JOURNAL OF COMPUTATIONAL PHYSICS 113, 177–200 (1994)

A Fast Adaptive Vortex Method in Three Dimensions*

ANN S. ALMGREN

Lawrence Livermore National Laboratory, Livermore, California 94550

THOMAS BUTTKE

Courant Institute of Mathematical Sciences, New York University, New York, New York 10003

AND

PHILLIP COLELLA

University of California, Berkeley, Berkeley, California 94720

Received March 25, 1992; revised December 1, 1993

The method of local corrections (MLC) developed by Anderson for two spatial dimensions is a particle–particle particle–mesh method, in which the calculation of the velocity field induced by a collection of vortices is split into two parts: (i) a finite difference velocity field calculation using a fast Poisson solver, the results of which are used to represent the velocity field induced by vortices far from the evaluation point; and (ii) an N-body calculation to compute the velocity field at a vortex induced by nearby vortices. We present a fast vortex method for incompressible flow in three dimensions, based on the extension of the MLC algorithm from two to three spatial dimensions and the use of adaptive mesh refinement in the finite difference calculation of the MLC. Calculations with a vortex ring in three dimensions show that the break-even point between the MLC with AMR and the direct method is at $N \approx 3000$ on a Cray Y-MP; for $N \approx 64{,}000$ MLC with AMR can be 12 times faster than the direct method. Results from calculations of two colliding inviscid vortex rings demonstrate the increased resolution which can be obtained using fast methods.

INTRODUCTION

Vortex methods are used to approximate time-dependent incompressible flows. They are particle methods based on the Lagrangian formulation of the flow equations, in which vorticity is the quantity carried by the particles. The configuration of vortex elements at a given time determines the velocity field via an N-body Biot–Savart calculation, which is then used to update the positions of the vortices. In three dimensions, the vorticity itself must be updated as well. Vortex methods are especially useful for flows which are dominated by localized vorticity distributions, e.g., shear flows, wakes, and jets. In these flows most of the vorticity is confined to a relatively small portion of the flow, and then a method based on following the vorticity can be very economical.

Point vortex methods were first introduced by Rosenhead in 1931 [25]. A general stable vortex method suitable for high Reynolds number and inviscid flow calculations in two and three dimensions was developed by Chorin [14, 15]; see also Leonard [23]. Convergence of these methods has been established [3, 8, 9, 18, 20].

The usefulness of these methods has been seriously limited in the past by their cost. The accuracy of the methods and their ability to resolve small scales increase with the number of particles, N, as does the time and expense. The cost of the N-body calculation is $O(N^2)$, making it prohibitively expensive for relatively few vortices (on the order of thousands). Fast vortex methods have been developed to try to maintain the accuracy and adaptivity of the standard vortex method while increasing the speed. These fast methods approximate the $O(N^2)$ velocity calculation with a fast calculation whose cost is $O(N \log N)$ for large N.

One of the earliest fast techniques used to approximate a particle method is known as cloud-in-cell. In this method,

* Research supported at UC Berkeley by the Army Research Office under Grant DAALO3-88-K-0197; by ARPA and the National Science Foundation under Grant DMS-8919074; and by a National Science Foundation Presidential Young Investigator award under Grant ACS-8958522. Research at NYU supported by the Department of Energy under Grant DE-FG02-88ER25053. Research at the Institute for Advanced Study supported by the National Science Foundation under Grant DMS-9100383. Computational support for the calculations presented was provided by the San Diego Supercomputer Center (DMS-91002S), the National Center for Supercomputing Applications (DMS-910022N), the Pittsburgh Supercomputing Center (DMS-920003P), and the Center for Computational Sciences at the NASA Goddard Space Flight Center.

0021-9991/94 $6.00

the N-body calculation is replaced by a finite difference Poisson solve, with no distinction made between the fields due to nearby versus distant particles. A second type of fast method is known as PPPM, for particle–particle particle–mesh. This incorporates the idea of separating the calculation into a far-field component, which can be calculated on a grid, and a near-field component, which must be calculated directly. In the PPPM method these two components result from distinct forces. See Hockney and Eastwood [21] for a discussion of these two methods and their limitations. A different type of fast method is based on the hierarchical structure known as a treecode; see Barnes and Hut [7]. This method groups particles spatially before computing approximte interactions. A more systematic approach to this hierarchical ordering is the fast multipole method developed by Greengard and Rokhlin [13, 19], which approximates the velocity field using a multipole expansion of the stream function (in two dimensions). Anderson [2] developed an "implementation of the multipole method without multipoles," based on the same principles as the fast adaptive multipole method, but using a representation by Poisson integrals rather than multipoles. Van Dommelen and Rundensteiner [27] have presented a method also similar to the adaptive multipole method, but with Laurent series rather than Taylor series, and using a different algorithm for the sorting and collecting of vortices.

We present here an extension of a different fast vortex method known as the method of local corrections (MLC). In the MLC, developed by Anderson [1] in two dimensions and here extended to three dimensions, a uniform grid is introduced on the computational domain enclosing the vortices, and the velocity field is calculated on that grid. A corrected form of this velocity is then interpolated onto the vortices, and local interactions are calculated directly. This is similar to previous PPPM algorithms, but it differs in one important respect. The MLC algorithm more accurately separates the local N-body effects from the far-field solution as represented on the grid; in particular, the interpolation from the grid is performed on values that are discretely harmonic.

While the MLC is faster than the standard vortex methods for N in the thousands, there is further efficiency to be gained by using adaptive mesh refinement (AMR) [10, 11, 24] on the grid. The accuracy of the MLC has very weak dependence on the mesh spacing of the computational grid used to calculate the far-field velocity, as long as the mesh spacing is sufficiently larger than the intervortex spacing and small enough that the method does not reduce to the $O(N^2)$ method for most vortices [5, 6]. Thus, within these limits, the mesh spacing can be chosen solely on the basis of timing considerations. With the use of adaptive mesh refinement, the mesh is refined where the vortices are most concentrated, thereby reducing the time spent in local interactions, while increasing as little as possible the cost of solving the Poisson equation. This adaptivity is even more important in three dimensions than in two, because vorticity is often limited to a smaller fraction of the domain in three dimensions. The improvement in timing by using AMR with the MLC can be substantial; for example, for a vortex ring in three dimensions with $N \approx 64{,}000$, the speedup of the MLC with AMR over the uniform MLC is approximately three; the speedup of MLC with AMR over the standard vortex method is approximately 12.

THREE DIMENSIONAL VORTEX METHOD

Vortex methods are based on the vorticity-stream function formulation of the Euler equations for incompressible, inviscid fluid flow [16],

$$\frac{D\boldsymbol{\omega}}{Dt} = \frac{\partial \boldsymbol{\omega}}{\partial t} + (\mathbf{u}\cdot\nabla)\boldsymbol{\omega} = (\boldsymbol{\omega}\cdot\nabla)\mathbf{u},$$

$$\boldsymbol{\omega} = \nabla \times \mathbf{u},$$

where $\boldsymbol{\omega}$ is the vorticity, $\mathbf{u}$ is the velocity. The velocity $\mathbf{u}$ is known from the stream function Ψ, which in turn can be found from the vorticity:

$$\mathbf{u} = \nabla \times \Psi, \qquad \Psi = -(\Delta)^{-1}\,\boldsymbol{\omega}.$$

Following Chorin [15], we discretize the original vorticity field into N nonsingular computational elements,

$$\begin{aligned}\boldsymbol{\omega}(\mathbf{x}, t) &= \sum_{i=1}^{N} \boldsymbol{\omega}_i(t) f_\delta(|\mathbf{x} - \mathbf{x}_i^C(t)|) \\ &= \sum_{i=1}^{N} \Gamma_i(t)\, \mathbf{l}_i(t) f_\delta(|\mathbf{x} - \mathbf{x}_i^C(t)|),\end{aligned}$$

where Γ_i, $\mathbf{l}_i$, $\boldsymbol{\omega}_i = \Gamma_i \mathbf{l}_i$ and $\mathbf{x}_i^C$ are the circulation, vector length, strength, and location of the center, respectively, of the ith vortex segment, and $f_\delta(r)$ is the core function with core radius δ:

$$f_\delta(r) = \begin{cases} \dfrac{3}{4\pi\delta^3} & \text{for } r < \delta \\ 0 & \text{for } r \geq \delta. \end{cases}$$

We require that the integral of the core function over the region of its support be unity; this accounts for the $3/4\pi$ seen in the expression. This core function has been shown to give second-order convergence of the standard vortex method [4].

To initialize the method, for nonperiodic problems we first choose a finite number of closed vortex curves within the flow to approximate the support of the vorticity at $t = 0$. Each curve is then approximated by a N_{seg}-sided polygon,

where N_{seg} is the number of segments per filament. Each side of the polygon now represents one segment in the filament. The endpoints of the vortex segments are the vertices of the polygon, and the centers are defined as midpoints of each pair of adjoining vertices. This is similar to placing the center of a vortex segment on a vortex curve and aligning the segment with the curve at that point, but this latter method would have to be followed with an algorithm to connect adjoining segments. The initialization we use guarantees connected segments. In summary, the relations satisfied by the top, bottom, center, and length of the ith segment ($\mathbf{x}_i^T$, $\mathbf{x}_i^C$, and $\mathbf{x}_i^B$, and $\mathbf{l}_i$, respectively) are

$$\mathbf{x}_i^T(0) - \mathbf{x}_i^B(0) = \mathbf{l}_i(0),$$
$$\tfrac{1}{2}(\mathbf{x}_i^T(0) + \mathbf{x}_i^B(0)) = \mathbf{x}_i^C(0).$$

We advect the positions of the vortex endpoints according to

$$\frac{d\mathbf{x}_i^{T/B}}{dt}(t) = \mathbf{u}(\mathbf{x}_i^{T/B}(t))$$

using the second-order time-stepping procedure

$$\mathbf{x}_i^* = \mathbf{x}_i^n + \mathbf{u}(\mathbf{x}_i^n)\,\Delta t$$
$$\mathbf{x}_i^{n+1} = \mathbf{x}_i^n + 0.5(\mathbf{u}(\mathbf{x}_i^n)) + \mathbf{u}(\mathbf{x}_i^*)\,\Delta t,$$

where $\mathbf{x}_i^n \approx \mathbf{x}_i(n\,\Delta t)$, $\mathbf{x}_i^{n+1} \approx \mathbf{x}_i((n+1)\,\Delta t)$. Once the locations of the top and bottom of each segment are updated, the center points must be redefined as well:

$$\mathbf{x}_i^{C,n+1} = \tfrac{1}{2}(\mathbf{x}_i^{T,n+1} + \mathbf{x}_i^{B,n+1}).$$

The velocity field induced by all the vortex elements is a linear superposition of the velocity fields due to each vortex element, which are found using the Biot–Savart law. We can invert the expression $\boldsymbol{\omega} = \nabla \times \mathbf{u}$ using the stream function Ψ, to find $\mathbf{u}$ from $\boldsymbol{\omega}$. The infinite domain Green's function for the three-dimensional Laplacian is

$$G(\mathbf{x}) = -\frac{1}{4\pi r},$$

where $r = \sqrt{x^2 + y^2 + z^2}$. The kernel $\mathbf{K}(\mathbf{x}) = \nabla \times G$ satisfying $\nabla \times (G * \boldsymbol{\omega})$ is

$$\frac{1}{4\pi r^3}\begin{bmatrix} 0 & z & -y \\ -z & 0 & x \\ y & -x & 0 \end{bmatrix},$$

and the velocity $\mathbf{u}$ is given by $\mathbf{u} = \mathbf{K} * \boldsymbol{\omega}$. Substituting the discretization for $\boldsymbol{\omega}$, we find

$$\mathbf{u}(\mathbf{x}, t) = \sum_{j=1}^{N} \Gamma_j(t)(\mathbf{K}(\mathbf{x} - \mathbf{x}_j^C(t)) * f_\delta)\,\mathbf{l}_j(t)$$
$$= \sum_{j=1}^{N} \Gamma_j(t)(\mathbf{K}_\delta(\mathbf{x} - \mathbf{x}_j^C(t))\,\mathbf{l}_j(t),$$

where we now define the desingularized kernel $\mathbf{K}_\delta = \mathbf{K} * f_\delta$.

Substituting the core function and the discrete desingularized kernel explicitly, we see that the velocity field at $\mathbf{x}$ due to a single vortex segment with center at $\mathbf{x}_0$, circulation Γ, and length $\mathbf{l} = (l_x, l_y, l_z)$ is

$$u = \frac{\Gamma}{4\pi}(l_y(z - z_0) - l_z(y - y_0))\,h(r),$$

$$v = \frac{\Gamma}{4\pi}(l_z(x - x_0) - l_x(z - z_0))\,h(r),$$

$$w = \frac{\Gamma}{4\pi}(l_x(y - y_0) - l_y(x - x_0))\,h(r),$$

where

$$h(r) = \begin{cases} \dfrac{1}{r^3} & \text{for } r > \delta \\ \left(4 - 3\dfrac{r}{\delta}\right)\Big/\delta^3 & \text{for } r \leqslant \delta. \end{cases}$$

The circulation Γ_i of each segment is initially determined as the integral of vorticity across the cross-sectional area represented by the ith segment. The strength of each segment is then defined as its circulation times its length, $\boldsymbol{\omega}_i = \Gamma_i \mathbf{l}_i$. These segments are pieces of vortex lines in the flow (vortex lines are simply defined as curves tangent to the vorticity). By the Kelvin circulation theorem we know that circulation around vortex lines is constant in time, and so the circulation of these computational elements can be held constant in time, i.e.,

$$\frac{D\Gamma_i(t)}{Dt} = 0.$$

However, the lengths $\mathbf{l}_i$ of the segments change as the endpoints move, and so the strength of each vortex evolves as well. Note that the stretching term in the original equation is implicitly incorporated by the relative motion of the endpoints of the segments.

Since the divergence of the curl of a flow field is identically zero, we know that vortex lines cannot end in a flow; they must be closed curves, extend infinitely, or end on a boundary in inviscid flow. For our computations without boundaries (using infinite domain or periodic boundary conditions) we initialize the vorticity into segments connected in closed filaments, with each filament a discrete

approximation to an integral curve of the vorticity. In that case, $\mathbf{x}_{i-1}^{\mathrm{T}} = \mathbf{x}_i^{\mathrm{B}}$, for segments $i-1$, i located on the same filament. Vortex segments, once connected, will remain connected for all time, and for N vortex segments there are only N rather than $2N$ independent endpoints.

One consequence of the variable lengths of the segments is that as the vorticity in a region of flow increases, the lengths of the segments may become disproportionately large. Thus, as part of the algorithm, we check at each time step whether the segment lengths have exceeded a preset critical length. If they have, we divide the long segments in half, giving each new vortex the same circulation as the original vortex. This can result in the number of vortices growing rapidly as the calculation proceeds. The critical length is defined for each segment as twice its initial length. Newly created segments inherit the same critical length as their "parents."

METHOD OF LOCAL CORRECTIONS IN THREE DIMENSIONS

The MLC is a method which reduces the cost of calculating the velocity at the vortices. The goal of the MLC is to replace the full $O(N^2)$ velocity calculation with a fast calculation whose cost varies as $O(N \log N)$ for large N. This is achieved by separating the velocity calculation into several steps: calculation of the far-field velocity on a grid, interpolation of a corrected form of this velocity from the grid onto the vortices, and calculation of local interactions between nearby vortices. The algorithm is sketched below, and the details of the sorting follow.

