SIAM J. NUMER. ANAL.
Vol. 33, No. 4, pp. 1425–1450, August 1996

008

AN EFFICIENT TWO-DIMENSIONAL VORTEX METHOD WITH LONG TIME ACCURACY*

IBRAHIM BLESS RANERO† AND TOMÁS CHACÓN REBOLLO‡

Abstract. This paper deals with efficient techniques for the numerical solution of two-dimensional free-space incompressible Euler equations. We develop an algorithm for fast computation of velocity in a vortex method based upon discretization of vorticity by finite elements. We prove that the method with fast computation of velocity is numerically stable and convergent with second-order accuracy. Some standard numerical tests show that the algorithm with Delaunay regridding bears good stability and accuracy properties for long integration times, with a relatively low computational cost. Moreover, the algorithm is found to be more accurate than high-order vortex–blob methods with regridding for long enough integration times.

Key words. vortex methods, finite elements, Delaunay regridding, long time accuracy

AMS subject classification. 65N30

1. Introduction.

This paper deals with the numerical solution of two-dimensional free-space incompressible Euler equations by means of vortex methods with finite elements.

Lagrangian methods, and in particular vortex methods, have many favorable practical properties for the numerical simulation of incompressible flows at a high Reynolds number (cf. Leonard [18], Majda [19] for a detailed bibliography and Anderson [1] for flows in bounded domains). Vortex methods essentially introduce no numerical viscosity and are quite accurate and stable, at least for short times (cf. Beale and Majda [5], Perlman [20]). A complete theory of stability and convergence of vortex methods for free-space two- and three-dimensional Euler equations has been developed since the late seventies. In Anderson and Greengard [2] and Beale and Majda [3], [4] an analysis of convergence in l^p-norms, with finite p, may be found. Lately, a theory of convergence in l^∞-norms was developed, mostly by Hou and his collaborators (cf. Hou and Lowengrub [16], Hou [14], Hou [15]). Those works prove that vortex methods are essentially stable in l^p- and l^∞-norms, but stability is conditioned to a relatively high order of consistence. Such "conditioned stability" appears in all convergence proofs of vortex methods, as essentially related to the singularity of the Biot–Savart kernel.

In practice, vortex methods compute quite accurate solutions of Euler equations for relatively short integration times. Beale and Majda [5] and Perlman [20] have tested vortex–blob algorithms with high-order kernels. Those experiments confirm the theoretical predictions of order of convergence for moderate times; however, at later times the high-order accuracy progressively deteriorates. This seems to be due to the progressive increase of local grid size that, in general, takes place as the initial grid is deformed by the flow. This leads to an increasing loss of accuracy in the computation of velocities. As stability is conditioned to enough accuracy, a progressive loss of stability also occurs.

Beale and Majda conclude that to obtain accurate solutions for long integration times, regridding techniques are usually needed. Introducing regridding techniques decreases the local grid size and allows stability and accuracy for longer times. Unfortunately, this introduces increasing levels of numerical diffusion, counterbalancing the main feature of vortex methods.

Our purpose here is essentially to develop a vortex method in which it is possible to reduce the local grid size without introducing numerical diffusion at the grid nodes. Our

*Received by the editors May 19, 1993; accepted for publication (in revised form) September 28, 1994. This research was partially supported by Spanish M. E. C. Project DGICYT PB91-0619.

†Departamento de Matemática Aplicada, Universidad Complutense de Madrid, Avda. de la Complutense, s/n. 20840 Madrid, Spain (ibrahim@sunma2.mat.ucm.es).

‡Departamento de Ecuaciones Diferenciales y Análisis Numérico, Universidad de Sevilla, C/ Tarfia, s/n. 41080 Sevilla, Spain (chacon@numer2.us.es).

work is based upon a vortex method introduced in Chacon and Hou [10]. In this method the vorticity is discretized by means of piecewise linear finite elements. Also, the mesh points of the initial triangulation are transported along the streamlines of the discrete flow. Thus, the vorticity is accurately computed at the mesh points, since the vorticity is conserved along streamlines. This method shares many nice properties of vortex methods, being in particular nondissipative. The method is proved to be stable and convergent with second-order accuracy in the uniform norm. However, the stability is conditioned to an order of accuracy bigger than one, much as in classical vortex methods.

Numerical experiments show that for short integration times this method is quite accurate, but that as time increases, the triangulations generally become degenerated. This produces a fast loss of accuracy in a short time after degeneration. On the other hand, the method uses a technique of computation of discrete velocities that requires an amount of operations of order $O(N^2)$, N being the number of grid nodes in the support of the vorticity. These two drawbacks make the method unfeasible in practical cases.

