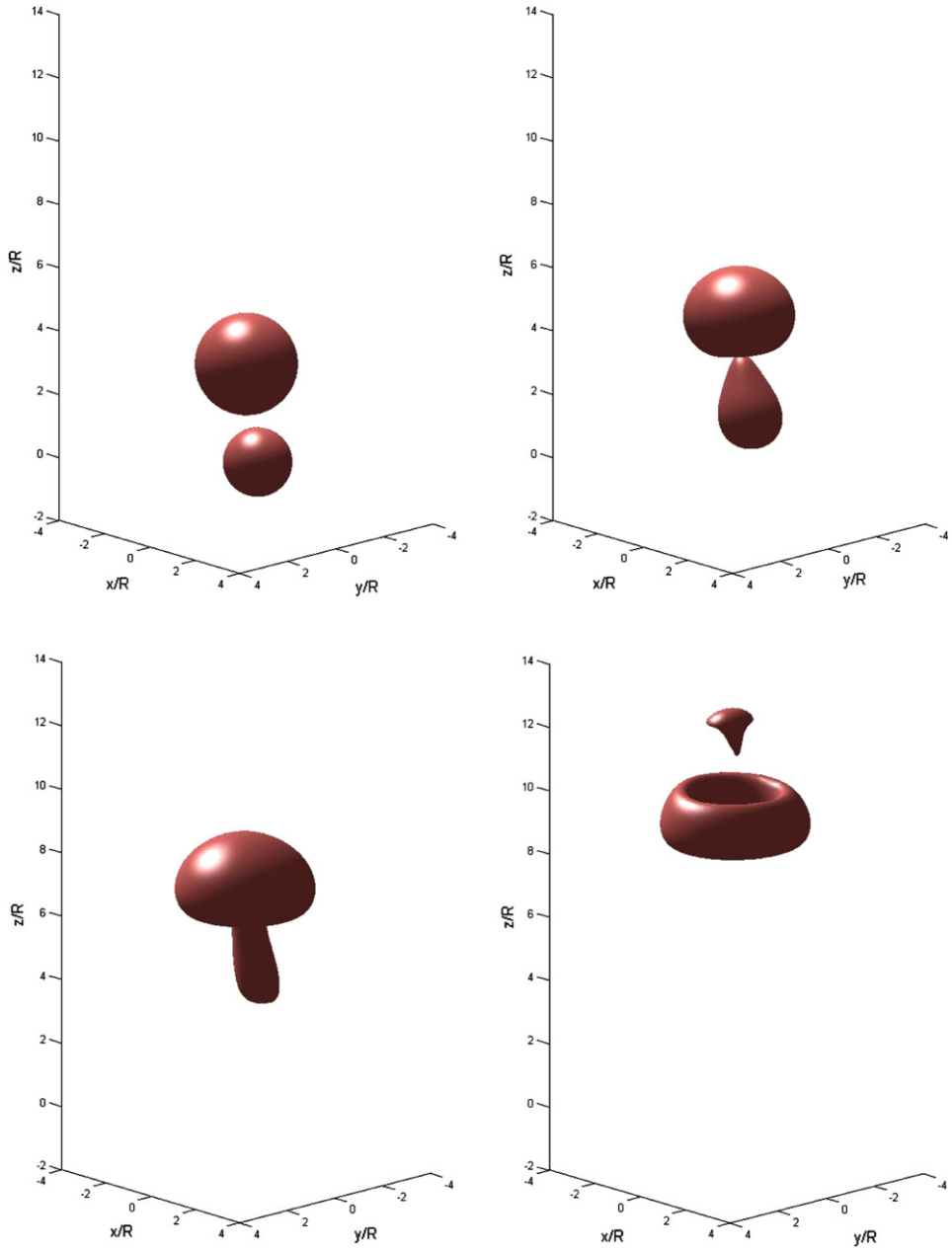

Fig. 13. Evolution of two thermal spheres; temperature isosurface, $\theta = 0.3$, for $\bar{t} = 0$, 3.5, 7 and 10.5. The upper sphere with a radius of 1.5 centered at (0,0,3), and the lower one with a radius of 1 centered at $(0.5, 0, 0)$. The two spheres are initially at the same temperature.

Parallel to the formation and distortion of the two side-by-side rings as the two hot spheres rise, we observe the formation of two crescent shaped vortical structures that "hang" below the two rings as they propagate upwards. These two crescents form as the two sphere interconnect from below, as seen in Fig. 9 at $t = 3.3$. The baroclinic generation associated with the bridge between the two initial spheres is consistent with the temperature distribution within this bridge, as shown in Fig. 11. Note that the vorticity forms at the interface between the hot fluid originally in the spheres and the cold fluid outside.

To quantify the vorticity within these crescent shaped structures and show their impact on the velocity field, we plot their vorticity distribution and the total velocity on a y–z plane located half way between the original two spheres in Fig. 12. The plot shows that these two structures contribute significantly to the upward motion