(a) Find at every grid point $\mathbf{i}$ a field $\mathbf{g}^{\Omega}$ which satisfies

$$\mathbf{g}_{\mathbf{i}}^{\Omega} \approx (\Delta^h \mathbf{u}^{e,h})_{\mathbf{i}}.$$

Here Δ^h is the discrete Laplacian operator with mesh spacing h; $\mathbf{u}_{\mathbf{i}}^{e,h}$ is defined as the exact velocity field induced by the vortices at grid point $\mathbf{i}$, calculated as if these were point vortices.

(b) Solve

$$\Delta^h \tilde{\mathbf{u}} = \mathbf{g}^{\Omega}$$

for the velocity $\tilde{\mathbf{u}}$ on the grid with appropriate boundary conditions.

Note that if $\mathbf{g}^{\Omega}$ were defined exactly as the discrete Laplacian of the velocity due to every vortex at every grid point and if the boundary conditions were specified exactly, then $\tilde{\mathbf{u}}_{\mathbf{i}} = \mathbf{u}_{\mathbf{i}}^{e,h}$ at every grid point to within the specified precision of the Poisson solver. However, this is greater accuracy than is needed (since other errors in the method would swamp this error), and so in the MLC the discrete Laplacian is approximated rather than computed exactly at every point. The contribution of each vortex to $\mathbf{g}^{\Omega}$ is defined as exactly $\Delta^h \mathbf{u}^{e,h}$ *near* the vortex, but it is set to zero at grid points *far* from the vortex. We thereby approximate the value of the discrete Laplacian with the value of the exact Laplacian, which is zero at every point away from the vortex since the velocity due to a point vortex is harmonic. The error of the approximation is just the error of the discrete Laplacian for a harmonic function, which is proportional to the higher derivatives of $\mathbf{u}^{\mathrm{exact}}$. These derivatives fall off rapidly away from the vortex.

(c) Interpolate a corrected form of this grid velocity onto the vortices. The field induced by the nearby vortices will be included in an explicit sum in the last step, so first the effeect of these vortices in the interpolated velocity field is removed. This is done by subtracting the contribution of these vortices to the velocity on the grid prior to interpolation. Then this corrected velocity field is interpolated onto the vortices. Note that since the effect of the nearby vortices is entirely eliminated from the interpolated velocity for $C = D$, this field is discretely harmonic.

(d) Finally, add the velocity due to the nearby vortices to the velocity interpolated from the grid in a direct sum using the desingularized kernel, so as to achieve higher-order accuracy.

In the above algorithm we need a mechanism for distinguishing between "near" and "far" vortices. This is done by sorting all of the vortices into "bins" at the beginning of each time step; this sorting is based on the locations of the vortex segment centers. The centers of the bins are placed at the grid points, and each bin is defined as the box of width h around that center. Then all sorting of near and far vortices and near and far grid points is done using the bin indices.

Let $\Omega = [0, 1]^3$ be the physical domain of the problem. Define a uniform grid G^0 of M^3 points on Ω, with mesh spacing $h = 1/M$. Define $B^{\mathbf{i}}$ as the bin centered on the point $\mathbf{i}h$, and define the $|\cdot|_B$ norm such that $|\mathbf{i} - \mathbf{m}|_B$ is the minimum distance (in units of the mesh spacing) between any point in $B^{\mathbf{i}}$ and any point in $B^{\mathbf{m}}$. Around every grid point now define $R^{\mathbf{i}}$ and $R_0^{\mathbf{i}}$ as

$$R^{\mathbf{i}} = \bigcup_{\mathbf{m}:|\mathbf{m}-\mathbf{i}|_B \leq (D+1)} B^{\mathbf{m}},$$

$$R_0^{\mathbf{i}} = \bigcup_{\mathbf{m}:|\mathbf{m}-\mathbf{i}|_B \leq D} B^{\mathbf{m}},$$

where D is called the spreading distance.

Step (a), the construction of $\mathbf{g}^{\Omega}$, can be broken down into two parts:

(1) For each $\mathbf{i}$ in the interior of Ω:

(i) compute by direct interaction the exact velocity at every grid point $\mathbf{m}$ in $R^{\mathbf{i}}$ due to every vortex n in $B^{\mathbf{i}}$, as if each vortex were a point vortex:

$$\mathbf{u}_{\mathbf{m}}^{e,h} = \sum_{n:\mathbf{x}_n \in B^{\mathbf{i}}} \mathbf{K}(\mathbf{m}h - \mathbf{x}_n)\, \omega_n.$$

(ii) Calculate the Laplacian of this velocity field at every point in R_0^i. Define

$$\mathbf{g}^i = \begin{cases} \Delta^h \mathbf{u}^{e,h} & \text{inside } R_0^i \\ \mathbf{0} & \text{in interior}(\Omega) - R_0^i. \end{cases}$$

Note that $\mathbf{g}^i$ is defined on the interior of the entire domain, but it carries information only about the vortices in B^i.

(2) Superimpose these fields $\mathbf{g}^i$ to form

$$\mathbf{g}^\Omega = \sum_{\mathbf{i} \in \text{interior}(\Omega)} \mathbf{g}^i.$$

Note that the work to represent the velocity field on the entire grid due to all the vortices breaks down as follows: for each vortex, evaluate the exact velocity in a subset of the grid; for each bin of vortices, evalute the Laplacian at an even smaller subset of the grid; for the whole domain (i.e., only once) solve the equation $\Delta^h \tilde{\mathbf{u}} = \mathbf{g}^\Omega$.

For the local corrections step, around each grid point $\mathbf{i}$ define

$$S^i = \bigcup_{m: |m-i|_\beta \leq C} B^m,$$

where C is called the correction radius. If $C = D$ then $S^i = R_0^i$. A vortex p in bin B^i is defined as *near* to a vortex k in bin B^i if B^i is in S^j. Note that p is near k implies that k is near p (this is not necessarily true when AMR is added to the MLC). This part of the calculation ((c) and (d)) is performed one bin at a time. For each $\mathbf{i}$ such that B^i contains vortices:

(1) Define the points to be used in the interpolation stencil $\{\mathbf{X}_m\}$, $m = 1, ..., N_I$. Compute by direct interaction the exact velocity at each point $\mathbf{X}_m$ due to every vortex in S^i, as if each vortex were a point vortex, and subtract this field from the existing velocity at these grid points:

$$\hat{\mathbf{u}}^i(\mathbf{X}_m) = \tilde{\mathbf{u}}(\mathbf{X}_m) - \sum_{n: \mathbf{x}_n \in S^i} \mathbf{K}(\mathbf{X}_m - \mathbf{x}_n)\, \boldsymbol{\omega}_n.$$

Note that $\hat{\mathbf{u}}^i$ and $\hat{\mathbf{u}}^j$, $\mathbf{i} \neq \mathbf{j}$, might both be defined at a grid point $\mathbf{X}$ but would have different values because S^i and S^i contain different vortices.

(2) Interpolate this corrected field $\hat{\mathbf{u}}^i$ from the interpolation points $\mathbf{X}_m$ onto each vortex p in B^i:

$$\mathbf{u}(\mathbf{x}_p) = I(\hat{\mathbf{u}}^i(\mathbf{X}_1), ..., \hat{\mathbf{u}}^i(\mathbf{X}_{N_I}); \mathbf{x}_p).$$

After this interpolation, the velocity of every vortex in B_i is due only to the vortices outside S_i.

(3) In this final step, every *local* interaction is calculated directly, incorporating the higher-order shape functions. Add the velocity due to every vortex n in S_i to the existing velocity of every vortex p in B_i using $\mathbf{K}_\delta$ rather than $\mathbf{K}$:

$$\mathbf{u}(\mathbf{x}_p) := \mathbf{u}(\mathbf{x}_p) + \sum_{n: \mathbf{x}_n \in S^i} \mathbf{K}_\delta(\mathbf{x}_p - \mathbf{x}_n)\, \boldsymbol{\omega}_n.$$

In summary, the velocity at vortex p, located in bin B^i, can be written

$$\mathbf{u}(\mathbf{x}_p) = I(\hat{\mathbf{u}}^i; \mathbf{x}_p) + \sum_{n: \mathbf{x}_n \in S^i} \mathbf{K}_\delta(\mathbf{x}_p - \mathbf{x}_n)\, \boldsymbol{\omega}_n.$$

There are a number of parameters which affect the accuracy and cost of the MLC. We distinguish here between the error inherent in using a vortex method to approximate the solution to the Euler equations and the error which results from approximating the standard vortex method with the MLC.

The first type of error, that of the vortex method itself, depends on the intervortex spacing, the core function, and the time step. The second type of error, that of approximating the standard vortex method with the MLC, can be separated into two parts: (a) the error in representing the velocity on the grid, and (b) the error in interpolating the corrected velocity from the grid onto the vortices. The first error results from approximating the value of the discrete Laplacian of the velocity due to a vortex element by zero away from that element; this error depends on the spreading distance D. As D increases for constant grid spacing h, we make this approximation on fewer points farther away from the vortex element, and thus in the limit $D = M$ this error is machine zero (assuming that the Poisson equation has been solved to this precision). The second error results from interpolation. For constant h, as we increase the correction radius C we are interpolating not only a smaller fraction of the total velocity field (since the velocity we are interpolating is due to fewer vortices), but also a smoother function, since the corrected velocity is due only to vortices outside the correction radius. Thus as C increases the interpolation error goes to zero, and in the limit $C = D = M$ the MLC effectively reduces to the standard vortex method.

Parameter studies in two dimensions by Anderson [1] show that the error in vortex positions calculated at finite time using the MLC with C and D in the range $1 \leq C, D \leq 4$ is comparable to the error of the solution computed using the direct method. The relative error is less than 3% when $C = D = 1$, and less than 0.2% when $C = D = 2$. Parameter studies by the authors show similar results for three-dimensional calculations. Thus for our calculations we choose $C = D = 1.5$, which is borne out by the results of Anderson [1] and Baden [5, 6] and our own studies to give sufficient accuracy. By sufficient accuracy we mean that the errors due to using the MLC to approximate the direct method are significantly smaller than the errors due to the vortex method discretization.

The principal difficulty in extending this algorithm to three dimensions is the construction of a suitably accurate interpolation scheme. In two dimensions, Anderson exploits the fact that the velocity induced by a vortex is a potential flow field away from the support of the vorticity to write the velocity components as the real and imaginary parts of a complex analytic function. He then uses complex polynomial interpolation to construct I, an interpolation function with a highly compact stencil relative to its accuracy. In three dimensions, we seek to mimic Anderson's algorithm by defining an interpolation stencil which is fourth-order accurate, but the velocity induced by a single vortex segment is not a potential flow field away from the support of the vorticity. However, the velocity field is divergence-free, and the Laplacian of each of its components vanishes. We take advantage of these features to construct an accurate interpolation function with a compact stencil. Note that, since the velocity field which is interpolated results only from *far* vortices, the higher derivatives of $\mathbf{u}$ do exist and remain finite as N increases, since for fixed C the ratio of the distance of the nearest vortices to the distance between interpolation points remains constant.

Consider that we want to interpolate a scalar function u onto position (x_0, y_0, z_0) from an interpolation stencil centered at grid point (i, j, k) of a uniform grid with mesh spacing h. Assume that (x_0, y_0, z_0) lies closer to (ih, jh, kh) than to any other grid point. Define $x = x_0 - ih$, $y = y_0 - jh$, $z = z_0 - kh$. We see that $|x| \leqslant h/2$, $|y| \leqslant h/2$, $|z| \leqslant h/2$. Using a Taylor expansion, we can write

$$\begin{aligned} u(x_0, y_0, z_0) = u(ih, jh, kh) &+ xu_x + yu_y + zu_z \\ &+ \tfrac{1}{2}(x^2u_{xx} + y^2u_{yy} + z^2u_{zz}) \\ &+ xyu_{xy} + yzu_{yz} + xzu_{xz} \\ &+ \tfrac{1}{6}(x^3u_{xxx} + y^3u_{yyy} + z^3u_{zzz}) \\ &+ \tfrac{1}{2}(x^2yu_{xxy} + xy^2u_{xyy} + x^2zu_{xxz} \\ &+ xz^2u_{xzz} + y^2zu_{yyz} + yz^2u_{yzz}) \\ &+ xyzu_{xyz} + O(h^4), \end{aligned}$$

where $u_x = \partial u/\partial x$, $u_{xy} = \partial^2 u/\partial x\,\partial y$, $u_{xyz} = \partial^3 u/\partial x\,\partial y\,\partial z$, and so on. All derivatives here are evaluated at (ih, jh, kh).

To create a fourth-order interpolation scheme, we must approximate the first derivatives to $O(h^3)$, the second derivatives to $O(h^2)$, and the third derivatives to $O(h)$, since x, y, and z are of $O(h)$. For example, we define

$$\begin{aligned} s_x^+ &= u_{i+1,j,k+1} + u_{i+1,j,k-1} + u_{i+1,j-1,k} + u_{i+1,j+1,k}, \\ s_x^- &= u_{i-1,j,k+1} + u_{i-1,j,k-1} + u_{i-1,j-1,k} + u_{i-1,j+1,k} \\ f_x &= (s_x^+ - s_x^- + 2(u_{i+1,j,k} - u_{i-1,j,k}))/(12h), \end{aligned}$$

with f_y, f_z defined in a similar fashion. If we Taylor-expand each term in the above expressions for f_x, f_y, f_z about (ih, jh, kh), we see by cancellation of the zeroth and all the first- and second-order terms that these are third-order approximations to the first derivatives u_x, u_y, u_z.

For the second and mixed third derivatives, we use standard finite difference approximations. For example,

$$\begin{aligned} f_{xx} &= (u_{i+1,j,k} + u_{i-1,j,k} - 2u_{i,j,k})/h^2 \\ f_{xy} &= ((u_{i+1,j+1,k} - u_{i-1,j+1,k}) \\ &\quad - (u_{i+1,j-1,k} - u_{i-1,j-1,k}))/(4h^2) \\ f_{xxy} &= ((u_{i+1,j+1,k} - 2u_{i,j+1,k} + u_{i-1,j+1,k}) \\ &\quad - (u_{i+1,j-1,k} - 2u_{i,j-1,k} + u_{i-1,j-1,k}))/(2h^3) \\ f_{xyz} &= (((u_{i+1,j+1,k+1} - u_{i-+1,j+1,k+1}) \\ &\quad - (u_{i+1,j-1,k+1} - u_{i-1,j-1,k+1})) \\ &\quad - ((u_{i+1,j+1,k-1} - u_{i-1,j+1,k-1}) \\ &\quad - (u_{i+1,j-1,k-1} - u_{i-1,j-1,k-1})))/(8h^3). \end{aligned}$$

Instead of approximating f_{xxx} by the first derivative in x of f_{xx}, which would increase the width of the stencil, we use the fact that u is a harmonic function to obtain an expression for it in terms of the mixed third derivatives,

$$u_{xxx} = (-u_{yy} - u_{zz})_x = -u_{xyy} - u_{xzz}.$$

The interpolation scheme, in terms of the terms defined above, can be written

$$\begin{aligned} u(x_0, y_0, z_0) = u(ih, jh, kh) &+ xf_x + yf_y + zf_z \\ &+ \tfrac{1}{2}(x^2f_{xx} + y^2f_{yy} + z^2f_{zz}) \\ &+ xyf_{xy} + yzf_{yz} + xzf_{xz} \\ &+ \tfrac{1}{6}((3x^2 - y^2)\,yf_{xxy} + (3x^2 - z^2)\,zf_{xxz} \\ &+ (3y^2 - x^2)\,xf_{yyx} + (3y^2 - z^2)\,zf_{yyz} \\ &+ (3z^2 - x^2)\,xf_{zzx} + (3z^2 - y^2)\,yf_{zzy}) \\ &+ xyzf_{xyz} + O(h^4). \end{aligned}$$

ADAPTIVE MESH REFINEMENT

The accuracy of the MLC has very weak dependence on the mesh spacing of the grid used to calculate the far-field velocity, as long as the mesh spacing h is sufficiently larger than the intervortex spacing h_v [5, 6] and sufficiently small that the method does not reduce to the $O(N^2)$ calculation for most vortices. Thus, within these limits, we can choose the mesh spacing by timing considerations alone. The goal of AMR with MLC is to reduce the cost of the local corrections by creating smaller bins in regions where the vortices are concentrated, while increasing as little as possible the cost of solving the Poisson equation.