This paper reports some modifications of this method that render it accurate for very long times, with a relatively low computational cost. At first, we develop a fast technique to compute the discrete velocity in the Chacon–Hou algorithm. This is an adaptation of the technique introduced in Greengard and Rokhlin [12] to our context of discretization of vorticity by finite elements. This reduces the amount of computational work required by the method to $O(N \log^2 N)$. We prove an estimate of error, in terms of the uniform norm of the vorticity, for the computation of discrete velocities by this technique.

Furthermore, we prove that if in the Chacon–Hou algorithm the velocity is computed by this technique with enough accuracy, the numerical solution is still convergent in the uniform norm with second-order accuracy. The proof deals essentially with the fact that our algorithm keeps constant the uniform norm of the discrete vorticity. This allows us to obtain uniform-in-time estimates for the error in the computation of the velocities with the fast algorithm and ensure the stability of the algorithm.

We also prove that if Delaunay regridding is introduced, the algorithm with fast computation of velocity is still convergent with second-order accuracy. Delaunay regridding constructs the triangulation that maximizes the smallest angle of all possible triangulations supported by a given cloud of points. Introducing this regridding technique in our method produces the only effect of redefining the connections between grid points. The values of the discrete vorticity at the grid points remain unchanged. Thus, the local grid size is reduced, without introducing numerical diffusion. The algorithm is convergent independent of the actual time-stepping strategy used to apply Delaunay regridding.

We finally report some numerical experiments dealing with test cases considered by Beale and Majda. We confirm our theoretical expectations on the number of operations in the computation of discrete velocities. We also perform some tests of the modified Chacon–Hou algorithm, introducing Delaunay regridding. These tests show excellent properties of stability and accuracy, for very long time intervals, in the test cases considered. The errors are conserved close to the initial values, and the theoretical convergence orders are confirmed numerically, even for long integration times. We also observe that this algorithm compares advantageously to a desingularized vortex method of the same order, and also to high-order vortex–blob methods with regridding for long enough times of integration.

Our paper is organized as follows. In §2, we describe the algorithm reported in Chacon and Hou [10]. In §3 we develop the technique for fast computation of velocities. Section 4 is devoted to the analysis of the convergence of the modified Chacon–Hou algorithm. Finally, in §5 we report our numerical tests.

2. Description of the base algorithm. In this section we shall describe the vortex method with finite elements introduced in Chacon and Hou [10], as some relevant properties of this method motivate our work.

We are interested in the numerical solution of free-space Euler equations in two space dimensions, with a homogeneous condition at infinity. In vorticity (ω)–streamfunction (Ψ) formulation, these equations are

$$\text{(1)} \qquad \begin{cases} \omega_t + (u \cdot \nabla)\omega = 0 & \text{in } \boldsymbol{R}^2 \times]0, T[, \\ \omega(x, 0) = \omega_0(x) & \text{in } \boldsymbol{R}^2, \\ -\Delta\Psi = \omega & \text{in } \boldsymbol{R}^2, \\ \lim\limits_{|x|\to\infty} \Psi(x) = 0. \end{cases}$$

Here, $u = (u_1, u_2)$ is the velocity field, defined by the two-dimensional Biot–Savart law,

$$\text{(2)} \qquad u(x, t) = (K * \omega)(x) = \int_{\boldsymbol{R}^2} K(x - y)\, \omega(y, t)\, dy,$$

where K is the Biot–Savart kernel,

$$\text{(3)} \qquad K(x) = \frac{1}{2\pi |x|^2} (-x_2, x_1).$$

Equations (1) are equivalent to the usual formulation of Euler equations (cf. Anderson and Greengard [2], Kato [1]).

The equation for the vorticity in (1) may be integrated exactly on the streamlines $X(t; s, \alpha)$ associated to the velocity field u. The curve $t \in [0, T] \to X(t; s, \alpha) \in \boldsymbol{R}^2$ describes the trajectory of a fluid particle whose position at time $t = s$ is the point $\alpha \in \boldsymbol{R}^2$. It satisfies the following ordinary differential equation:

$$\text{(4)} \qquad \begin{cases} \dfrac{dX}{dt}(t; s, \alpha) = u(X(t; s, \alpha), t) & \text{in } [0, T], \\ X(s; t, \alpha) = \alpha. \end{cases}$$

Then,

$$\text{(5)} \qquad \omega(X(t; s, \alpha), t) = \omega(\alpha, s) \quad \forall \alpha \in \boldsymbol{R}^2 \text{ and } \forall s,\ t \text{ in } [0, T].$$

Chacon and Hou show that if ω is approximated by a piecewise polynomial function on polygons, then the corresponding velocity given by (2) may be computed analytically. This, together with (5), suggests that we approximate the vorticity by finite elements on a moving grid whose nodes describe streamlines of the flow. The algorithm of Chacon and Hou is based upon this idea. Although its description needs rather complex notation, we shall provide it as needed throughout this paper.