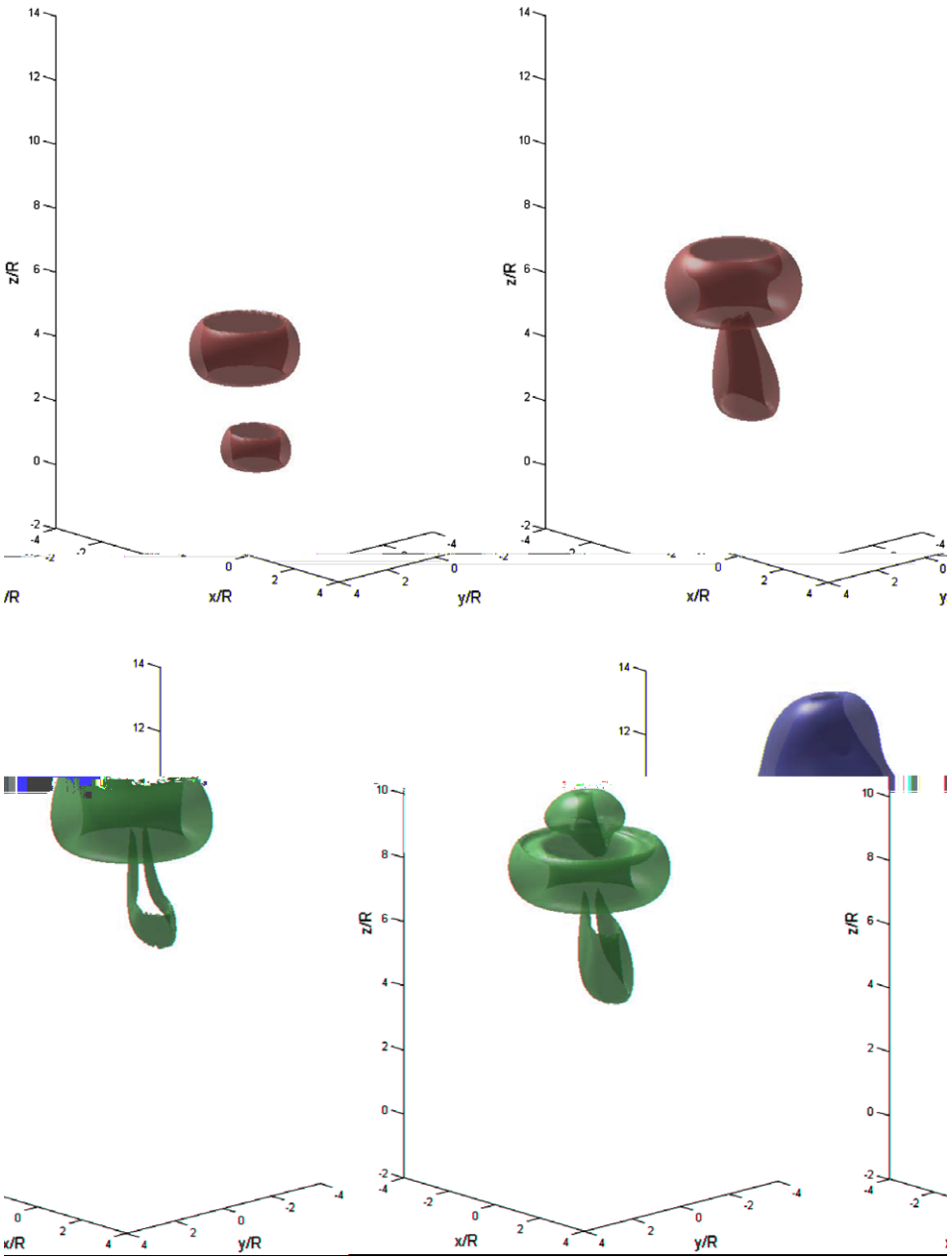

Fig. 14. Evolution of two thermal spheres; vorticity isosurface, $|\omega| = 1.4$, for $\bar{t} = 2.5$, 5.5, 8.5 and 11.25.

at the anti-symmetry plane. It is interesting to observe that, at the late times, the vorticity associated with the original two rings decay, while that contained in these two crescents persist. This is confirmed by Fig. 8, at $t = 17.1$.

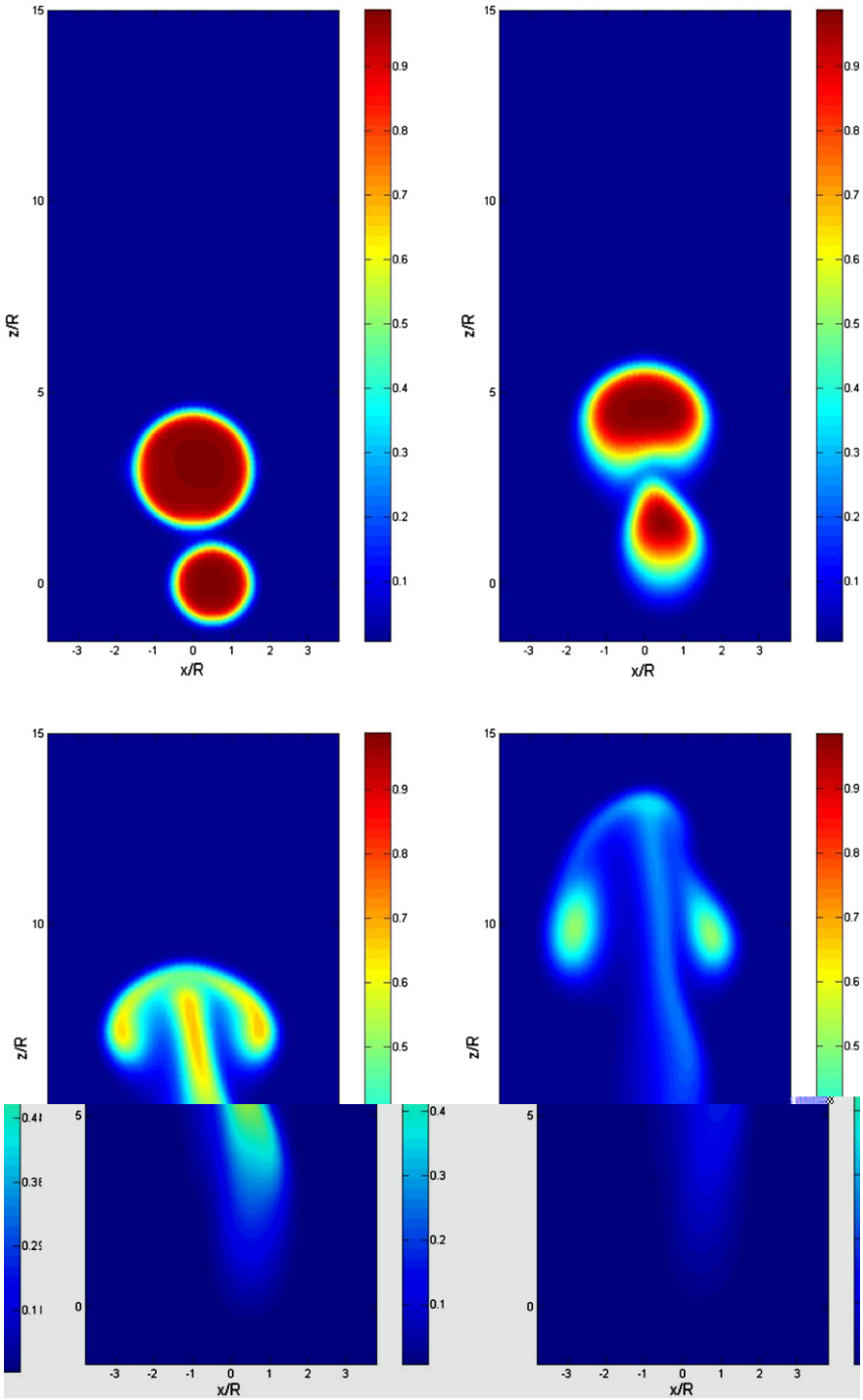

Fig. 15. Evolution of two thermal spheres; temperature contours, for $\bar{t} = 0$, 3.5, 7 and 10.5. The upper bubble with a radius of 1.5 centered at $(0, 0, 3)$, and the lower one with a radius of 1 centered at $(0.5, 0, 0)$. The two bubbles are initially at the same temperature.