Adaptive mesh refinement introduces a hierarchy of levels and grids. A grid G^l is defined by the set of points, uniformly spaced by h_l in each coordinate direction, which cover a rectangular subset of the domain. The level l of a grid is defined as the $\log_2$ of the ratio of the grid spacing of the level 0 grid, h_0, to the grid spacing h_l. The ratio of h_{l-1} to h_l is called the refinement ratio of level l. Here we consider refinement ratios of two. For the sake of exposition, we will assume in the following discussion that there is only one grid at each level. However, this is *not* assumed in the implementation, although it is the case in computation presented.

Let us define here a hierarchy of grids on levels $l = 0, ..., l_{\max}$, where G^0 is the base grid, as defined in the uniform grid MLC, and $G^{l_{\max}}$ is the grid at the finest level $l_{\max}$. Only the base grid G^0 is defined over the whole domain Ω; all finer grids cover only some subset of Ω. Grids at different levels are aligned so that each grid point in a level l grid (for $l > 0$) with even spatial indices is at the same physical location as a grid point at level $l-1$. Thus the points, or bin centers, are coincident, but note that this implies that the faces of the bins are not.

We now define a *composite grid* as a union of grids at different levels, with each finer grid nested inside the next coarser grid, i.e.,

$$G^l \subset G^{l-1}, \qquad 1 \leqslant l \leqslant l_{\max}.$$

The composite grid $G^{0:l}$ is defined as

$$G^{0:l} = \bigcup_{i=0}^{l} G^i.$$

Each vortex is sorted into a group V^l, where l is the highest level at which there exists a grid such that the vortex is properly contained in G^l; i.e., a vortex in V^l must lie in the interior of G^l and at least $b+1$ bins from the boundary ∂G^l, where $b = \max(C, D)$. This ensures that the MLC can properly represent the right-hand side of the Poisson equation and perform the local corrections correctly on the level l grid for a vortex in V^l.

The velocity due to a vortex in V^l is represented on the composite grid $G^{0:l}$. The MLC is performed once on $G^{0:l}$ for each level $l = 0, ..., l_{\max}$. The right-hand side for the Poisson equation is defined using only vortices in V^l, and boundary conditions due only to vortices in V^l are defined on G^0. The Poisson equation is then solved on $G^{0:l}$ as described below, generating a velocity field $\tilde{\mathbf{u}}^{l:m}$ on each G^m, $0 \leqslant m \leqslant l$, of $G^{0:l}$. In our notation, the velocity $\tilde{\mathbf{u}}^{l:m}$ results from vortices at level l and is defined on a grid at level m. Note that because the Poisson equation is solved separately for each group of vortices V^l, the right-hand side is only nonzero on G^l; elsewhere in the domain it is set to zero. The velocity is then interpolated onto all vortices, with local corrections done only on G^l. This procedure is repeated for all levels, $0 \leqslant l \leqslant l_{\max}$, adding the contributions from each set of vortices V^l until the velocity of every vortex p due to vortices at all levels has been calculated. We can express the full algorithm as

$$\mathbf{u}(\mathbf{x}_p) = \sum_{l=0}^{l_{\max}} \mathbf{u}^l(\mathbf{x}_p),$$

where

$$\mathbf{u}^l(\mathbf{x}_p) = \begin{cases} I^m(\tilde{\mathbf{u}}^{l:m}; \mathbf{x}_p) & \text{if} \quad m < l \\ I^l(\hat{\mathbf{u}}^{l:l}; \mathbf{x}_p) + \displaystyle\sum_{k \in V^l, k \text{ near } p} K\delta(\mathbf{x}_p - \mathbf{x}_k)\, \omega_k & \\ \qquad\qquad \text{if} \quad m = l, & \end{cases}$$

and

$$\hat{\mathbf{u}}^{l:l}(\mathbf{x}_p) = \tilde{\mathbf{u}}^{l:l}(\mathbf{x}_p) - \sum_{k \in V^l, k \text{ near } p} K(\mathbf{x}_p - \mathbf{x}_k)\, \omega_k.$$

The level m at which the interpolation is done is the finest level such that $m \leqslant l$ and $\mathbf{x}$ is in the interior of G^m. The interpolation stencil I^m is composed of points in G^m with spacing h_m. Note that the vortices in V^l correct only the velocity $\tilde{\mathbf{u}}^{l:l}$ on G^l; grid points in G^l are the only points within a distance Ch_l of the vortices in V^l.

We include here error measurements from a calculation of a vortex ring in three dimensions. The relative L_2 error of the velocity field is shown in Table I for $N = 8011$ and $N = 16232$. The first column in this table specifies the level of refinement; "16" refers to a uniform 16^3 grid, "16-32" refers to a 16^3 base grid with one level of refinement above that (so that the finest level has $h = \frac{1}{32}$), and so on. The ring has a radius of 0.1 around the z-axis and cross-sectional

TABLE I

Relative L_2 Norm of the Error in Velocity for Different Levels of Mesh Refinement

	Relative L_2 error	
Level	16232 vortices	8011 vortices
16	7.1e-4	7.8e-4
16-32	6.9e-4	7.7e-4
16-64	7.5e-4	8.2e-4
16-128	7.6e-4	
32	7.1e-4	7.8e-4
32-64	7.5e-4	8.2e-4
32-128	7.6e-4	

Note. Calculations for a three-dimensional vortex ring of radius 0.1 around the z-axis and cross-sectional radius 0.02. Here $\delta = h^{0.75}$.

radius 0.02. Here we see evidence that the accuracy of the MLC, with or without AMR, shows very little dependence on the variation of the mesh spacing.

MULTIGRID WITH AMR

The stencil of the three-dimensional discrete Laplacian is presented below [17]. Consider the cube of 27 points $(\mathbf{i}+\mathbf{s}) \equiv (i+s_1,\ j+s_2,\ k+s_3)$, $|s_i| \leq 1$, immediately surrounding the point $\mathbf{i}$. Define *face points* on the cube such that $|s_1|+|s_2|+|s_3|=1$. Similarly define *edge points* by $|s_1|+|s_2|+|s_3|=2$, and let *corner points* be those satisfying $|s_1|+|s_2|+|s_3|=3$. Then the 27-point Laplacian can be written

$$(\Delta^h u)_{\mathbf{i}} = \frac{1}{h^2}\left(-\frac{128}{30}u_{\mathbf{i}} + \frac{1}{30}\sum_{\text{corner points}} u_{\mathbf{i}+\mathbf{s}} + \frac{1}{10}\sum_{\text{edge points}} u_{\mathbf{i}+\mathbf{s}} + \frac{7}{15}\sum_{\text{face points}} u_{\mathbf{i}+\mathbf{s}}\right).$$

The MLC requires the solution of the Poisson equation, $\Delta^h \tilde{\mathbf{u}} = \mathbf{g}^{\Omega}$ on the grid $G^0 = [0, M]^3$ covering the domain Ω, where M is assumed to be a power of 2. We use multigrid [12] with Gauss–Seidel relaxation, red–black ordering, and V-cycles to solve this equation. Multigrid is a multilevel relaxation method; i.e., it solves the equation $Lu^0 = \rho^0$ by iterating on the equation $v^{m+1} = v^m + \lambda(Lv^m - \rho)$, where λ is the relaxation parameter, m is the relaxation counter, and v^m is an approximation to the solution u^0, which is defined on the base grid G^0. (Do not be confused by the superscripts—u^0 refers to the solution on the base grid G^0, while v^m refers to the mth iteration of v.) The iteration in m continues until v^m is sufficiently close to the exact solution u, as measured by the value of the residual, $R = \rho - Lv^m$. We choose λ so that the term $v_{\mathbf{i}}^m$ does not appear in the right-hand side of the equation for $v_{\mathbf{i}}^{m+1}$; i.e., $\lambda = -1/C_0$, where $C_0 = -128/(30h^2)$ is the coefficient of the $v_{\mathbf{i}}$ term in the definition of $(\Delta^h v)_{\mathbf{i}}$.

Recall the definition of *face* points, *edge* points, and *corner* points from above. We define the coarsening operator, I^C, such that $R^{\text{coarse}} = I^C R^{\text{fine}}$, for each component of $\mathbf{i}$ even, by

$$R^{\text{coarse}}_{\mathbf{i}/2} = \frac{1}{64}\left(8R^{\text{fine}}_{\mathbf{i}} + \sum_{\text{corner points}} R^{\text{fine}} + 2\sum_{\text{edge points}} R^{\text{fine}} + 4\sum_{\text{face points}} R^{\text{fine}}\right).$$

The coarsening operator is used to average the residuals; the velocity is coarsened using a simple projection, P, defined by

$$(Pu^n)_{\mathbf{i}} = u^n_{2\mathbf{i}}.$$

We define the interpolation operator I^F in the interior of a grid as simple trilinear interpolation.

Let $l_{\min} = \log_2 M$. Then in keeping with the notation of the previous section, we define G^l, $0 \geq l \geq l_{\min}$, as the lth level grid, where $l=0$ is still the base level grid. Each grid covers all of Ω. Note that these new grids are coarser than the base grid, and l for these grids is negative. The solution of the equation $\Delta^{h_0} u^0 = \rho^0$ is found as follows. Note that the l in v^l, e^l, and R^l refers to the level, not the relaxation counter:

$R^0 := \rho^0 - \Delta^{h_0} v^0$.
While $(|R^0| < \varepsilon)$
  $R^0 := \rho - \Delta^{h_0} v^0$
  $e^0 := \text{MGRelax}(0, R^0, h_0)$
  $v^0 := v^0 + e^0$
EndWhile
$u^0 := v^0$

PROCEDURE MGRelax(l, R^l, h_l).

$e^l := 0$
$e^l := \text{Relax}(R^l, e^l, h_l)$
If $(l > l_{\min})$ then
  $h_{l-1} := 2h_l$
  $R^{l-1} := I^C(R^l - \Delta^{h_l} e^l)$.
  $e^{l-1} := \text{MGRelax}(l-1, R^{l-1}, h_{l-1})$
  $e^l := e^l + I^F e^{l-1}$.
  $e^l := \text{Relax}(R^l, e^l, h_l)$
Endif
Return e^l

Here, and in what follows, Relax(ρ^l, u^l, h_l) is a procedure that performs a point relaxation for the operator Δ^{h_l}, given a right-hand side ρ^l and an initial guess u^l, and returns the relaxed solution. The particular scheme we use here is Gauss–Seidel relaxation with red–black ordering.

When AMR is added to the MLC, we need to solve the Poisson equation on each composite grid $G^{0:l}$ for $0 \leq l \leq l_{\max}$. To solve this equation on $G^{0:\hat{l}}$, we must satisfy

$$\Delta^{h_{\hat{l}}} u^{\hat{l}} = \rho^{\hat{l}} \qquad \text{in interior}(G^{\hat{l}}),$$

and for $0 \leq l \leq \hat{l}$,

$$\Delta^{h_l} u^l = 0 \qquad \text{in interior}(G^l) - \text{interior}(G^{l+1})$$
$$u^l = Pu^{l+1} \qquad \text{in } G^{l+1},$$
$$u^{l+1} = I^{\partial} u^l \qquad \text{on } \partial G^{l+1}.$$

Note that we do not attempt to satisfy any equation with Δ^{h_l} in the interior of G^{l+1}; only the finest level equation possible is satisfied at any given point in the domain. The equation containing Δ^{h_l} is satisfied on the boundary of G^{l+1}, $l \geq 0$, however, and this gives the appropriate matching condition.

Since the G^l do not cover the entire domain G^0, we need to interpolate values from G^{l-1} onto the boundaries of the G^l, $l>0$, during each multigrid V-cycle. We use a fourth-order interpolation function, I^∂, on these boundaries, since the boundary values are not smoothed in iterations at the finer levels.

A single V-cycle of the modified multigrid procedure is presented below, starting from the finest level, $l=\hat{l}$, of $G^{0:\hat{l}}$. This procedure is initialized by setting $v^l, \rho^l = 0$ for $0 \leqslant l \leqslant \hat{l}$, except that v^0 is set equal to the Dirichlet boundary conditions on ∂G^0, and $\rho^{\hat{l}}$ is the right-hand side induced by the vortices on the finest grid.

We note here that the multigrid procedure with AMR is very similar to that starting with a uniform grid (which is why multigrid is an appropriate choice for solving the Poisson equation on adaptive composite grids), with the exception that the solution v^l is updated on the boundaries for $l>0$, and the correction e^l is now nonzero on ∂G^l. The corrections themselves carry the boundary conditions for relaxation at the finer levels. This is in contrast to the levels $l<0$ in multigrid, for which the boundary conditions for the relaxation on the residual equation are homogeneous Dirichlet when the imposed boundary conditions are Dirichlet.

$R^{\hat{l}} := \rho^{\hat{l}} - \Delta^{h_{\hat{l}}} v^{\hat{l}}$
While $(|R^{\hat{l}}| < \varepsilon)$
  $l := \hat{l}$
  $R^l := \rho^l - \Delta^{h_l} v^l$
  While $(l>0)$
    $e^l := 0$
    $e^l := \text{Relax}(R^l, e^l, h_l)$
    $v^{l-1} := P(v^l + e^l)$
    $R^{l-1} := \begin{cases} I^C(R^l - \Delta^{h_l} e^l) & \text{in interior}(G^l) \\ -\Delta^{h_{l-1}} v^{l-1} & \text{elsewhere} \end{cases}$
    $l := l-1$
  EndWhile
  $e^0 := \text{MGRelax}(0, R^0, h_0)$
  $v^0 := v^0 + e^0$
  $l := 1$
  While $(l < \hat{l})$
    $e^l := e^l + I^F e^{l-1}$ in interior(G^l)
    $e^l := e^l + I^\partial e^{l-1}$ on ∂G^l
    $e^l := \text{Relax}(R^l, e^l, h_l)$
    $v^l := v^l + e^l$
    $l := l+1$
  EndWhile
EndWhile

TIMING RESULTS

In Table II we present the timings for a single velocity evaluation using the direct (N^2) method, the MLC with a uniform grid, and the MLC with AMR. The timings for MLC and MLC with AMR used the optimal grid or grid hierarchy. These calculations were done for a single vortex ring with N vortices on a Cray Y-MP using the cft77 compiler. The optimal uniform grid for MLC in the table ranges from 8^3 for $N \leqslant 4000$ to 64^3 for $N \approx 64000$. The optimal grid refinement for MLC with AMR ranges from $l_{\max}=0$ for $N \leqslant 4000$ to $l_{\max}=4$ for $N \approx 64000$, with a base grid of 8^3. In the MLC and MLC with AMR calculations, the correction radius and spreading distance are $C=D=1.5$.