Consider a triangulation T_h of $\boldsymbol{R}^2$ with triangles $\{\tau_i^0\}_{i \in \boldsymbol{N}}$ and nodes $\{\alpha_j\}_{j \in \boldsymbol{N}}$. We shall assume that h denotes the length of the longest side of all triangles of T_h.

Denote by V_h the space of continuous piecewise affine finite elements on triangulation T_h, defined by

$$\text{(6)} \qquad V_h = \{\, v_h \in C^0(\boldsymbol{R}^2) : v_{h|_\tau} \text{ is affine for all triangles } \tau \in T_h \,\}.$$

A function $v_h \in V_h$ is uniquely determined by the values $v_h(\alpha_j)\ \forall i \in \boldsymbol{N}$.

Assume that we know an approximation $\hat{X}_j^n$ to the point $X(t_n; 0, \alpha_j)$ for each $j \in \boldsymbol{N}$. Define the approximate function $\hat{X}^n \in V_h^n$ to the flow map $X(t; 0, \cdot)$, respectively, in such a way that it verifies

$$\hat{X}^n(\alpha_j) = \hat{X}_j^n \quad \forall\, j \in \boldsymbol{N}.$$

Then, $\hat{T}_h^n = \{\hat{\tau}_i^n = \hat{X}^n(\tau_i^0) \,\forall\, \tau_i^0 \in T_h\}$ defines a triangulation of $\boldsymbol{R}^2$, provided the triangles $\hat{\tau}_i^n$ do not overlap. If this is the case, we also define the piecewise affine finite element space $\hat{V}_h^n$ on triangulation $\hat{T}_h^n$ in the same way that V_h is defined on T_h in (6).

We shall denote by $\hat{P}_h^n\left(x; \{\omega_j\}\right)$ the canonical interpolation operator on $\hat{V}_h^n$, defined by

$$\hat{P}_h^n(\hat{X}_j^n) = \omega_j \quad \forall\, j \in \boldsymbol{N}.$$

We shall also denote by $\{\Phi_j\}_{j\in N}$ the canonical base of V_h. Each Φ_j is uniquely defined by

$$\Phi_j(\alpha_k) = \begin{cases} 1 & \text{if } j = k, \\ 0 & \text{if } j \neq k. \end{cases}$$

We are now ready to state the Chacon–Hou algorithm.

ALGORITHM A. Suppose that ω_0 is of compact support. Denote $\omega_j = \omega_0(\alpha_j)$.

1. Initialization

(i) Triangulation;

$$\hat{X}_j^0 = \alpha_j \quad \text{and} \quad \hat{T}_h^0 = T_h.$$

(ii) Vorticity;

$$\hat{\omega}_h^0(x) = \hat{P}_h^0(x, \{\omega_j\}).$$

(iii) Velocity;

$$\hat{u}_h^0 = K \star \hat{\omega}_h^0.$$

2. Time iteration

(i) Update Lagrangian mesh points by the second-order Adams–Bashforth method,

$$\hat{X}_j^{n+1} = \hat{X}_j^n + \frac{\Delta t}{2}\left[3\,\hat{u}_h^n(\hat{X}_j^n) - \hat{u}_h^{n-1}(\hat{X}_j^{n-1})\right].$$

(ii) Construct a piecewise linear approximation to $X(t; 0, \cdot)$,

$$\hat{X}^{n+1} = \sum_{j\in N} \hat{X}_j^{n+1}\,\Phi_j.$$

(iii) Update vorticity;

$$\hat{\omega}_h^{n+1}(x) = \hat{P}_h^{n+1}(x, \{\omega_j\}).$$

(iv) Update velocity;

$$\hat{u}_h^{n+1} = K \star \hat{\omega}_h^{n+1}.$$

Algorithm A may be viewed as a vortex–blob method with a cutoff function varying in space and time and nonoverlapping smoothing parameter $\delta = h$. Thus, Algorithm A shares with

vortex methods the property of being nondissipative. Furthermore, it is convergent without overlapping the smoothing "blobs."