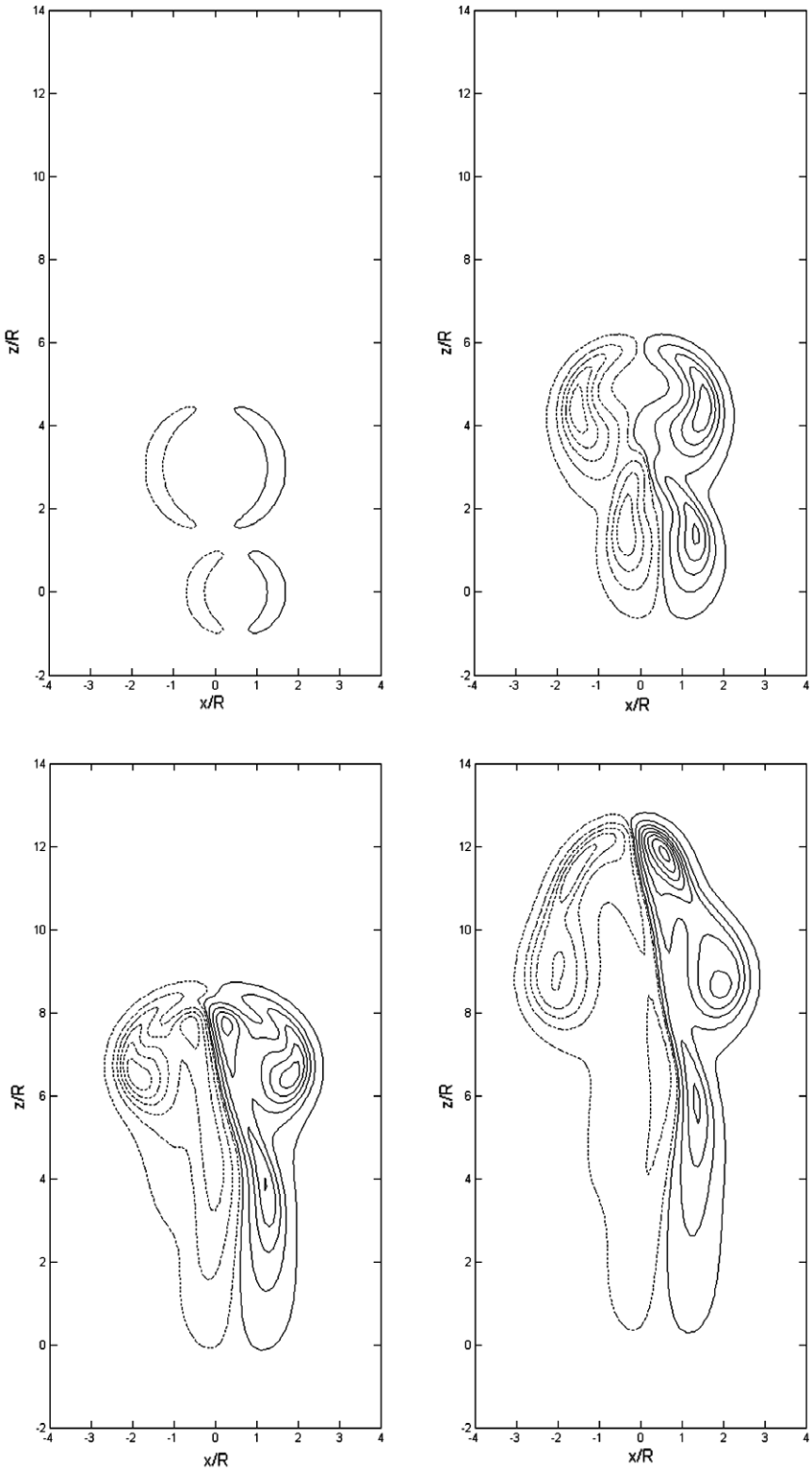

Fig. 16. Evolution of two thermal spheres; vorticity contours for $\bar{t} = 0$, 3.5, 7 and 10.5. The two first contour values are 0.1 and 0.5, then their values vary linearly with an increment of 0.5.

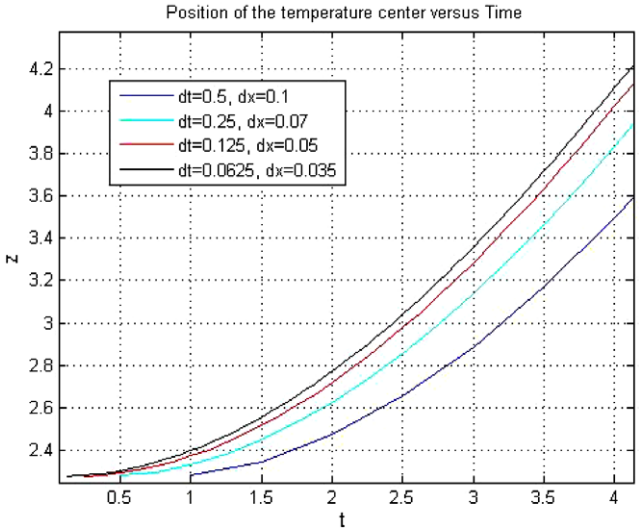

Fig. 17. Position of the temperature center of the buoyant spheres for different resolutions.

5.2.2. Two spheres with different sizes

Another calculation that demonstrates the three-dimensional capability of our method is that shown in Figs. 13 and 14, where initially the upper sphere has a radius of 1.5 and the lower one has a radius of 1. To manifest the three-dimensionality, the centers of the two spheres are shifted in the lateral direction with respect to the vertical (gravity) direction. Hence, the upper sphere is centered at $(0,0,3)$, and the lower one at $(0.5,0,0)$. We use the same numerical parameters that we used in the single buoyant sphere simulation. The temperature isosurface, $\theta = 0.3$, and vorticity isosurface, $|\omega| = 1.4$, are plotted in 3D in Figs. 13 and 14 respectively. The temperature and vorticity contours are plotted in Figs. 15 and 16. The two spheres are initially at the same temperature. Initially, the vorticity generated on the sides of both spheres roll up forming a vortex ring, as it was the case for the buoyant sphere case. However, soon after the formation of the initial vorticity, the two newly formed rings exhibit the traditional vortex ring leap frogging mechanism. The initial eccentricity augments this motion.

The results show that the asymmetry introduced by the eccentricity persists, as manifested by the motion of the smaller sphere towards the left while it passes through the larger sphere. The material in the smaller sphere is drawn into a long thin structure by the stronger vortical structure formed by the larger sphere. Meanwhile, as that structure "punches" through the larger sphere, it forces more of its fluid to move towards the left side.