Table III shows a breakdown of the CPU time for a single velocity evaluation into the different stages of the calculation. The initial data is a pair of vortex rings as described in the next section, with $N=39060$. The base grid of each computation with AMR is 8^3; $8-32$ indicates a base grid of 8^3 with two levels of refinement, $8-64$ is a base grid of 8^3 with three levels of refinement, and so on. The uniform grid cited is 32^3. The total timings are different than those in Table II because of the different number of vortices and different initial data.

TABLE II

Timing Comparison for a Single Velocity Evaluation Using the Direct (N^2), MLC and MLC with AMR Methods

	Time (CPU seconds)		
N	Direct	MLC	MLC with AMR
1023	0.34	0.69	0.69
2016	1.48	1.83	1.83
3940	4.92	4.61	4.39
8766	24.2	15.8	11.1
17641	98.0	49.5	24.5
31988	322.6	105.7	49.3
63759	1281.0	314.5	104.3

Note. Calculations done for a vortex ring with N vortices on a Cray Y-MP with the cft77 compiler.

TABLE III

Time per Operation for One Velocity Evaluation Using MLC, with Three Different Levels of AMR and for a Uniform Grid Case

	Time (CPU seconds)			
Operation	8-32	8-64	8-128	32
Calculation of $g^{Q^l \max}$	2.9	3.3	5.3	2.9
Direct calculation of boundary conditions	6.6	6.6	6.6	21.6
Solution of Poisson equation	1.0	2.4	7.8	1.8
Correction and interpolation of velocity field	10.8	16.6	25.0	10.8
Direct local interactions	51.3	22.9	10.1	51.3
Total time for full velocity evaluation	72.6	51.8	54.8	88.4

Note. Timings are on a Cray Y-MP with the cft77 compiler.

The issue of where and by how many levels to refine the mesh in MLC with AMR is an important one, because it is the correct choice of refinement which allows the significant savings in time of calculation. Ideally, we would like to have a very simple criterion for refinement, e.g., refine whenever the number of vortices per bin is above some $N_{\max}$, where $N_{\max}$ is independent of level or the configuration of the vortices. However, this is unrealistic, since choosing to refine any one bin may or may not lead to a larger grid at that level, and the fact that we run the calculation on a vector machine means that the time of the calculation is not a simple linear function of the operation count. It might be possible to implement a refinement strategy similar to that presented in [2], but at present the refinement strategy is trial and error.

For the various MLC calculations, we required a suitable

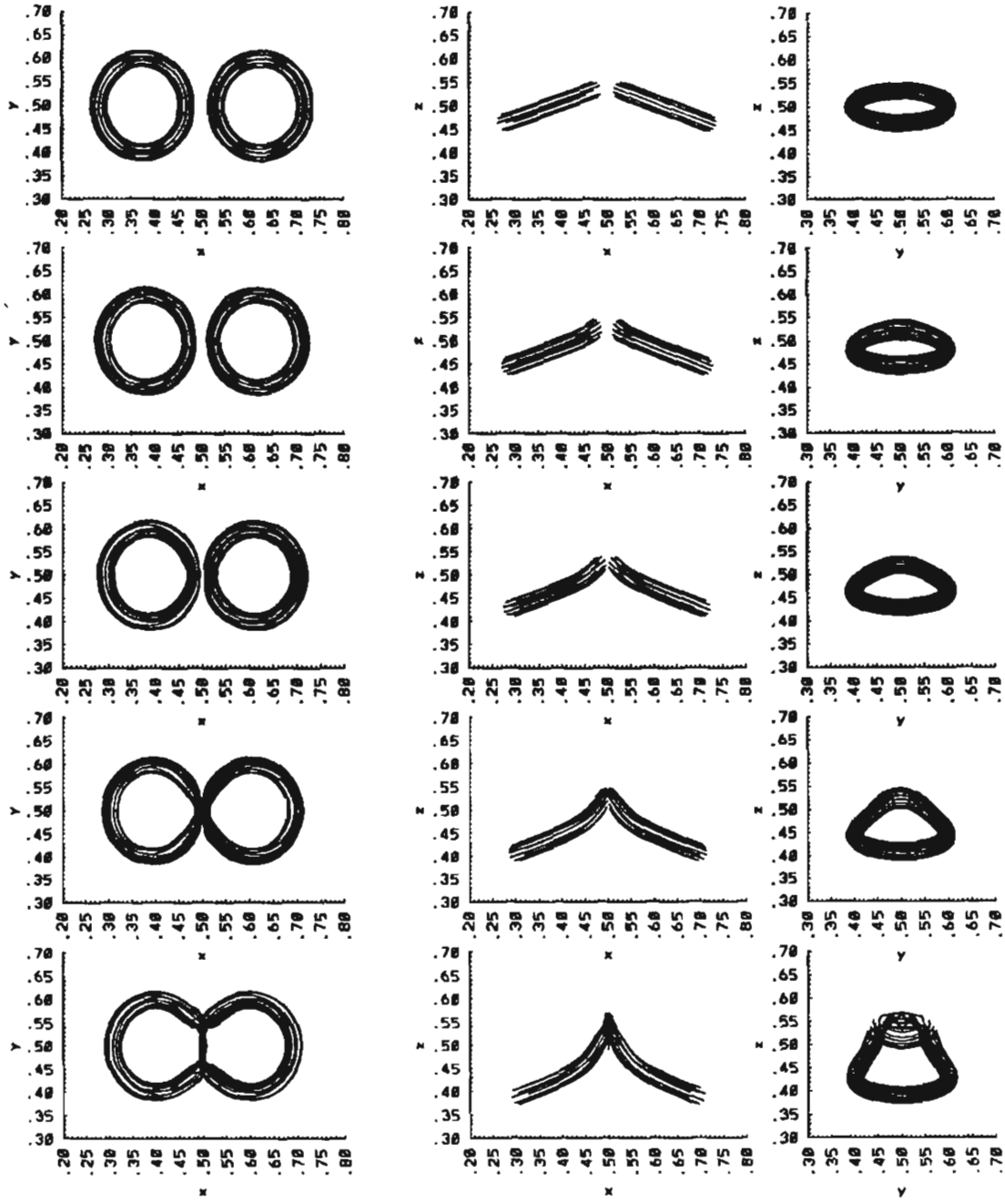

FIG. 1. Three perspectives of the vortex rings at times $t = 0, 16, 32, 48, 64$.

version of "infinite domain" boundary conditions in the finite difference calculation. These were obtained by computing the values of the velocity field due to the vortices on the boundary points of the grid using the direct N-body interaction. As we see in Table III, for the number of vortices presented here, the time spent on the boundary condition calculation was less than 10% of the total CPU time when AMR was used. For larger problems, it is possible to use variations on the ideas in [2] to derive faster boundary condition algorithms as well as more general combinations of boundary conditions.

We comment here that informal timing comparisons were done between the MLC with AMR and an adaptive formulation of a version of the fast multipole method. These comparisons indicated that there was an N below which the direct method was faster than either "fast" method and that above that N there were values of N for which each of the "fast" methods was fastest. We do not claim that the MLC with AMR is faster than the fast multipole method in all cases, rather that there are calculations where one might prefer to use the MLC with AMR, for speed or other considerations, such as the ability to impose boundary conditions.

COMPUTATIONAL RESULTS

The dynamics in the region of contact between two co-rotating inviscid "colliding" vortex rings are interesting

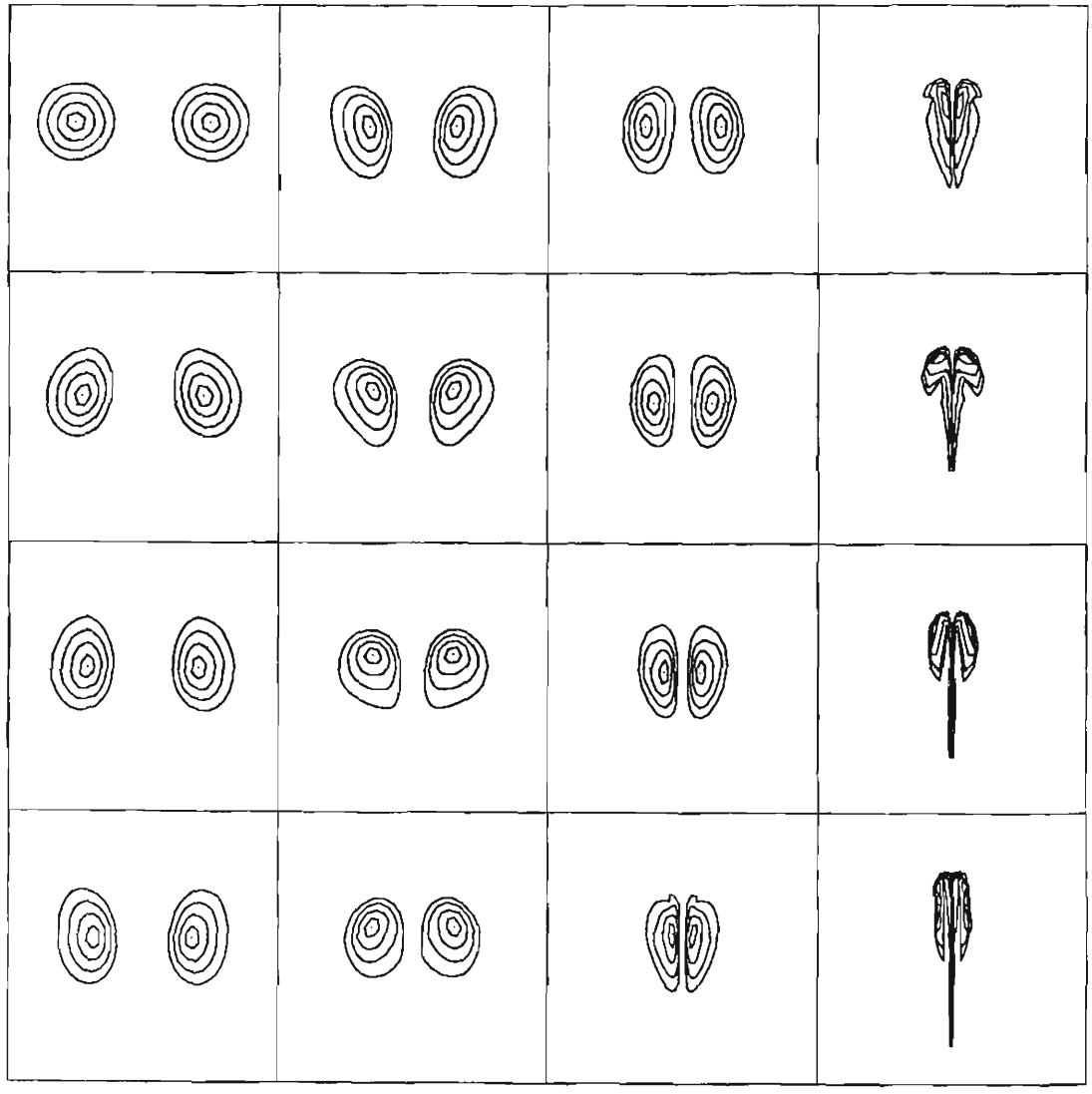

FIG. 2a. Intersection of the vortex rings from the $N = 5490$, $\delta = 0.012$ calculation with the $y = 0.5$ plane. Cross sections are shown at times $t = 0, 8, 12, 16, 20, \ldots, 64$; the time progression starts at the upper left and proceeds downwards.

because in a short time the rings exhibit large vortex stretching and large deformation of the originally circular cross sections. The colliding rings "reconnect" in the sense that the vortex lines from the two rings become very close; however, the vortex method preserves the distinct vortex curves. This problem has been studied previously using vortex methods (see Anderson and Greengard [4] and Winckelmans and Leonard [28]) and was motivated by the experiments of Kambe and Takao [22] and Schatzle [26].

We present here the results from three different calculations of colliding rings using the MLC with AMR. We use the same ring dimensions as in [4]; however, there the largest number of vortex elements used was $N = 5490$ and the values of δ were 0.010, 0.012, 0.015. In [28] the N cited is 2200, but the ring dimensions had different ratios than those we use. The three calculations differ in the original resolution and/or the core radius of the vortex segments. We will refer to these calculations as $N = 5490$, $\delta = 0.012$; $N = 39060$, $\delta = 0.012$; and $N = 39060$, $\delta = 0.006$, where N is the number of vortex segments at $t = 0$ and δ is the core radius. Note that the value of N is the *initial* value; by the end of each calculation the number of computational elements had increased through the refinement of stretched segments as discussed earlier in the description of the

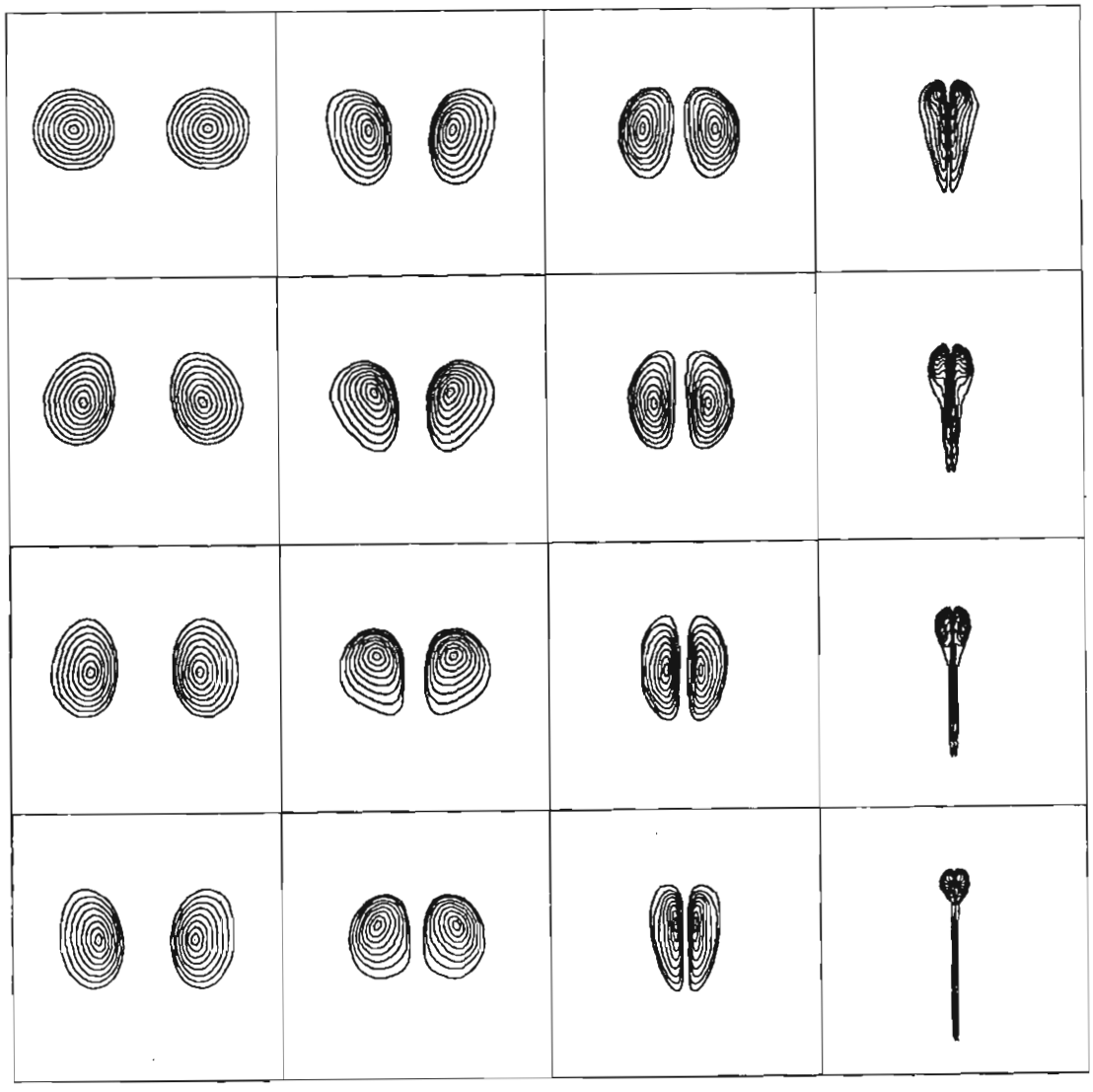

FIG. 2b. Intersection of the vortex rings from the $N = 39060$, $\delta = 0.012$ calculation with the $y = 0.5$ plane. Cross sections are shown at times $t = 0$, 8, 12, 16, 20, ..., 64; the time progression starts at the upper left and proceeds downwards.