Chacon and Hou prove that Algorithm A is convergent in the uniform norm with second-order accuracy under some regularity properties of the family of initial triangulations. Accuracy of order higher than one appears to be crucial to ensure the stability of the method.

Some numerical tests show that this method is accurate only for relatively short time intervals, much as are vortex–blob methods (see Perlman [20]). However, at later times the triangulations become degenerated and accuracy is progressively lost in a short time. Thus, regridding techniques are needed to increase the interval of accuracy of the method. Chacon and Hou prove that the method is convergent for longer time intervals if some specific regridding techniques are introduced. This occurs in particular if the grid nodes are regrouped to form new triangulations, so that their regularity is conserved.

3. Fast computation of discrete velocities. The method of computation of discrete velocities introduced by Chacon and Hou requires computations of order $O(N^2)$, where N is the number of grid points in the support of $\hat{w}_h^0$. This makes the algorithm slow even for moderately large grid sizes. In this section we develop an adaptation of the algorithm introduced in Greengard and Rokhlin [12] (cf. also Greengard [13]) that computes an approximation to the discrete velocities by means of Taylor expansions. This reduces the computational complexity to $O(N \log^2 N)$. We omit most of the proofs of the results presented in this section, as they are adaptations of the corresponding ones given by Greengard and Rokhlin. Let T_h be a triangulation of $\boldsymbol{R}^2$ with nodes $\{\alpha_j\}_{j\in N}$. We assume a certain uniformity in the spatial distribution of the grid nodes of T_h. This is needed to ensure the convergence of the interpolation by finite elements (cf. Ciarlet [11]).

We are given a vorticity $\tilde{\omega}_h \in V_h$, where V_h denotes the space of piecewise affine elements on T_h defined in (6). Moreover, we assume that $\tilde{\omega}_h$ has compact support.

3.1. Truncated expansions. Our first goal is to obtain a truncated Laurent expansion that approximates the far-field velocity induced by the vorticity supported by a given subset Q of $\boldsymbol{R}^2$. This vorticity is given by

$$\tilde{\omega}_Q = \sum_{\alpha_i \in Q} \tilde{\omega}_i \, \Phi_i. \tag{7}$$

Define the set

$$Q_h = \left\{ x \in \boldsymbol{R}^2 \,/\, d(x, Q) = \inf_{y \in Q} |x - y| \le h \right\}.$$

Then, $\operatorname{supp} \tilde{\omega}_Q$ may include Q, but in any case $\operatorname{supp} \tilde{\omega}_Q \subset Q_h$.

Denote by $\tilde{u}_Q = (u_1, u_2)$ the velocity induced by $\tilde{\omega}_Q$:

$$\tilde{u}_Q(x) = \int_{\operatorname{supp} \tilde{\omega}_Q} K(x - x') \, \tilde{\omega}_Q(x') \, dx'.$$

Then, $\mathcal{U} = u_1 - iu_2$ is a function of complex variables, analytic in $\boldsymbol{C} \setminus \operatorname{supp} \tilde{\omega}_Q$, as the real and imaginary parts of $\mathcal{U}$ satisfy the Cauchy–Riemann conditions in $\boldsymbol{C} \setminus \operatorname{supp} \tilde{\omega}_Q$. Moreover,

$$\mathcal{U}(z) = \frac{1}{2\pi i} \int_{\operatorname{supp} \tilde{\omega}_Q} \frac{1}{z - z'} \, \tilde{\omega}_Q(x') \, dx', \tag{8}$$

where

$$z = x_1 + ix_2, \quad z' = x_1' + ix_2', \quad x' = (x_1', x_2').$$

This allows us to expand $\mathcal{U}$ in a Laurent series as follows.