A convergence study is performed by repeating the same simulations for different grid sizes, $\Delta x = 0.035$, 0.05, 0.07 and 0.1. The corresponding time step sizes are determined by $c_\omega = c_g = 2^{-1/2}$. The temperature centroid is defined in (37). The position of the buoyant spheres temperature center is plotted in Fig. 17. The error in the temperature centroid is defined in (38). The order of convergence is found to be 1.45. It is similar to what was found previously.

6. Conclusion

Vortex methods have been used in the simulation of high Reynolds number complex flows, especially when fast transitions and strong distortion of the underlying vortical structure of the flow are expected. The Lagrangian, self-adaptive nature of the calculations makes it possible to resolve strong gradients wherever and whenever they arise, while maintaining a coarse resolution when uniform zones continue to exist. For this reason, vortex methods have been particularly successful in resolving the evolution of shear layers. In cases when the flow is driven by body forces, that is, when the vorticity is continuously generated by the interaction between the density gradients and the pressure gradients, it is important that the vorticity source term is also evaluated accurately. Since flow gradients are involved in computing the source terms, it is important to apply

compatible schemes in simulating the flow dynamics and the evolution of scalar gradients. The method presented in this paper achieves this compatibility.

Solution of a number of buoyancy driven generic flows in three dimensions demonstrates the success of the method in resolving the temperature gradients and the corresponding vorticity structures, and in particular show its convergence. Complex vortical structures that arise in the early and late stages were reproduced.

The method shows sub quadratic convergence because the source terms integration is first-order. However, this is not an inherent limitation and one can improve the convergence order by applying second-order integration for the source terms. Second-order interpolation kernels are also available for diffusion, and high order splitting can be used. The method is also being used to simulating reacting flow, and results will be published soon.

Acknowledgment

This work was supported by a grant from the US Department of Energy, Office of Science, MICS.

Appendix A. Multi-purpose adaptive treecode

A.1. Problem definition

Lagrangian simulations using vorticity or gradients as particle weights require the solution of several differential equations that model the convection and source terms in the original conservation equations, to update the field. One after needs to compute the vortical velocity, $\mathbf{u}_\omega$, from the vorticity field, $\boldsymbol{\omega}$. One also needs to obtain the expansion velocity, $\mathbf{u}_\varepsilon$, from the divergence field, ε, generated by the volumetric expansion of material elements due to, e.g., a chemical reaction. In the case of using gradients for the primitive variable, e.g. in the transport element methods, there is also a need to recover the conserved scalar field, s (or θ in the main texts), from the information on its gradient, $\mathbf{g}$. Additionally, the evolutions of the weights of vortex elements and/or transport elements rely on the information regarding the gradient of the velocity field. Thus, it is necessary to compute $\nabla\mathbf{u}_\omega$ and $\nabla\mathbf{u}_\varepsilon$ simultaneously from $\boldsymbol{\omega}$ and ε directly.

In summary, the problem can be restated as follows. During each time step of the simulation, it is necessary to invert one or many of the following equations:

$$\boldsymbol{\omega} = \nabla \times \mathbf{u}_\omega, \tag{A1.1}$$

$$\varepsilon = \nabla \cdot \mathbf{u}_\varepsilon, \tag{A1.2}$$

$$\mathbf{g} = \nabla s. \tag{A1.3}$$

That is, knowing $\boldsymbol{\omega}$, ε, and $\mathbf{g}$, we need to calculate $\mathbf{u}_\omega$, $\mathbf{u}_\varepsilon$, and their gradients, as well as s. $\boldsymbol{\omega}$, ε, and $\mathbf{g}$ are all discretized into Lagrangian computational elements or particles:

$$\boldsymbol{\omega}(\mathbf{x},t) \approx \sum_{j=1}^{N} \mathbf{W}_j(t) f_\delta(\mathbf{x} - \boldsymbol{\chi}_j(t)) \quad \text{(vortex elements)}; \tag{A1.4}$$

$$\varepsilon(\mathbf{x},t) \approx \sum_{j=1}^{N} E_j(t) f_\delta(\mathbf{x} - \boldsymbol{\chi}_j(t)) \quad \text{(divergence elements)}; \tag{A1.5}$$

$$\mathbf{g}(\mathbf{x},t) \approx \sum_{j=1}^{N} \mathbf{G}_j(t) f_\delta(\mathbf{x} - \boldsymbol{\chi}_j(t)) \quad \text{(transport elements)}; \tag{A1.6}$$

The weights $\mathbf{W}_j$, E_j, and $\mathbf{G}_j$ correspond to the vorticity, divergence, and gradient assigned to each computational element, i.e., $\mathbf{W}_j = [\boldsymbol{\omega} \mathrm{d}V]_j$, $E_j = [\varepsilon \mathrm{d}V]_j$, and $\mathbf{G}_j = [\mathbf{g} \mathrm{d}V]_j$. $\boldsymbol{\chi}_j$ is the location of the jth particle. f_δ is a desingularized radially symmetric core function of radius δ, given by $f_\delta(\mathbf{x}) \equiv \delta^{-3} f(|\mathbf{x}|/\delta)$. The function f must be smooth and rapidly decaying at infinity. In this work, we use the low-order algebraic core function [1]

$$f(\rho) = \frac{3}{4\pi} \frac{1}{(1+\rho^2)^{5/2}}. \tag{A1.7}$$

In $\mathbf{R}^3$, the solution of (A1.1) is given by the Biot–Savart law,

$$\mathbf{u}_\omega(\mathbf{x}) = -\frac{1}{4\pi} \int \frac{\mathbf{x}-\mathbf{y}}{|\mathbf{x}-\mathbf{y}|^3} \times \boldsymbol{\omega}(\mathbf{y}) \mathrm{d}\mathbf{y}. \tag{A1.8}$$

With (A1.4), (A1.8) can be rewritten as follows:

$$\mathbf{u}_\omega(\mathbf{x}) = \sum_{j=1}^{N} \mathbf{K}_\delta(\mathbf{x}\mathbf{1}$$

the smallest leaf size, N_0, which is predefined by the user, the process terminates and returns the tree structure.