method. The time step for the $N = 5490$ calculation was $\Delta t = 2.5 \times 10^{-5}$, the time step for the $N = 39060$, $\delta = 0.012$ calculation was $\Delta t = 1.25 \times 10^{-5}$, and for the $N = 39060$, $\delta = 0.006$ calculation the time step was 1.25×10^{-5} for the first three-quarters of the calculation and $\Delta t = 0.625 \times 10^{-5}$ for the remaining time. These time steps were chosen such that further reduction of the time step did not change the results.

In our calculations, the large radius of each ring is 0.1, and the small radius (radius of the cross section) is 0.02. The rings were centered at $(0.5 \pm 0.125, 0.5, 0.5)$; then each ring was inclined toward the other at an angle of 20° from the $z = 0.5$ plane, so that the co-rotating rings move towards each other. During the calculations the rings were periodically shifted in the z-direction to keep them approximately centered in the computational domain. The total circulation of each ring is 20.0, with the vorticity uniformly distributed across the cross section.

The computational domain covers the box $[0, 1]^3$. The rings are deliberately chosen to be small relative to the domain, so that boundary conditions can be defined on a relatively coarse (8^3) grid without loss of accuracy.

The initialization of the vortex segments was done as follows: first, points equally spaced in the radial direction

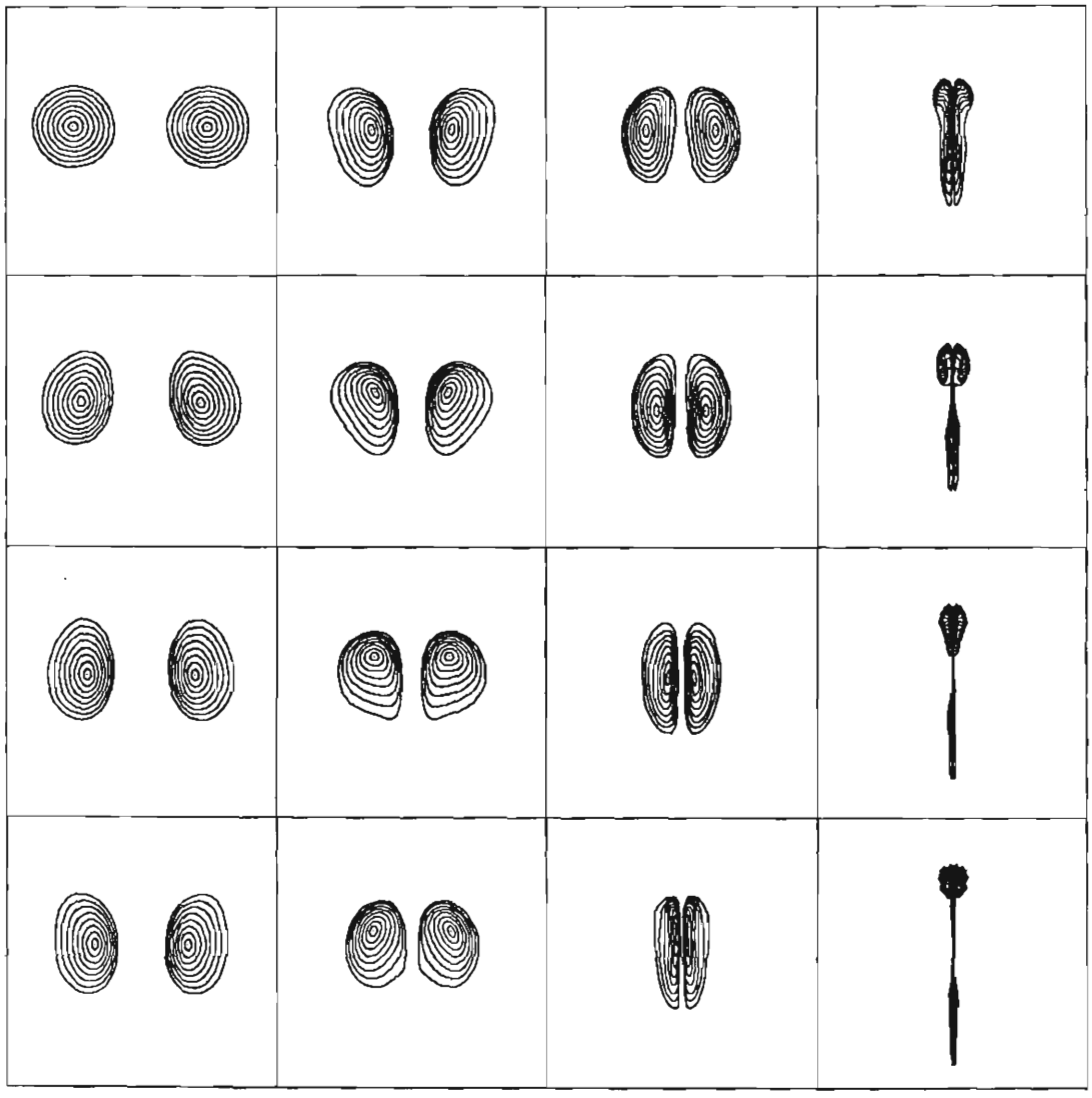

FIG. 2c. Intersection of the vortex rings from the $N = 39060$, $\delta = 0.006$ calculation with the $y = 0.5$ plane. Cross sections are shown at times $t = 0$, 8, 12, 16, 20, ..., 64; the time progression starts at the upper left and proceeds downwards.

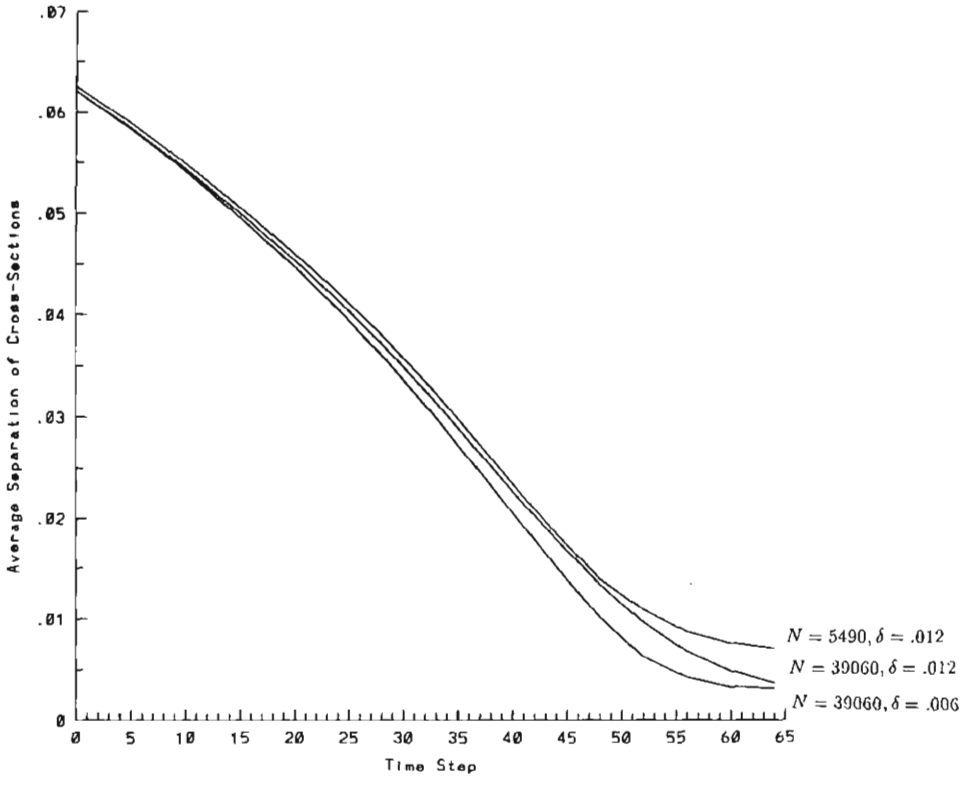

FIG. 3. Time evolution of the average separation of the closest cross sections of the vortex rings for the three calculations.

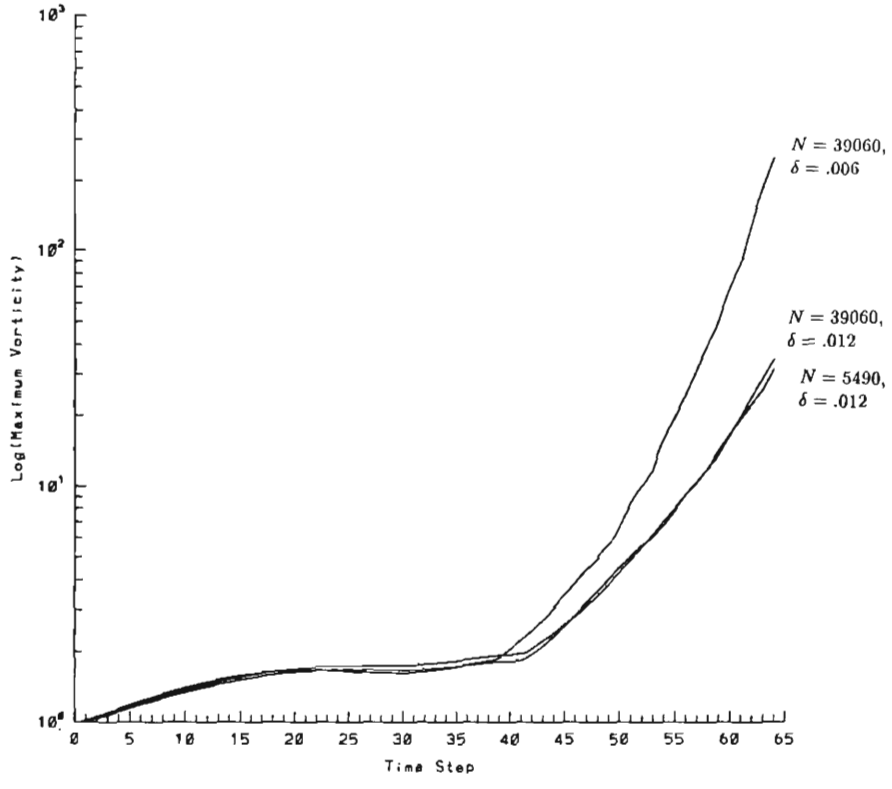

FIG. 4. Time evolution of the log of maximum vorticity.

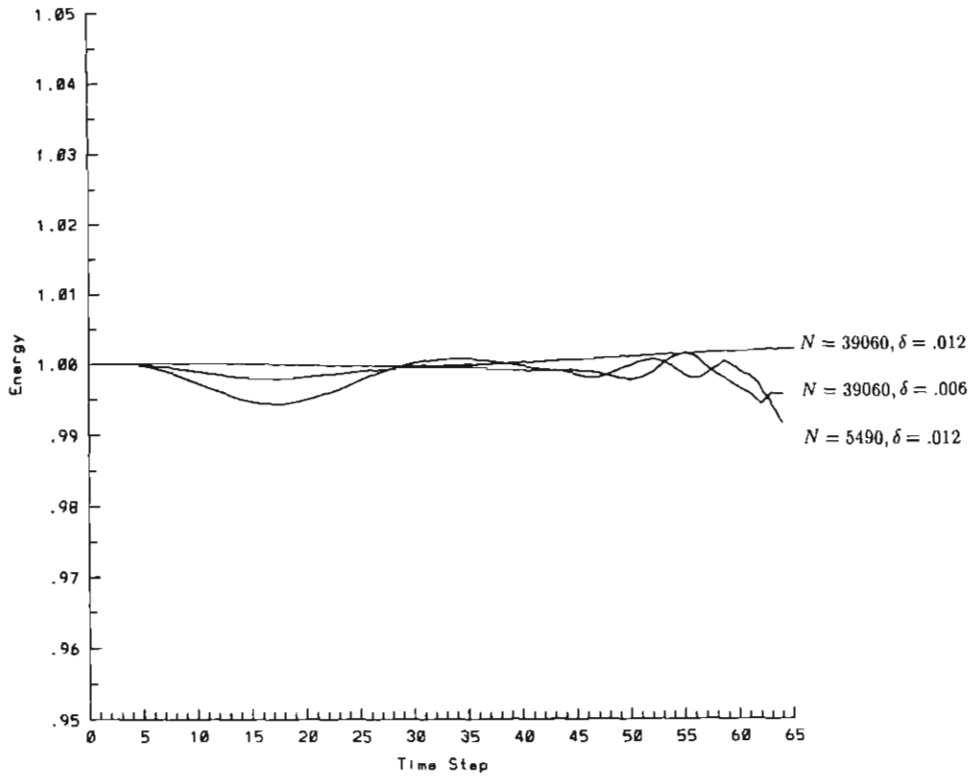

FIG. 5. Time evolution of the normalized energy for the three calculations.

were placed within the cross section of each ring. For the $N = 5490$ calculation, there were four radial locations and a center point; the radial location at $r = n\,\Delta r_0$, $n = 1, ..., 4$, had $6n$ points equally spaced at $\theta = m\,\Delta\theta_n$, $m = 1, ..., 6n$, where $\Delta\theta_n = (1/n)(\pi/3)$. The $N = 39060$ calculations had eight radial locations and a center point; the radial location at $r = n\,\Delta r_1$, $n = 1, ..., 8$, also had $6n$ points spaced at $\theta = m\,\Delta\theta_n$, $m = 1, ..., 6n$. Here $\Delta r_0 = (0.02)/4.5$, $\Delta r_1 = (0.02)/8.5$. A curve through each point was then traced out in the azimuthal direction around the central axis of the ring. These curves were approximated by N_{seg}-sided polygons as described earlier. There were 61 filaments and originally 45 segments per filament in the $N = 5490$ calculation; there were 217 filaments and originally 90 segments per filament in the $N = 39060$ calculation.