THEOREM 3.1. *Assume* $\operatorname{supp}\tilde{\omega}_Q \subset D(z_0, r)$, *where* $D(z_0, r)$ *denotes the complex disc of center* z_0 *and radius* $r > 0$. *Then,*

$$\mathcal{U}(z) = \sum_{k=0}^{\infty} \frac{a_k}{(z - z_0)^{k+1}} \quad \forall\, z \in \boldsymbol{C} \setminus D(z_0, r), \tag{9}$$

where

$$a_k = \frac{1}{2\pi i} \int_{\operatorname{supp}\tilde{\omega}_Q} (z' - z_0)^k \tilde{\omega}_Q(x')\, dx'. \tag{10}$$

Moreover, given $p \geq 1$, *let us denote by* $\mathcal{U}_p(z)$ *the p-term truncated expansion* (9). *Then, the following error estimate holds*:

$$|\mathcal{U}(z) - \mathcal{U}_p(z)| \leq \frac{A}{|z - z_0| - r} \left(\frac{r}{|z - z_0|}\right)^{p+1} \quad \text{if} \quad |z - z_0| > r, \tag{11}$$

where

$$A = \frac{1}{2\pi} \int_{\operatorname{supp}\tilde{\omega}_Q} |\tilde{\omega}_Q|. \tag{12}$$

The coefficients a_k given by (11) may be computed analytically by means of a complex version of Green's theorem. Indeed, as $\operatorname{supp}\tilde{\omega}_Q$ is a reunion of triangles τ of T_h, for each such triangle there exist b_τ, c_τ, and d_τ such that

$$\tilde{\omega}_{Q|_\tau}(x_1, x_2) = b_\tau\, x_1 + c_\tau\, x_2 + d_\tau = \frac{1}{2}\,\bar{\rho}_\tau\, z + \frac{1}{2}\,\rho_\tau\, \bar{z} + d_\tau, \quad \text{where } \rho_\tau = b_\tau + i\, c_\tau.$$

Thus,

$$a_k = \frac{1}{2\pi i} \sum_{\tau \subset \operatorname{supp}\tilde{\omega}_Q} \left[\frac{1}{2}\,\bar{\rho}_z \int_\tau (z - z_0)^k\, z\, dz + \frac{1}{2}\,\rho_\tau \int_\tau (z - z_0)^k\, \bar{z}\, dz + d_\tau \int_\tau (z - z_0)^k\, dz\right].$$

The mentioned complex version of Green's formula is used to compute the integrals in (3.1). It is stated as follows.

LEMMA 3.2. *Let* B *be a bounded measurable subset of* $\boldsymbol{R}^2$ *with Lipschitz boundary* ∂B. *Let* f *and* g *be two functions of complex values defined and analytic in some open set containing* $B \cup \partial B$. *Then,*

$$\int_B \bar{f}'(z)\, g(z)\, dz = \frac{1}{2i} \int_{\partial B} \bar{f}(z)\, g(z)\, dz.$$

Let us now consider a triangle $\tau \in T_h$ included in $\operatorname{supp}\tilde{\omega}_Q$. By taking

$$f(z) = z, \qquad g(z) = (z - z_0)^k\, z,$$

we obtain

$$\int_\tau (z - z_0)^k\, z\, dz = \frac{1}{2i} \int_{\partial\tau} \bar{z}\, z\, (z - z_0)^k\, dz.$$

Also, by taking

$$f(z) = \frac{z^2}{2}, \qquad g(z) = (z - z_0)^k,$$

$$\int_\tau (z - z_0)^k \, \bar{z} \, dz = \frac{1}{4i} \int_{\partial\tau} \bar{z}^2 \, z \, (z - z_0)^k \, dz.$$

A similar choice allows us to find the last term in (3.1).

Finally, the computation of the coefficients a_k reduces to that of polynomials on segments of straight lines, which is obtained analytically.

Our purpose now is to shift the centers of the truncated Laurent expansion obtained above and to convert these expansions into Taylor expansions. We do this in the next three lemmas.

LEMMA 3.3. *Assume* $\operatorname{supp} \tilde{\omega}_Q \subset D(z_0, r)$. *Let* $z_1 \in \boldsymbol{C}$ *be such that* $|z_0 - z_1| > r$. *Then,*

$$\mathcal{U}(z) = \sum_{k=0}^{\infty} \frac{b_k}{(z - z_1)^{k+1}} \quad \forall z \in \boldsymbol{C} \setminus D(z_1, |z_1 - z_0| + r), \tag{13}$$

where

$$b_k = \sum_{l=0}^{k} \binom{k}{l} a_l (z_1 - z_0)^{k-l}. \tag{14}$$

Moreover, if we denote by $\mathcal{U}_p^*$ *the p-term truncated expansion* (13), *the following error estimate holds*:

$$|\mathcal{U}(z) - \mathcal{U}_p^*(z)| \le \frac{A}{|z - z_1| - (|z_1 - z_0| + r)} \left[\frac{|z_1 - z_0| + r}{|z - z_0|} \right]^{p+1}, \tag{15}$$

where

$$A = \frac{1}{2\pi} \int_{\operatorname{supp} \tilde{\omega}_Q} |\tilde{\omega}_Q|.$$

The transformation of Laurent expansions into Taylor expansions is made as follows.