Once the tree is constructed, (A1.9), (A1.11)–(A1.14) are rewritten in the following form:

$$\mathbf{u}_\omega(\mathbf{x}) = \sum_c \sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\chi}_j) \times \mathbf{W}_j, \tag{A1.15}$$

$$\nabla \mathbf{u}_\omega(\mathbf{x}) = \sum_c \sum_{j=1}^{N_c} \nabla_{\mathbf{x}} \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\chi}_j) \times \mathbf{W}_j, \tag{A1.16}$$

$$\mathbf{u}_\varepsilon(\mathbf{x}) = -\sum_c \sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\chi}_j) E_j, \tag{A1.17}$$

$$\nabla \mathbf{u}_\varepsilon(\mathbf{x}) = -\sum_c \sum_{j=1}^{N_c} \nabla_{\mathbf{x}} \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\chi}_j) E_j, \tag{A1.18}$$

$$s(\mathbf{x}) = -\sum_c \sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\chi}_j) \cdot \mathbf{G}_j, \tag{A1.19}$$

where c denotes a cluster containing particles. The particle–cluster interactions are evaluated either by using Taylor approximation or by direct summation, following the same strategy described in [33]. The procedure uses a complex combination of theoretical error estimates and empirical computational time estimates to determine the best order of the approximation and the best size of the cluster. A parallel implementation of the same algorithm that uses k-means clustering to distribute the load among a number of processors is documented in [34].

A.3. Taylor approximation

To derive a Taylor approximation for a particle–cluster interaction, $\mathbf{K}_\delta(\mathbf{x}, \mathbf{y})$ in (A1.15) is expanded in a Taylor series with respect to $\mathbf{y}$, around the cluster center $\mathbf{y}_c$, such that

$$\begin{aligned}
\sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\chi}) \times \mathbf{W}_j &= \sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}, \mathbf{y}_c + (\boldsymbol{\chi}_j - \mathbf{y}_c)) \times \mathbf{W}_j \\
&= \sum_{j=1}^{N_c} \sum_{\mathbf{k}} \frac{1}{\mathbf{K}!} D_y^{\mathbf{k}} \mathbf{K}_\delta(\mathbf{x}, \mathbf{y}_c)(\boldsymbol{\chi}_j - \mathbf{y}_c)^{\mathbf{k}} \times \mathbf{W}_j \\
&= \sum_{\mathbf{k}} \mathbf{a}_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c) \times \mathbf{m}_{\mathbf{k}}^{\omega}(c).
\end{aligned} \tag{A1.20}$$

Here, $\mathbf{a}_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c)$ is the $\mathbf{k}$th Taylor coefficient of $\mathbf{K}_\delta(\mathbf{x}, \mathbf{y})$ at $\mathbf{y} = \mathbf{y}_c$:

$$\mathbf{a}_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c) = \frac{1}{\mathbf{k}!} D_{\mathbf{y}}^{\mathbf{k}} \mathbf{K}_\delta(\mathbf{x}, \mathbf{y}_c), \tag{A1.21}$$

and $\mathbf{m}_{\mathbf{k}}^{\omega}(c)$ is the $\mathbf{k}$th moment of the vortex elements in cluster c about its center $\mathbf{y}_c$:

$$\mathbf{m}_{\mathbf{k}}^{\omega}(c) = \sum_{j=1}^{N_c} (\boldsymbol{\chi}_j - \mathbf{y}_c)^{\mathbf{k}} \mathbf{W}_j. \tag{A1.22}$$

$\mathbf{k} = (k_1, k_2, k_3)$ is an integer multi-index with $k_i \geqslant 0$, and $\mathbf{k}! = k_1! k_2! k_3!$. For $\mathbf{x} \in \mathbf{R}^3$, $\mathbf{x}^{\mathbf{k}}$ is interpreted in a standard way, i.e., $x_1^{k_1} x_2^{k_2} x_3^{k_3} \in \mathbf{R}$. The infinite series in (A1.20) is approximated by a finite sum,

$$\sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\chi}_j) \times \mathbf{W}_j \approx \sum_{|\mathbf{k}| \leqslant p} \mathbf{a}_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c) \times \mathbf{m}_{\mathbf{k}}^{\omega}(c), \tag{A1.23}$$

where $|\mathbf{k}| = k_1 + k_2 + k_3$. The order of the approximation, p, must be chosen so that the error due to the truncation remains small.

For the particle–cluster interactions in (A1.16), a similar series with different set of Taylor coefficients is developed.

$$\sum_{j=1}^{N_c} \nabla_{\mathbf{x}} \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\chi}_j) \times \mathbf{W}_j = \sum_{j=1}^{N_c} \nabla_{\mathbf{x}} \mathbf{K}_\delta(\mathbf{x}, \mathbf{y}_c + (\boldsymbol{\chi}_j - \mathbf{y}_c)) \times \mathbf{W}_j$$
$$= \sum_{j=1}^{N_c} \sum_{\mathbf{k}} \frac{1}{\mathbf{K}!} D_y^{\mathbf{k}} \nabla_{\mathbf{x}} \mathbf{K}_\delta(\mathbf{x}, \mathbf{y}_c)(\boldsymbol{\chi}_j - \mathbf{y}_c)^{\mathbf{k}} \times \mathbf{W}_j = \sum_{\mathbf{k}} \mathbf{c}_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c) \times \mathbf{m}_{\mathbf{k}}^{\omega}(c), \tag{A1.24}$$

where $\mathbf{c}_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c)$ is the $\mathbf{k}$th Taylor coefficient of $\nabla_{\mathbf{x}} \mathbf{K}_\delta(\mathbf{x}, \mathbf{y})$ at $\mathbf{y} = \mathbf{y}_c$:

$$\mathbf{c}_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c) = \frac{1}{\mathbf{k}!} D_{\mathbf{y}}^{\mathbf{k}} \nabla_{\mathbf{x}} \mathbf{K}_\delta(\mathbf{x}, \mathbf{y}_c), \tag{A1.25}$$

which yields a three-by-three matrix for each set of $(\mathbf{x}, \mathbf{y}_c)$. Again, the infinite sum is truncated into a finite one up to an appropriate order p:

$$\sum_{j=1}^{N_c} \nabla_{\mathbf{x}} \mathbf{K}_\delta(\mathbf{x}, \boldsymbol{\chi}_j) \times \mathbf{W}_j \approx \sum_{|\mathbf{k}| \leqslant p} \mathbf{c}_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c) \times \mathbf{m}_{\mathbf{k}}^{\omega}(c). \tag{A1.26}$$

To evaluate either (A1.23) or (A1.26), we need the Taylor coefficients, i.e., $\mathbf{a}_{\mathbf{k}}$ or $\mathbf{c}_{\mathbf{k}}$. An efficient method to obtain $\mathbf{a}_{\mathbf{k}}$ was proposed in [33]. The Rosenhead–Moore kernel (A1.10) is given by the gradient of the Plummer potential:

$$\mathbf{K}_\delta(\mathbf{x}, \mathbf{y}) = -\nabla_{\mathrm{y}} \phi_\delta(\mathbf{x}, \mathbf{y}), \tag{A1.27}$$

where

$$\phi_\delta(\mathbf{x}, \mathbf{y}) = \frac{1}{4\pi} \frac{1}{(|\mathbf{x} - \mathbf{y}|^2 + \delta^2)^{\frac{1}{2}}}. \tag{A1.28}$$