Figure 1 shows three perspectives of the two vortex rings at times $t = 0$, 16, 32, 48, 64, from the $N = 39060$, $\delta = 0.012$ calculation. These integer times are multiples of the time step $\Delta t = 2.5 \times 10^{-5}$ of the $N = 5490$ calculation; $t = 64$ is also the final time computed in [4]. Shown in this figure are the computational elements, the vortex segments connected into closed filaments. In this calculation there are 217 filaments per ring; here, however, only the outermost filaments (at $r = 8\Delta r_1$) are shown.

The area of interest in this calculation is the region of contact between the two rings. Figures 2a–c show the evolution of this region in time for the three calculations. The intersection of each ring with the $y = 0.5$ plane is shown at times $t = 0$, 8, 12, 16, 20, ..., 64; this intersection contains 61 or 217 (for $N = 5490$ or $N = 39060$, respectively) points per ring, each point representing a filament. The filaments which were originally at the same radial distance from the center of the cross section are then connected on the plot. Thus, at $t = 0$ we see concentric circles, but as the rings approach each other the cross sections become very noncircular.

We see from Figs. 2a–c that the overall development of the ring in the three calculations is very similar for $t \leqslant 48$ and begins to vary after $t = 48$. At $t = 64$ the most pronounced difference is the presence of "arms" in the $N = 5490$ calculation; these "arms" do not appear in either of the $N = 39060$ calculations, nor in the $N = 3904$, $\delta = 0.012$ calculations of [4]. (Our calculations with $N = 5490$ were repeated using the direct method to verify that the MLC with AMR was not responsible for the observed phenomena; there is no distinguishable difference between the calculations using the direct method and the MLC with AMR.) Since these features do not appear in the more refined calculations, we conclude that they result from insuf-

ficient resolution of the ring. In [4], the initial placement of vortices is on a rectangular rather than on a circular grid, and the arms do not appear; we suggest that the difference in the initial gridding is responsible for the difference in the results and that, therefore, this calculation is underresolved.

We have also observed that the large-scale deformation, shown in Fig. 1, is almost identical for the three calculations. Figure 3 shows the time evolution of the average separation in the x-direction of the cross sections of the rings shown in Fig. 2, and we see very similar qualitative behavior of the different calculations. Figure 4 shows the evolution of the maximum vorticity in time, normalized so that the initial vorticity in each calculation is given a value of 1.0; again, the qualitative behavior is quite similar among

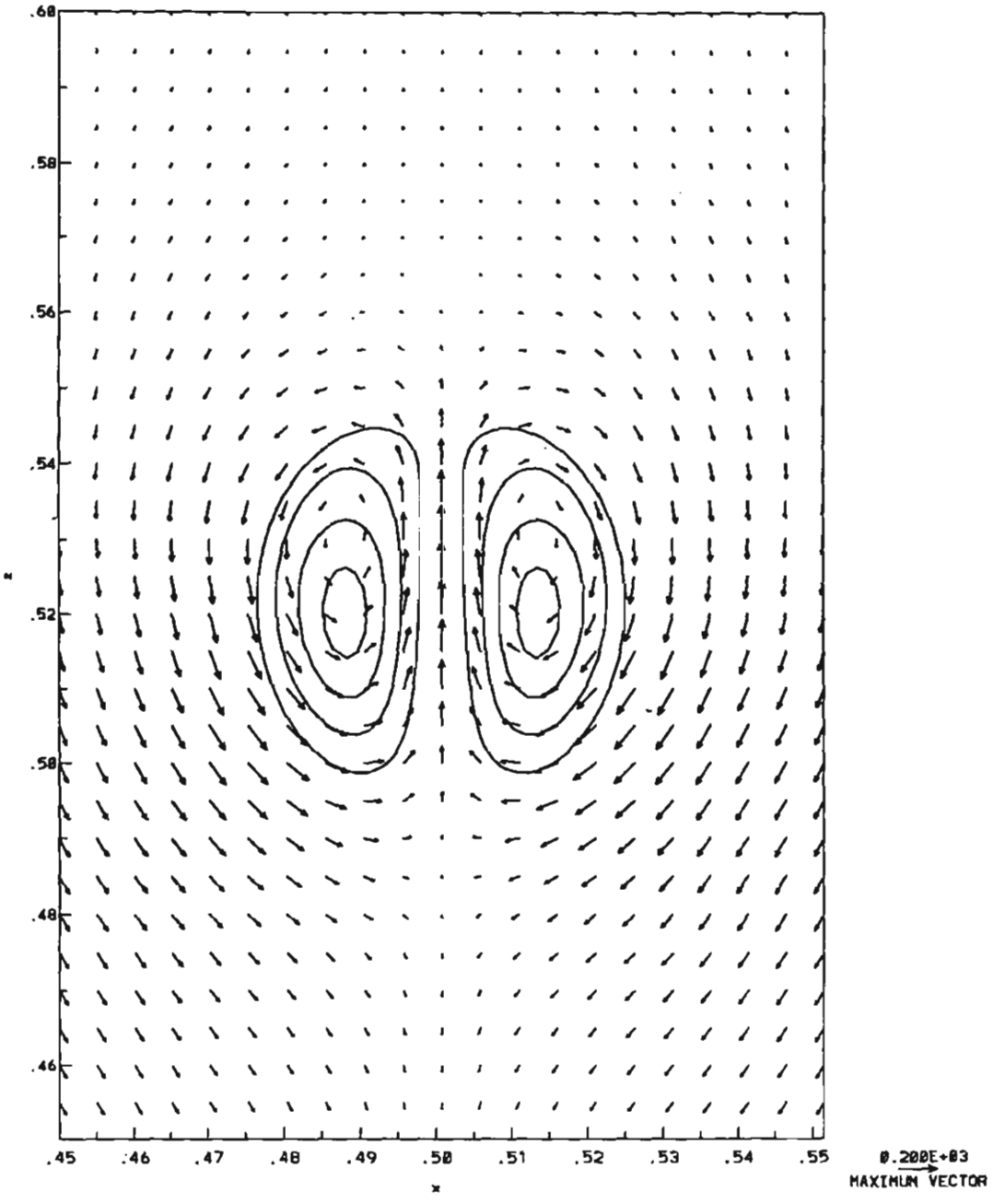

FIG. 6a. Velocity and ring cross sections at $t = 40$, $y = 0.5$, from the $N = 39060$, $\delta = 0.012$ calculation.

the calculations. The quantitative agreement is strong until $t \approx 38$; after this all the curves show a sharp rise, but we do still observe differences due to different δ. Figure 5 shows the time evolution of the energy $E(t)$, normalized by the value $E(t=0)$. The energy is defined as

$$E(t) = \tfrac{1}{2} \sum_{j=1}^{N} \mathbf{u}(\mathbf{x}_j^C, t) \cdot (\mathbf{r} \times \boldsymbol{\omega}_j(t)),$$

where $\mathbf{r}$ is the vector from the origin to $\mathbf{x}_j^C$. There is no explicit mechanism in the vortex method formulation to conserve energy, so the extent to which energy is conserved is a useful diagnostic.

The similarities seen here between the different calculations are consistent with the results in [4], which are from calculations with $N = 3904$ calculations and $\delta = 0.010$,

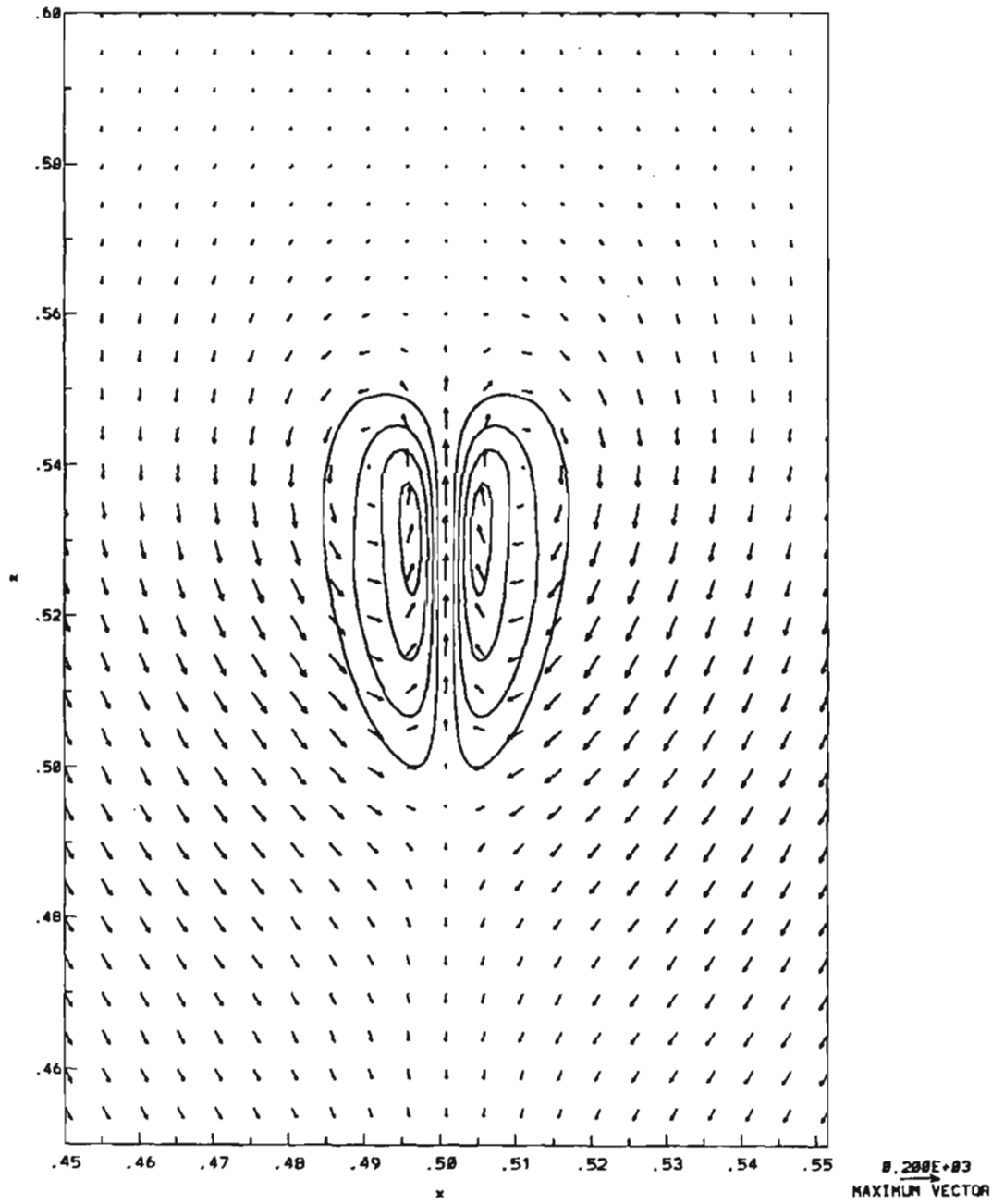

FIG. 6b. Velocity and ring cross sections at $t = 48$, $y = 0.5$, from the $N = 39060$, $\delta = 0.012$ calculation.

0.012, and 0.015. There the overall evolution of the cross section in time is relatively independent of cross section until $t = 48$, and the average separations have the same behavior as those presented here.

From the above figures we conclude that the calculations have converged until $t \approx 40$ in vortex spacing for a fixed δ (compare the $N = 5490$, $\delta = 0.012$ and the $N = 39060$, $\delta = 0.012$ calculations) and are showing similarities as δ decreases. We do not claim to have achieved convergence as $\delta \to 0$, since computations for smaller δ are still prohibitively time-consuming, even with a fast method.

One indication of the resolution of the calculation is the extent to which nearby filaments undergo similar stretching. As we can see in Fig. 1, all of the stretching of the filaments

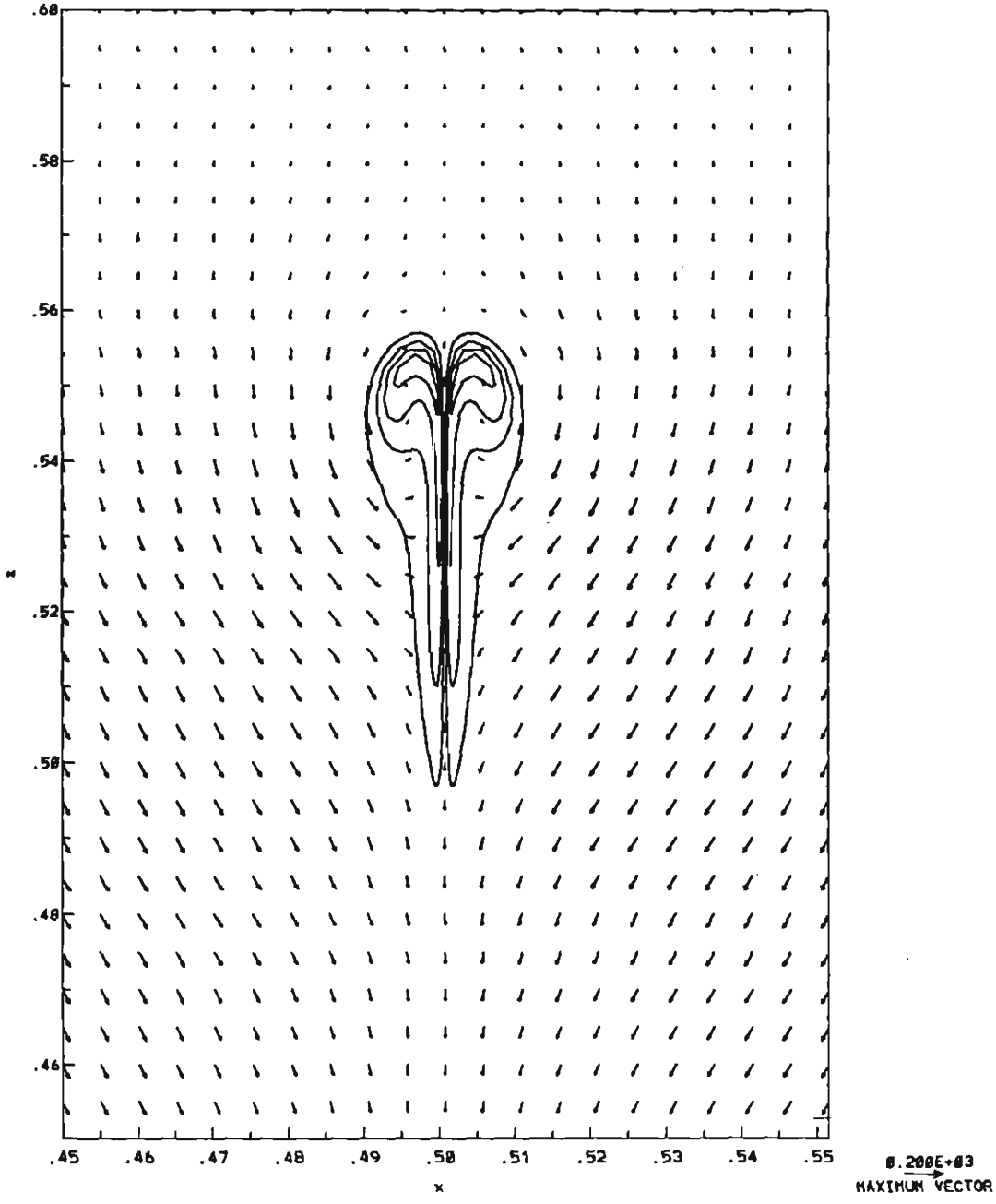

FIG. 6c. Velocity and ring cross sections at $t = 56$, $y = 0.5$, from the $N = 39060$, $\delta = 0.012$ calculation.

occurs in the region of contact between the rings; the rest of each ring undergoes virtually no stretching or deformation. However, even within the region of contact the maximum amount of stretching per filament varies significantly. To illustrate this, we measured the number of filaments per ring which contain vortex segments which have stretched at least 0.6 times the maximum value of stretch at that time. At $t = 48$ for the $N = 39060$, $\delta = 0.012$ calculation, that number is 76 out of 217, over one-third. However, at $t = 56$ the number has decreased to 22 filaments per ring, and by $t = 64$ only 13 filaments have stretched even 60% of the maximum amount. Only five filaments at $t = 64$ have stretched 80% or more of the maximum stretch at that time; ony two filaments have stretched 90% or more.