LEMMA 3.4. *Assume* $\operatorname{supp} \tilde{\omega}_Q \subset D(z_0, r)$. *Let* $z_1 \in \boldsymbol{C}$ *be such that* $|z_1 - z_0| > (c + 1)r$ *for some* $c > 1$. *Then,*

$$\mathcal{U}(z) = \sum_{k=0}^{\infty} \gamma_k (z - z_1)^k \quad \forall z \in D(z_1, r), \tag{16}$$

where

$$\gamma_k = (-1)^k \frac{1}{(z_1 - z_0)^{k+1}} \sum_{l=0}^{\infty} \binom{k+l}{l} \frac{a_l}{(z_1 - z_0)^l}. \tag{17}$$

Moreover, given an integer number $q \ge 1$, *define the truncated expansion*

$$\mathcal{U}_{pq}(z) = \sum_{k=0}^{p} \tilde{\gamma}_k \, (z - z_1)^k, \tag{18}$$

where the coefficients $\tilde{\gamma}_k$ *are defined by*

$$\tilde{\gamma}_k = (-1)^k \frac{1}{(z_1 - z_0)^{k+1}} \sum_{l=0}^{p+q} \binom{k+l}{l} \frac{a_l}{(z_1 - z_0)^l}. \tag{19}$$

Then, the following error estimate holds:

$$|\mathcal{U}(z) - \mathcal{U}_{pq}(z)| \leq \frac{A}{r}\frac{1}{c-1}\left[\left(\frac{1}{c}\right)^{p+1} + \frac{c+1}{c-1}\left(\frac{2}{c+1}\right)^{q+1}\right] \quad \forall z \in D(z_1, r), \tag{20}$$

where

$$A = \frac{1}{2\pi}\int_{\operatorname{supp}\tilde{\omega}_Q} |\tilde{\omega}_Q|.$$

Lemma 3.4 provides an error estimate for a truncated expansion $\mathcal{U}_{pq}$ that is defined only with a finite number of data: the $p + q$ coefficients $a_1, a_2, \ldots, a_{p+q}$ furnished by either Theorem 3.1 or Lemma 3.3.

The translation of the centers of the truncated Taylor expansions is given as follows.

LEMMA 3.5. *Given the complex numbers* z_1, z_2; $\rho_1, \rho_1, \ldots, \rho_p$, *the following holds*:

$$\sum_{k=0}^{p} \rho_k (z - z_1)^k = \sum_{l=0}^{p} \tilde{\rho}_k (z - z_2)^l, \tag{21}$$

where

$$\tilde{\rho}_k = \sum_{k=l}^{p} \rho_k (z_2 - z_1)^{k-l}. \tag{22}$$

Observe that as formula (21) is exact, the error bound (20) is still true for the truncated Taylor expansion with shifted center if $z_2 \in D(z_1, r)$ and $z \in D(z_2, r - |z_1 - z_2|)$.

Also, assume that the ρ_k are the coefficients of the Taylor expansion of $\mathcal{U}$ around z_1. Then, in general, the $\tilde{\rho}_l$ are not the coefficients of $\mathcal{U}$ around z_2. This would happen only if the sum in (22) were infinite.

3.2. Description of the algorithm. To describe to some extent the algorithm for fast computation of the discrete velocity, we shall need some specific definitions.

We assume that the support of the vorticity $\tilde{\omega}_h$ is included in a square of sides of length H,

$$\Omega = \left[-\frac{H}{2}, \frac{H}{2}\right[\times \left[-\frac{H}{2}, \frac{H}{2}\right[.$$

We refer to this square as *the computational domain*. We subdivide the computational domain into a family of boxes of decreasing size which will be linked by a hierarchy relation.

Let N be the number of nodes α_m in $\operatorname{supp}\tilde{\omega}_h$, and define the highest level of refinement:

$$J = \log_4 N.$$

Given a level of refinement $j = 0, 1, \ldots, J$, we denote by Q_k^j, $k = 1, 2, \ldots, 4^j$ the squares obtained by splitting each side of Ω into 2^j subintervals of length $h_j = \frac{H}{2^j}$. We denote by β_k^j the center of box Q_k^j. We assume that each box Q_k^j at level j is a Cartesian product of intervals of the form

$$Q_k^j = [a, b[\times [c, d[$$

for some $a < b$ and $c < d$ in $[-\frac{H}{2}, \frac{H}{2}[$. Then, the boxes at level j do not overlap, but their reunion is the whole computational box Ω.