We set the $\mathbf{k}$th Taylor coefficient of $\phi_\delta(\mathbf{x}, \mathbf{y})$ at as $\mathbf{y} = \mathbf{y}_c$ as

$$b_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c) = \frac{1}{\mathbf{k}!} D_{\mathbf{y}}^{\mathbf{k}} \phi_\delta(\mathbf{x}, \mathbf{y}_c). \tag{A1.29}$$

Then, $\mathbf{a}_{\mathbf{k}}$ is related to $b_{\mathbf{k}}$ as follows [33]:

$$\mathbf{a}_{\mathbf{k}}(\mathbf{x}, \mathbf{y}_c) = \frac{1}{\mathbf{k}!} \mathbf{D}_{\mathbf{y}}^{\mathbf{k}} \mathbf{K}_\delta(\mathbf{x}, \mathbf{y}_c) = \frac{1}{\mathbf{k}!} \mathbf{D}_{\mathbf{y}}^{\mathbf{k}} (-\nabla_{\mathbf{y}} \phi_\delta(\mathbf{x}, \mathbf{y}_c)) = -\frac{1}{\mathbf{k}!} D_{\mathbf{y}}^{\mathbf{k}} \left(\sum_{i=1}^{3} \mathbf{e}_i D_{\mathbf{y}}^{\mathbf{e}_i} \phi_\delta(\mathbf{x}, \mathbf{y}_c) \right),$$
$$= -\frac{1}{\mathbf{k}!} \sum_{i=1}^{3} \mathbf{e}_i D_{\mathbf{y}}^{\mathbf{k}+\mathbf{e}_i} \phi_\delta(\mathbf{x}, \mathbf{y}_c) = -\sum_{i=1}^{3} \mathbf{e}_i (k_i + 1) b_{\mathbf{k}+\mathbf{e}_i}(\mathbf{x}, \mathbf{y}_c), \tag{A1.30}$$

where $\mathbf{e}_i$ is the ith Cartesian-basis vector. Therefore, to compute $\mathbf{a}_{\mathbf{k}}$, it is sufficient to obtain $b_{\mathbf{k}}$. The calculation of $b_{\mathbf{k}}$ is performed recursively to reduce the computational load, using the following formula [33]:

$$|\mathbf{k}| R^2 b_{\mathbf{k}} - (2|\mathbf{k}| - \mathbf{1}) \sum_{i=1}^{3} (\mathbf{x} - \mathbf{y}) \cdot \mathbf{e}_i b_{\mathbf{k}-\mathbf{e}_i} + (|\mathbf{k}| - 1) \sum_{i=1}^{3} b_{\mathbf{k}-2\mathbf{e}_i} = 0 \tag{A1.31}$$

for $|\mathbf{k}| \geqslant 1$, where $b_0(\mathbf{x}, \mathbf{y}) = \phi_\delta(\mathbf{x}, \mathbf{y})$, $b_{\mathbf{k}}(\mathbf{x}, \mathbf{y}) = 0$ if any $k_i < 0$, and $R^2 = |\mathbf{x} - \mathbf{y}|^2 + \delta^2$.

To utilize the same machinery, we develop a similar relation for $\mathbf{c}_\mathbf{k}$ here:

$$\mathbf{c}_\mathbf{k}(\mathbf{x},\mathbf{y}_c) = \frac{1}{\mathbf{k}!} D_\mathbf{y}^\mathbf{k}(\nabla_\mathbf{x}\mathbf{K}_\delta(\mathbf{x},\mathbf{y}_c)) = \frac{1}{\mathbf{k}!} D_\mathbf{y}^\mathbf{K}(\nabla_\mathbf{y}\nabla_\mathbf{y}\phi_\delta(\mathbf{x},\mathbf{y}_c)) = \frac{1}{\mathbf{k}!} D_\mathbf{y}^\mathbf{k} \sum_{i=1}^{3} \mathbf{e}_i D_\mathbf{y}^{\mathbf{e}_i} \sum_{j=1}^{3} \mathbf{e}_j D_\mathbf{y}^{\mathbf{e}_j} \phi_\delta(\mathbf{x},\mathbf{y}_c)$$
$$= \frac{1}{\mathbf{k}!} \sum_{i=1}^{3}\sum_{j=1}^{3} \mathbf{e}_i\mathbf{e}_j D_\mathbf{y}^{\mathbf{k}+\mathbf{e}_i+\mathbf{e}_j} \phi_\delta(\mathbf{x},\mathbf{y}_c) = \sum_{i=1}^{3}\sum_{j=1}^{3} \mathbf{e}_i\mathbf{e}_j \frac{(\mathbf{k}+\mathbf{e}_i+\mathbf{e}_j)!}{\mathbf{k}!} b_{\mathbf{k}+\mathbf{e}_i+\mathbf{e}_j}(\mathbf{x},\mathbf{y}_c) \tag{A1.32}$$

with

$$\frac{(\mathbf{k}+\mathbf{e}_i+\mathbf{e}_j)!}{\mathbf{k}!} = \begin{cases} (k_i+1)(k_i+2), & \text{for } i=j, \\ (k_i+1)(k_j+1), & \text{otherwise.} \end{cases} \tag{A1.33}$$

Therefore, just as $\mathbf{a}_\mathbf{k}$, $\mathbf{c}_\mathbf{k}$ can be obtained from b_k. The only difference is that we need to evaluate $b_\mathbf{k}$ up to one order higher than that required for $\mathbf{a}_\mathbf{k}$. The additional cost of one more order does not matter much in most calculations, since most of particle–cluster interactions are dealt with at relatively low orders, namely, $p \leqslant 6$. As clearly seen in (A1.32), $\mathbf{c}_\mathbf{k}$ is symmetric in its indices, i and j, and we thus evaluate only six components instead of all the nine components separately.