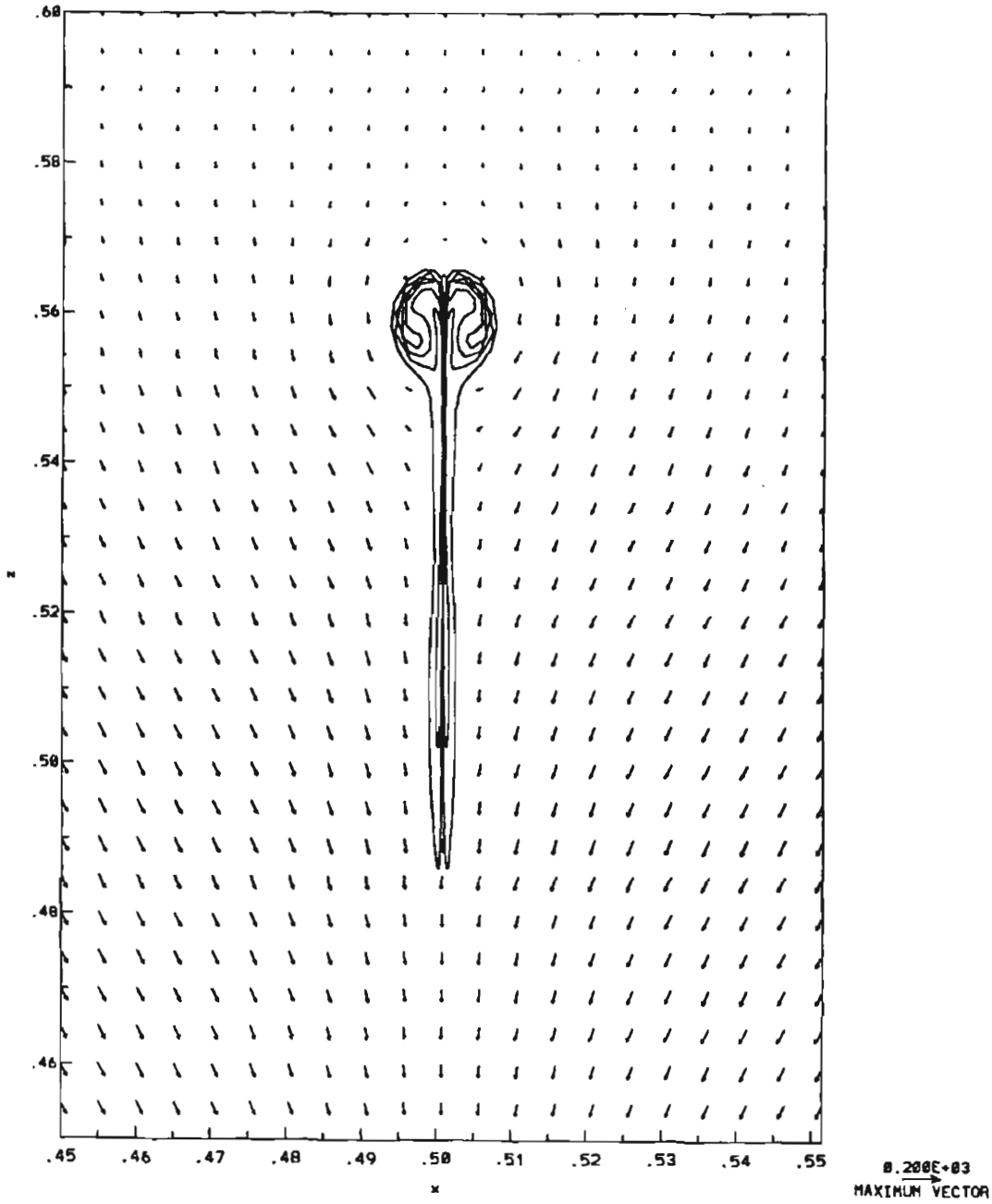

FIG. 6d. Velocity and ring cross sections at $t = 64$, $y = 0.5$, from the $N = 39060$, $\delta = 0.012$ calculation.

For the $N = 39060$, $\delta = 0.006$ calculation the number of filaments per ring at $t = 48$ which have stretched more than 60% of the maximum stretch is only 53 of 217; by $t = 64$ this has reduced to five filaments. There are only two filaments which have stretched 80% or more of the maximum value. We see from this that the value of maximum stretch that we measure at late times depends on the behavior of only a few distinct filaments, and we would need to compute with even more filaments to ensure that we had adequately resolved the intense vortex stretching. However, the absolute magnitude of this stretching seems not to affect the overall late-time dynamics more strongly because *all* of the filaments which undergo large stretching lie along the plane of contact of the rings. By symmetry, the corresponding filaments on the two rings undergo the same stretching, and we obtain the most complete cancellation of precisely these filaments.

In order to better understand the overall evolution of the region of contact between the rings, we superimposed the cross sections of the rings from the $N = 39060$, $\delta = 0.012$ calculation onto a plot of the velocity field in the $y = 0.5$ plane at times $t = 40$, $t = 48$, $t = 56$, and $t = 64$ (see Figs. 6a–d). In these figures only the filaments at every other radial station are plotted, for clarity.

At early times in these calculations, the rings are distinct from one another, and the velocity field can be viewed as a simple superposition of the self-induced velocities of two rings. The velocity which a vortex ring induces upon itself can be broken into two main flows: a uniform translational velocity, here downward and toward the other ring; and a rotation around the core. The translational velocity moves the rings towards each other, and the rotation about the core can be seen in Fig. 2 quite clearly. However, when the rings approach each other, the presence of the core of one

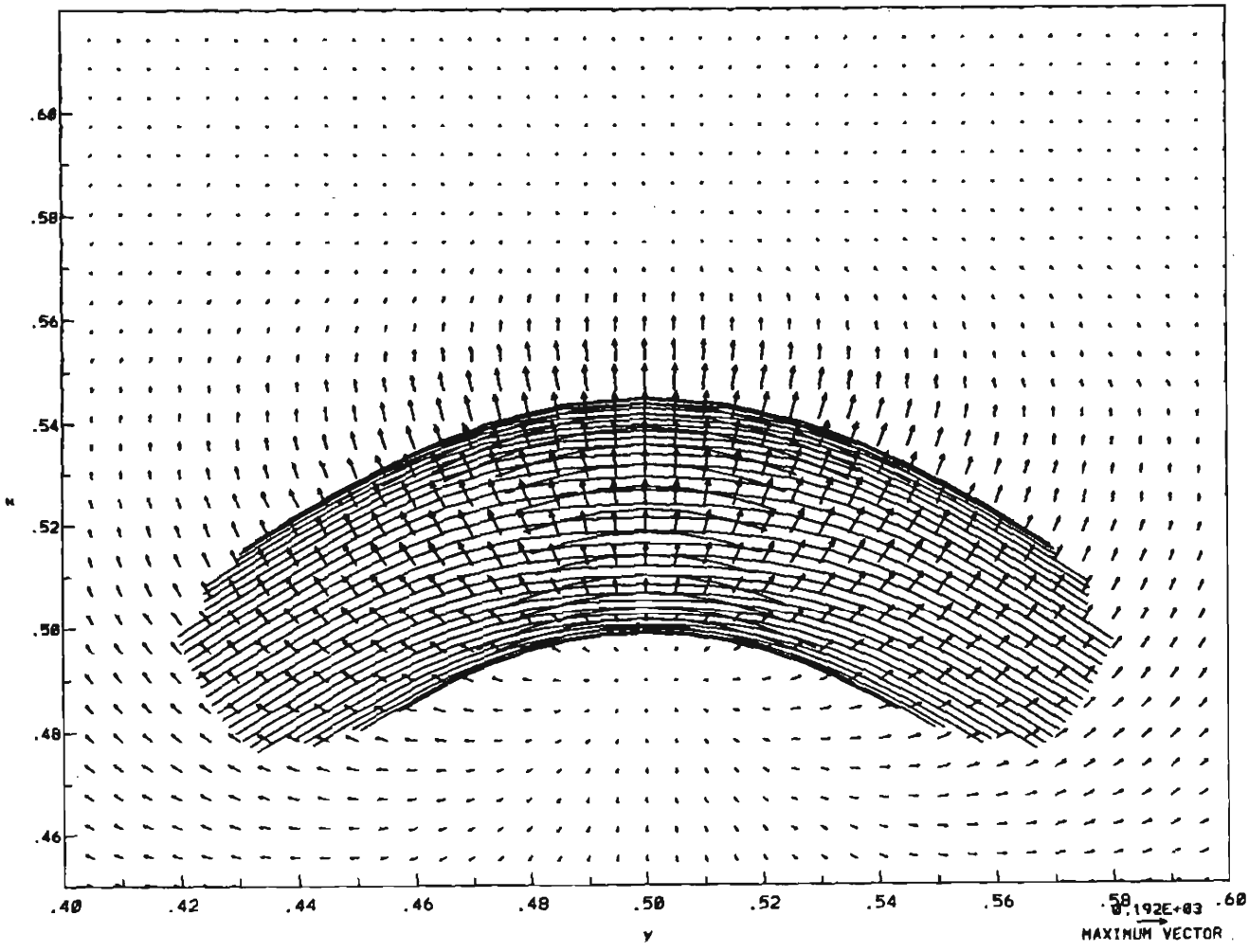

FIG. 7a. Velocity and ring segments at $t = 40$, $|x - 0.5| < 0.03$, from the $N = 39060$, $\delta = 0.012$ calculation.

ring interferes with the rotation of the core of the other, and the cores begin to flatten against each other.

We see at $t = 40$ (Fig. 6a) that the cores have begun to flatten, but the rotational velocity field is still approximately centered at the centers of the cross sections. However, by $t = 48$ (Fig. 6b) there is sufficient cancellation of vorticity in the filaments in each ring closest to the other ring that the rotational velocity field has moved its center outwards from the center of the cross section. By $t = 56$ the velocity field as seen in Fig. 6c swirls around a point on the edge of each cross section. This can only result from the cancellation of vorticity along the plane of contact.

We can see in these plots an explanation of the formation of "arms" in the $N = 5490$, $\delta = 0.012$ calculation. In Fig. 2a at $t = 48$ we see the beginning of the arms at the top of each cross section. The "arm" on each core begins as a small protuberance at $t = 48$, which can be explained by a single (or few) vortex segments being swept upwards by the strong velocity field in the region between the rings. These protuberances are then swept around the cross sections by the velocity field shown in Fig. 6b and 6c. Rather than this fluid being swept back into the "head," as it would have been had it been closer to the $x = 0.5$ plane, it is swept outside the center of the rotational velocity, forming the separate "arm" structure. This only occurs in the coarser calculations, we suspect, because in the more refined calculations the velocity gradient between adjacent filaments is not so large; hence a single (or few) filament is less likely to become separated, as is necessary for the formation of the arms.

It is the velocity in the y-direction which is responsible for the dramatic vortex stretching along the filaments. In Figs. 7a–d, we superimposed all vortex segments at the outer radial station ($r = 8\Delta r_1$) of one ring which were

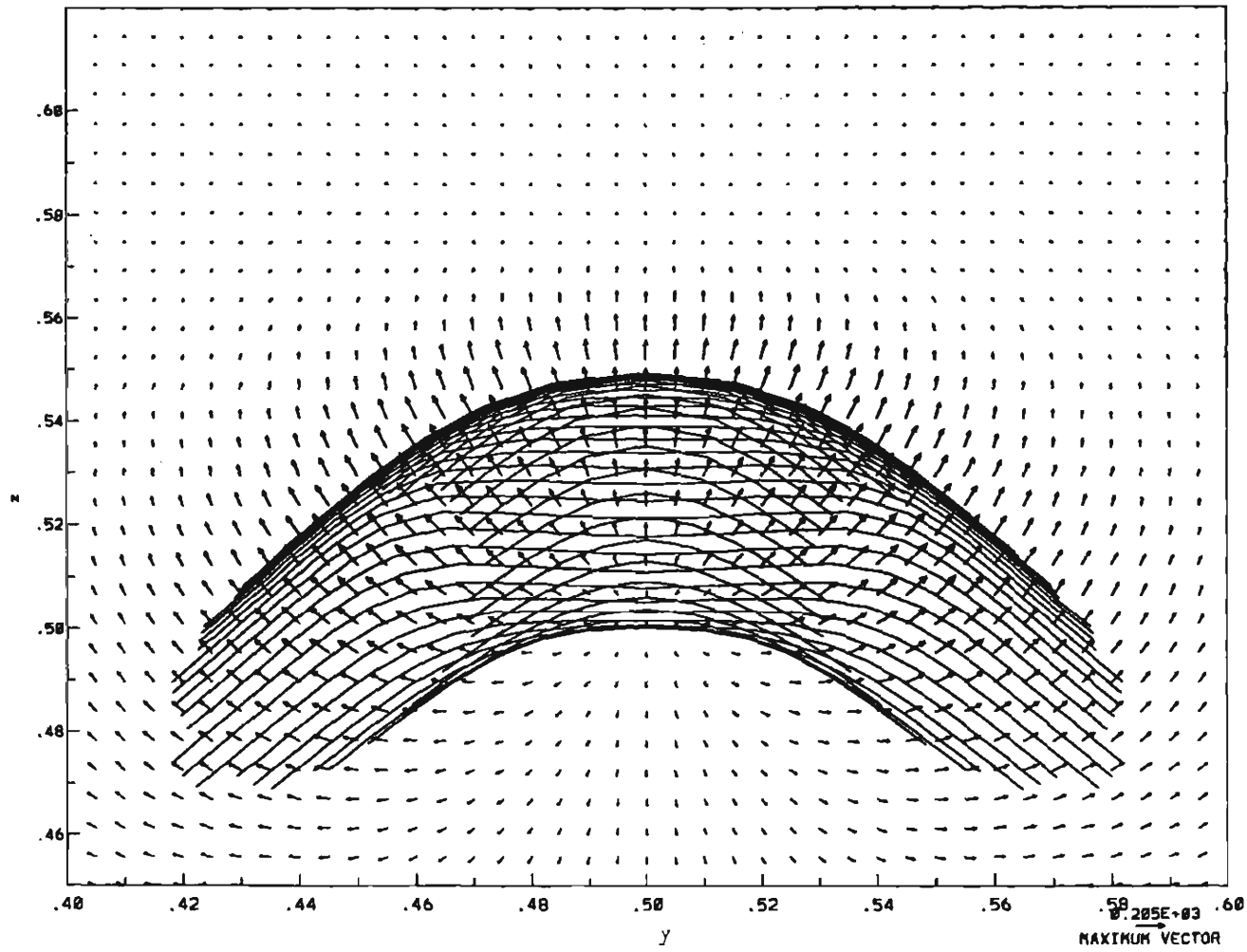

FIG. 7b. Velocity and ring segments at $t = 48$, $|x - 0.5| < 0.03$, from the $N = 39060$, $\delta = 0.012$ calculation.

located in the slab $|x-0.5| \leqslant 0.03$ onto a plot of the velocity in the $x=0.5$ plane. In these plots the remaining segments of each filament should extend behind the page; we are looking at one ring from the vantage point of the other ring. In Figs. 7a and 7b ($t=40$ and $t=48$) we see that the dominant component of the velocity in the y–z plane is in the z-direction, which is the field resulting from two independent rings. However, when the rings begin to form the joint "head–tail" structure they generate a stronger velocity component in the y-direction, and it is this straining field which causes the vortex stretching. This vortex stretching causes the tail, which initially is primarily one-dimensional, to become more sheet-like in the $x=0.5$ plane.