DEFINITION 3.6. *Given $d > 0$, we shall say that two boxes Q_k^j and Q_l^j of the same level j are d-separated if*

$$|\beta_k^j - \beta_l^k| \geq d\, h_j.$$

The fact that two boxes are well separated allows us to approximate uniformly on each one of them the velocity field induced by the vorticity supported by the other one by means of the truncated Taylor development furnished by Lemmas 3.4 and 3.5. To state this result we need to give a formal definition of the aspect ratio of a triangulation, as a measure of its regularity. The *aspect ratio* of a given triangle is the ratio between the diameter of the smallest circle that can circumscribe the triangle and that of the larger circle that can be inscribed in the triangle. The aspect ratio of a triangulation T_h is the largest of all aspect ratios of all triangles of T_h.

LEMMA 3.7. *Given a box Q_k^j of level j, denote by $\tilde{\omega}_k^j \in V_h$ the vorticity supported by Q_k^j:*

$$\tilde{\omega}_k^j = \sum_{\alpha_m \in Q_k^j} \tilde{\omega}_h(\alpha_m)\, \Phi_m. \tag{23}$$

Call $\tilde{\mathcal{U}}_k^j$ the (p, q)-term Taylor expansion associated to $\tilde{\omega}_k^j$ defined by (18). *Define*

$$\tilde{u}_k^j = K * \tilde{\omega}_k^j, \quad \hat{u}_k^j = \left(\Re(\tilde{\mathcal{U}}_k^j), -\Im(\tilde{\mathcal{U}}_k^j)\right).$$

Let $c > 1$ be given. Then, there exists a constant $\lambda > 0$ depending only on the aspect ratio of triangulation T_h, such that if $d = \lambda\,(c+1)$ and box Q_l^j is d-separated from Q_k^j, then

$$\max_{x \in Q_l^j} |\tilde{u}_k^j - \hat{u}_k^j| \leq B\, h_j\, \frac{c+1}{(c-1)^2} \left[\left(\frac{1}{c}\right)^{p+1} + \left(\frac{2}{c+1}\right)^{q+1} \right] \|\tilde{\omega}_h\|_\infty, \tag{24}$$

where B is a constant depending only on the aspect ratio of triangulation T_h.

Proof. As h is the longest length of all triangles of T_h, then

$$\operatorname{supp} \tilde{\omega}_k^j \subset B(\beta_k^j, r_j) \quad \text{with } r_j = \frac{1}{\sqrt{2}}\, h_j + h. \tag{25}$$

Also, as $h_J = \frac{1}{\sqrt{N}}$, there exist two positive constants ν and μ depending only on the aspect ratio of T_h, such that

$$\nu\, h \leq h_J \leq \mu\, h.$$

Thus,

$$r_j \leq \frac{1}{\sqrt{2}}\, h_j + \frac{1}{\nu}\, h_J \leq \lambda\, h_j, \qquad \text{where } \lambda = \frac{1}{\sqrt{2}} + \frac{1}{\nu}.$$

As the boxes are d-separated,

$$|\beta_k^j - \beta_l^j| \geq d\, h_j \geq \frac{d}{\lambda}\, r_j.$$

If we take $d = \lambda\,(c+1)$, we have the hypotheses of Lemma 3.4 with

$$z_0 = \beta_k^j, \quad z_1 = \beta_l^j, \quad r = r_j.$$

Estimate (24) follows immediately. □

This result could be used to derive an algorithm for fast computation of the velocity that would require a number of elementary operations of order $O(N^\sigma)$ for some $\sigma \in]1, 2[$ (cf. Buttke [8], [9]). However, Greengard and Rokhlin's algorithm utilizes the hierarchy of boxes introduced above to compute far-field vorticity effects at the coarsest level possible. This allows us to reduce the computational complexity to $O(N \log^2 N)$. To do so in our case, we still need a certain amount of technical definitions.