A similar construction has been developed for the particle–cluster interactions in (A1.17)–(A1.19), that is

$$\sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x},\boldsymbol{\chi}_j)E_j \approx \sum_{|\mathbf{k}|\leqslant p} \mathbf{a}_\mathbf{k}(\mathbf{x},\mathbf{y}_c) m_\mathbf{k}^\varepsilon(c), \tag{A1.34}$$

$$\sum_{j=1}^{N_c} \nabla_\mathbf{x}\mathbf{K}_\delta(\mathbf{x},\boldsymbol{\chi}_j)E_j \approx \sum_{|\mathbf{k}|\leqslant p} \mathbf{c}_\mathbf{k}(\mathbf{x},\mathbf{y}_c) m_\mathbf{k}^\varepsilon(c), \tag{A1.35}$$

where

$$m_\mathbf{k}^\varepsilon(c) = \sum_{j=1}^{N_c} (\boldsymbol{\chi}_j - \mathbf{y}_c)^\mathbf{k} E_j, \tag{A1.36}$$

and

$$\sum_{j=1}^{N_c} \mathbf{K}_\delta(\mathbf{x},\boldsymbol{\chi}_j)\cdot\mathbf{G}_j \approx \sum_{|\mathbf{k}|\leqslant p} \mathbf{a}_\mathbf{k}(\mathbf{x},\mathbf{y}_c)\cdot\mathbf{m}_\mathbf{k}^h(c), \tag{A1.37}$$

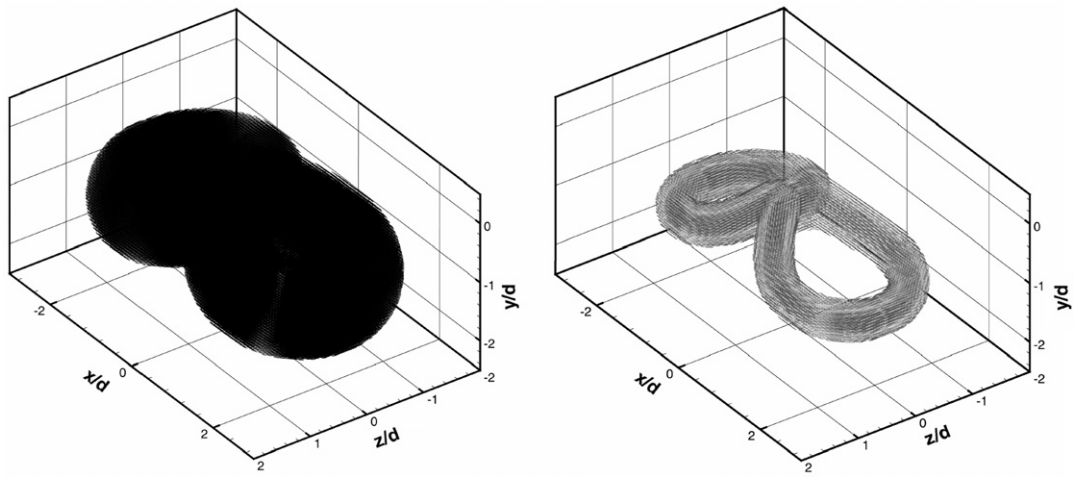

Fig. A.1. View of vortex element distribution at $t = 4.8$ s. Only element with $|\omega_i\,\mathrm{d}V_i| \leqslant 10^{-4}$ are plotted left, and all the elements are plotted right.

where

$$\mathbf{m}_{\mathbf{k}}^{h}(c) = \sum_{j=1}^{N_c} (\boldsymbol{\chi}_j - \mathbf{y}_c)^{\mathbf{k}} \mathbf{G}_j, \tag{A1.38}$$

Since all of these relations only require the evaluation of the same coefficients, namely, $\mathbf{a}_{\mathbf{k}}$ and $\mathbf{c}_{\mathbf{k}}$, the process can be efficiently integrated in a single treecode.

A.4. Numerical example

Tests are performed to check the algorithm's accuracy and efficiency. The velocity gradient field of colliding vortex rings is computed with the multi-purpose treecode fast summation routine, and the result is compared to the result of direct summation. Two separate rings are placed with an angle in the unbounded three-dimensional space (Fig. A.1). The number of vortex elements is 163,521. The velocity gradient is computed on a Cartesian grid with $x/d \in \{-2.5; 2.5\}$, $y/d \in \{-2.5; 0.5\}$, and $z/d \in \{-2; 2\}$, where d is the initial radius of each ring in the simulation. The grid contains 20,580 points. For test purposes, the algorithm is implemented in a double precision serial code on a Pentium4 workstation.

Typically, the order of the Taylor approximation used in the treecode shows dependance on the treecode tolerance parameter based on the potential [33]. Three different tolerance parameters are tried: $\varepsilon = 10^{-1}$, 10^{-2}, and 10^{-3}. The norm of the gradient matrix $\mathbf{A} = [a_{ij}]$ is defined as $\|\mathbf{A}\| = \sqrt{\sum_{i,j=1}^{3} a_{ij}^2}$. The relative error is the maximum value over all grid points of the relative error. We also computed the maximum value over all points of the gradient trace to check incompressibility. For three values of the accuracy parameter, $\varepsilon = 10^{-1}$, 10^{-2}, 10^{-3}, the trace of the velocity gradient matrix is about 10^{-16}, which suggests the treecode indeed compute the velocity gradient accurately. As observed in Table A.1, with the accuracy parameter $\varepsilon = 10^{-3}$, the relative

Table A.1
CPU time and error in the fast evaluation of velocity gradients as a function of the accuracy parameter (potential criterion) $\varepsilon = 10^{-1}$, 10^{-2}, 10^{-3}

Tolerance parameter ε	Direct summation	10^{-1}	10^{-2}	10^{-3}
CPU time (s)	755.89	20.17	25.96	41.87
Max relative error		3.65E−01	3.04E−02	2.50E−03
Max trace value		7.21E−16	6.66E−16	5.55E−16

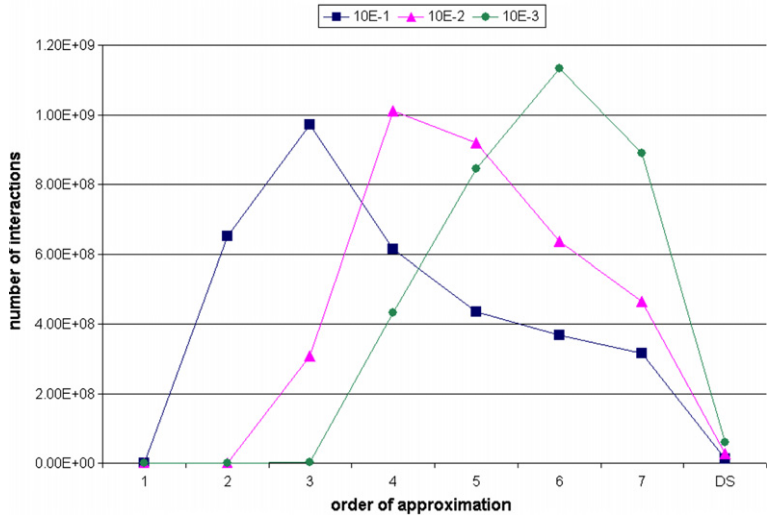

Fig. A.2. Number of interactions with targets points in the treecode as a function of the order of expansions in Taylor approximation.

error is about 0.25% and the CPU time is only 5.5% of the direct summation time. The ratio of computational time is around 18–1. As expected, the error decreases linearly as the tolerance parameter is reduced.