It is important to note the scale of the plots. Even by $t=56$ the "head" structure is entirely containined within one core radius $\delta=0.012$. Thus the interaction of every pair of vortices within the head (for a given $y=$ constant cross section) is mollified by the smoothing function. However, all the filaments do not lie within a single $\delta=0.006$ core radius, and this may account for the differences seen between the two $N=39060$ calculations. More specifically, this may account for the difference in the maximum rate of stretching; for $\delta=0.012$ the maximum stretch at $t=64$ is 38; for $\delta=0.006$ the maximum stretch at $t=64$ is 271, over seven times greater.

Another feature of the vortex dynamics in the region of contact is the intense folding of adjacent segments along a filament. In the $N=39060$ calculations, the angle between any two adjacent segments at $t=0$ was 3.07 radians (176°); at $t=64$ the minimum angle between two segments was 1.59 radians (91.1°) for the $\delta=0.012$ calculation, and 0.085 radians (4.9°) for the $\delta=0.006$ calculation. Thus in the latter calculation we observe the formation of "hairpins," pairs of almost antiparallel adjacent segments.

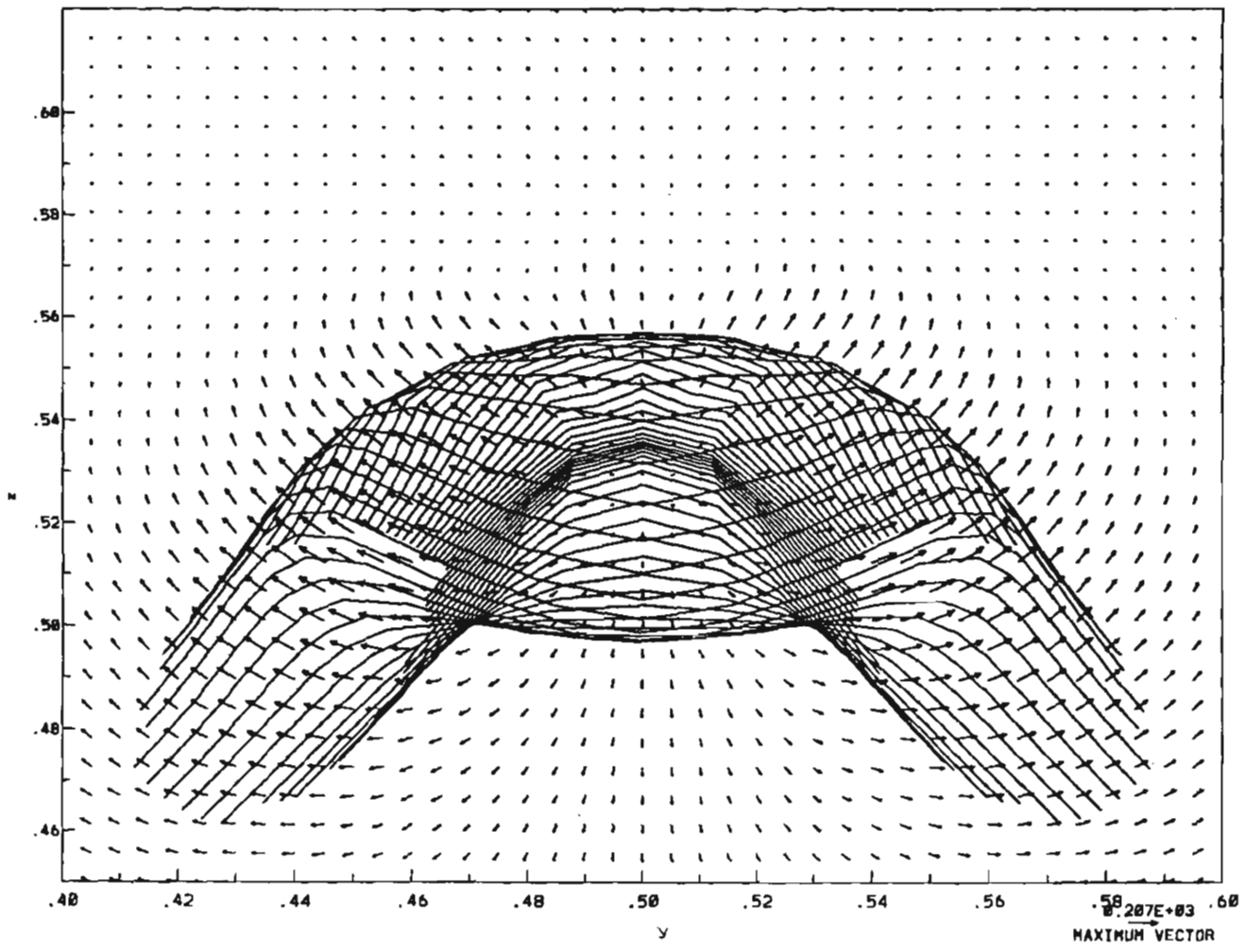

FIG. 7c. Velocity and ring segments at $t=56$, $|x-0.5|<0.03$, from the $N=39060$, $\delta=0.012$ calculation.

A preliminary version of segment refinement, based on curvature in addition to stretching, was implemented in an attempt to maintain better resolution along individual filaments. When this strategy, which maintained the minimum angle between segments above 2.65 radians (151°), was used for the $N = 5490$, $\delta = 0.012$ and the $N = 39060$, $\delta = 0.012$ calculations, many more vortices were introduced in the second half of the calculation, but the benefit of the additional refinement was ambiguous. The filaments did remain smoother, but the shape of the cross sections and the value of maximum stretch were not significantly affected, other than the elimination of the "arms" in the $N = 5490$ calculation. However, the arms were also not present in a $N = 10980$, $\delta = 0.012$ calculation, which had 61 filaments and initially 90 segments per filament. In fact, the number of vortex elements at $t = 64$ for the $N = 5490$ calculation with curvature refinement was only slightly below (11304 vs 11816) the final number of segments for the $N = 10980$ calculation, and conservation of energy was much better maintained in the $N = 10980$ calculation.

CONCLUSIONS

We have presented an algorithm for solving incompressible flow problems which combines adaptive mesh refinement (AMR) with the MLC into a fast adaptive three-dimensional vortex method. This new method maintains the accuracy of the MLC while achieving significant speedup for large number of vortices. Calculations of two colliding inviscid vortex rings using the MLC with AMR show that the resolution with which these rings had previously been studied was not in fact adequate to resolve the small-scale structure. However, using the MLC with AMR, we were

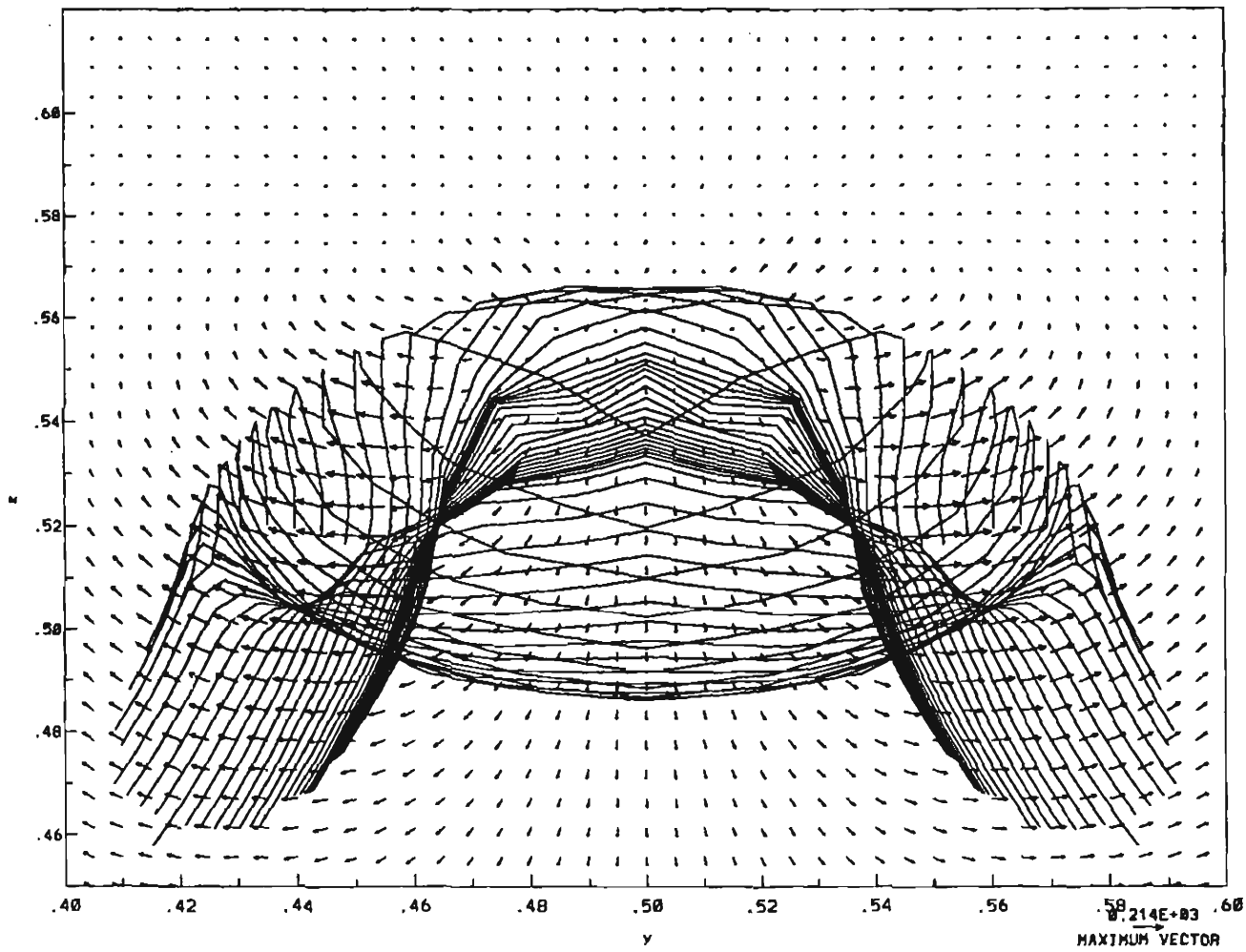

FIG. 7d. Velocity and ring segments at $t = 64$, $|x - 0.5| < 0.03$, from the $N = 39060$, $\delta = 0.012$ calculation.

able to compute with twice the resolution in each spatial dimension, thus demonstrating the usefulness of the new fast method.

REFERENCES

1. C. Anderson, *J. Comput. Phys.* **62**, 111 (1986).
2. C. Anderson, CAM Report 90-14, Department of Mathematics, UCLA, July 1990 (unpublished).
3. C. Anderson and C. Greengard, *SIAM J. Numer. Anal.* **22**, 413 (1985).
4. C. Anderson and C. Greengard, *Commun. Pure Appl. Math.* **42**, 1123 (1989).
5. S. B. Baden, Ph.D. thesis, EECS Department, UC Berkeley, 1987 (unpublished).
6. S. B. Baden, *Proceedings, UCLA Workshop on Vortex Methods, Los Angeles, CA*, in Lecture Notes in Mathematics, edited by C. Anderson and C. Greengard, Vol. 1360, p. 96 (Springer-Verlag, New York/Berlin, 1988).
7. J. E. Barnes and P. Hut, *Nature* **324**, 446 (1986).
8. J. T. Beale and A. Majda, *Math. Comput.* **39**, 1 (1982).
9. J. T. Beale and A. Majda, *Math. Comput.* **39**, 29 (1982).
10. M. J. Berger and J. Oliger, *J. Comput. Phys.* **54**, 484 (1984).
11. M. J. Berger and P. Colella, *J. Comput. Phys.* **82**, 64 (1989).
12. W. L. Briggs, *A Multigrid Tutorial* (SIAM, Philadelphia, 1987).
13. J. Carrier, L. Greengard, and V. Rokhlin, *SIAM J. Sci. Stat. Comput.* **9**, 669 (1988).
14. A. J. Chorin, *J. Fluid Mech.* **57**, 785 (1973).
15. A. J. Chorin, *SIAM J. Sci. Stat. Comput.* **1**, 1 (1980).
16. A. J. Chorin and J. Marsden, *A Mathematical Introduction to Fluid Mechanics* (Springer-Verlag, New York, 1979).
17. L. Collatz, *The Numerical Treatment of Differential Equations* (Springer-Verlag, New York, 1960).
18. C. Greengard, *Math. Comput.* **47**, 387 (1986).
19. L. Greengard and V. Rokhlin, *J. Comput. Phys.* **73**, 325 (1987).
20. O. H. Hald, *SIAM J. Numer. Anal.* **16**, 726 (1979).
21. R. W. Hockney and J. W. Eastwood, *Computer Simulations Using Particles* (McGraw-Hill, New York, 1981).
22. T. Kambe and T. Takao, *J. Phys. Soc. Japan* **31**, 591 (1971).
23. A. Leonard, *J. Comput. Phys.* **37**, 289 (1980).
24. S. F. McCormick, *Multilevel Adaptive Methods for Partial Differential Equations*, Frontiers in Applied Mathematics Series (SIAM, Philadelphia, 1989).
25. L. Rosenhead, *Proc. R. Soc. London Ser. A* **134**, 170 (1931).
26. P. Schatzle, Ph.D. thesis, California Institute of Technology, 1987.
27. L. van Dommelen and E. Rundensteiner, *J. Comput. Phys.* **83**, 126 (1989).
28. G. Winckelmans and A. Leonard, *Proceedings of the SIAM Workshop on Mathematical Aspects of Vortex Dynamics, Leesburg, VA, April 25–27, 1988*, edited by R. Caflisch.