Fig. A.2 shows the number of interactions at each order of approximation. The number of interactions is the number of cells multiplied by the average of particles per cell at each order. For $\varepsilon = 10^{-3}$, most interactions are evaluated at orders of $p = \{5, 6, 7\}$. The simulations presented in the main section of the paper were all performed with $\varepsilon = 10^{-3}$. Our numerical experiments show that the velocity field itself is about an order-of-magnitude more accurate than the gradient field. The evaluation of the gradients requires one or two more coefficients in Taylor expansions, which may slightly reduce the speed-up. However, since the coefficients are only computed once and used for both the calculations of the velocity and its gradients, the overall additional computational cost is minimized.

References

[1] A.J. Majda, A.L. Bertozzi, Vorticity and Incompressible Flow, Cambridge University Press, Cambridge, MA, 2002.
[2] G.H. Cottet, P.D. Koumoutsakos, Vortex Methods: Theory and Practice, Cambridge University Press, London, 2000.
[3] A.F. Ghoniem, A.K. Oppenheim, Numerical-solution for the problem of flame propagation by the random element method, AIAA Journal 22 (10) (1984) 1429–1435.
[4] A.F. Ghoniem, F.S. Sherman, Grid-free simulation of diffusion using random-walk methods, Journal of Computational Physics 61 (1) (1985) 1–37.
[5] C.R. Anderson, A vortex method for flows with slight density variations, Journal of Computational Physics 61 (1985) 417.
[6] A.F. Ghoniem, G. Heidarinejad, A. Krishnan, Numerical-simulation of a thermally stratified shear-layer using the vortex element method, Journal of Computational Physics 79 (1) (1988) 135–166.
[7] A. Krishnan, A.F. Ghoniem, Simulation of the rollup and mixing in Rayleigh Taylor flow using the vortex/transport element method, Journal of Computational Physics 99 (1992) 1–27.
[8] M.C. Soteriou, A.F. Ghoniem, On the application of the infinite reaction rate model in the simulation of the dynamics of exothermic mixing layers, Combustion Science and Technology 105 (4–6) (1995) 377–397.
[9] M.C. Soteriou, A.F. Ghoniem, On the effects of the inlet boundary condition on the mixing and burning in reacting shear flows, Combustion and Flame 112 (3) (1998) 404–417.
[10] M.C. Soteriou, Y. Dong, B.M. Cetegen, Lagrangian simulation of the unsteady near field dynamics of planar buoyant plumes, Physics of Fluids 14 (9) (2002) 3118–3140.
[11] O.M. Knio, A.F. Ghoniem, 3-Dimensional vortex simulation of rollup and entrainment in a shear-layer, Journal of Computational Physics 97 (1) (1991) 172–223.
[12] O.M. Knio, A.F. Ghoniem, The 3-dimensional structure of periodic vorticity layers under nonsymmetrical conditions, Journal of Fluid Mechanics 243 (1992) 353–392.
[13] D. Wee, A.F. Ghoniem, Modified interpolation kernels for treating diffusion and remeshing in vortex methods, Journal of Computational Physics 213 (2006) 263–293.
[14] S. Stanaway, B. Cantwell, P. Spalart, A numerical study of viscous vortex rings using a spectral method, TM 101004, NASA, 1988 SK Stanaway.
[15] Y.M. Marzouk, D. Wee, A.F. Ghoniem, Simulations of high Reynolds number transverse jets and analysis of the underlying vortical structures, in: 43rd AIAA Aerospace Sciences Meeting; Reno, NV, 10–14 January, 2005.
[16] G.S. Wickelmans, A. Leonard, Contributions to vortex particles methods for 3-dimensional incompressible unsteady flows, Journal of Computational Physics 109 (1993) 247.
[17] R.S. Scorer, Experiments on convection of isolated masses of buoyant fluid, Journal of Fluid Mechanics 2 (1957) 583.
[18] R.S. Scorer, Natural Aerodynamics, Pergamon Press, London, 1958.
[19] C.P. Wang, Motion of an isolated buoyant thermal, Physics of Fluids 14 (3) (1971) 1643.
[20] S. Lin, L. Tsang, C.P. Wang, Temperature field structure in strongly heated buoyant thermals, Physics of Fluids 15 (12) (1972) 2118.
[21] J.S. Turner, Buoyant vortex rings, Proceedings of Royal Society of London, Series A 239 (1957) 61.
[22] J.S. Turner, Buoyancy Effects in Fluids, Cambridge University Press, Cambridge, 1973.
[23] M.P. Escudier, T. Maxworthy, On the motion of turbulent thermals, Journal of Fluid Mechanics 61 (1973) 541–552.
[24] D.L. Marcus, J.B. Bell, The structure and evolution of the vorticity and temperature fields in thermals, Theoretical and Computational Fluid Dynamics 3 (1992) 327–344.
[25] G.R. Baker, D.I. Meiron, S.A. Orszag, Vortex simulations of the Rayleigh–Taylor instability, Physics of Fluids 23 (8) (1980) 1485.
[26] R.M. Kerr, Simulation of Rayleigh–Taylor flows using vortex blobs, Journal of Computational Physics 76 (1988) 48.
[27] G. Tryggvason, Numerical simulations of the Rayleigh–Taylor instability, Journal of Computational Physics 75 (1988) 253.
[28] J.A. Zufiria, Vortex-in-cell simulation of bubble competition in Rayleigh–Taylor instability, Physics of Fluids 31 (11) (1988) 3199.
[29] J.A. Zufiria, Linear analysis of the vortex-in-cell algorithm applied to Rayleigh–Taylor instability, Journal of Computational Physics 80 (1989) 387.
[30] S.G. Brecht, J.R. Ferrante, Vortex-in-cell calculations in three dimensions, Computer Physics and Communication 58 (1990) 25.
[31] S.H. Brecht, J.R. Ferrante, Vortex-in-cell simulation of buoyant bubbles in three dimensions, Physics of Fluids A 1 (7) (1989) 1166.

[32] J.H. Walther, P. Koumoutsakos, Three-dimensional vortex methods for particle-laden flows with two-way coupling, Journal of Computational Physics 167 (2001) 39–71.
[33] K. Lindsay, R. Krasny, A particle method and adaptive treecode for vortex sheet motion in three-dimensional flow, Journal of Computational Physics 172 (2001) 879–907.
[34] Y.M. Marzouk, A.F. Ghoniem, K-means clustering for optimal partitioning and dynamic load balancing of parallel hierarchical N-body simulations, Journal of Computational Physics 207 (2005) 493–528.