DEFINITION 3.8. *Given a box Q_k^j of level $j \geq 1$ and a coarser level $n \leq j$, denote by $P_{k_n}^n$ the only box of level n which includes Q_k^j. Given $d > 0$, we say that a box Q_l^j of level j lies in the* interaction zone *of box Q_k^j if Q_k^j and Q_l^j are d-separated, but there is no level $n = 0, 1, \ldots, j-1$ such that $P_{k_n}^n$ and $P_{l_n}^n$ are d-separated.*

We denote by S_k^j the set of indices l of all boxes Q_l^j of level j which lie in the interaction zone of box Q_k^j. We call the set S_k^j the interaction list *of box Q_k^j.*

DEFINITION 3.9. *Given two boxes Q_k^j and Q_l^j of level $j \geq 1$, we say that Q_l^j lies in the* close action zone *of Q_k^j if it does not lie in the interaction zone of Q_k^j, and there is no level $n = 0, 1, \ldots, j-1$ such that $P_{l_n}^n$ lies in the interaction zone of $P_{k_n}^n$.*

We call close action list *σ_k^j of box Q_k^j the set of indices l of all boxes Q_l^j of level j which lie in the close action zone of Q_k^j.*

Our algorithm for fast computation of velocity may now be described as follows.

ALGORITHM B.

Given $d > 1$ and $p, q \geq 1$ integers, compute an approximation $\hat{u}_h$ of $\tilde{u}_h = K * \tilde{\omega}_h$ in the computational domain Ω as follows.

(1) For each level $j = J, J-1, \ldots, 1$ and each box Q_k^j, $k = 1, 2, \ldots, 4^j$ of level j, compute a p-term truncated Laurent expansion $\mathcal{L}_k^j$ of the velocity induced by the vorticity $\tilde{\omega}_k^j$ supported by box Q_k^j, whose center is the center β_k^j of Q_k^j. This computation is recursively performed from the finest to the coarsest level, following Greengard and Rokhlin. To do this, use Theorem 3.1 and Lemma 3.3.

(2) For each level $j = 0, 1, 2, \ldots, 1$ and each box Q_k^j, $k = 1, 2, \ldots, 4^j$ of level j, compute the (p, q)-term truncated Taylor expansion $\mathcal{U}_k^j$ of the velocity induced by the vorticity $\tilde{\omega}_k^j$ supported by the computational domain Ω, but the close action zone of box Q_k^j. This computation is recursively performed from the coarsest to the finest level, following Greengard and Rokhlin. To do this, use Lemmas 3.4 and 3.5.

(3) For each box Q_k^J, $k = 1, 2, \ldots, 4^J$ of the finest level, compute $\hat{u}_h$ on Q_k^J as follows.

(i) Compute an approximation $\hat{u}_F$ to the velocity induced by the far-field effects of vorticity on Q_k^J by

$$\hat{u}_F(x) = \left(\Re\left[\mathcal{U}_k^J(\zeta)\right], -\Im\left[\mathcal{U}_k^J(\zeta)\right]\right) \quad \text{at each } x \in Q_k^J, \tag{26}$$

where

$$\zeta = x_1 + i\, x_2.$$

(ii) Compute exactly the velocity $\tilde{u}_C$ induced by vorticity supported by the zone of the close action of box Q_k^J:

$$\tilde{u}_C = K * \tilde{\omega}_C, \tag{27}$$

where

$$\tilde{\omega}_C = \sum_{m \in \sigma_k^j} \sum_{\alpha_n \in Q_m^j} \tilde{\omega}_h(\alpha_n)\, \Phi_n.$$

(iii) Compute finally

$$\hat{u}_h = \tilde{u}_C + \hat{u}_F. \tag{28}$$

In what follows, we use the notation Algorithm B $(p,q;d)$ to specify the dependency of Algorithm B upon the parameters p, q, and d.

Once the coefficients of truncated Laurent and Taylor expansions are computed (in our case, by means of Theorem 3.1 and Lemmas 3.3–3.5), Algorithm B is just a slight modification of Greengard and Rokhlin's algorithm. The only difference is that we introduce the integer parameter q to define the number of terms in the truncated Taylor expansions. Because of that, and because the sides of the boxes at the finest level are of order h, its computational complexity is of the same order.

LEMMA 3.10. *The number of elementary arithmetic operations required by Algorithm* B *to compute $\hat{u}_h$ at the nodes of triangulation T_h that lie inside supp $\tilde{\omega}_h$ is of order*

$$N\,(p^2 + q^2). \tag{29}$$

Lemma 3.4 provides fine error estimates of the true truncated Taylor expansions that are used in Algorithm B. This will also allow us to find an error estimate for $\hat{u}_h$, as follows.

THEOREM 3.11. *There exists a constant $\lambda > 0$ depending only on the aspect ratio of triangulation T_h such that for each $c > 1$, the velocity $\hat{u}_h$ computed by* Algorithm *B*(p,q;d), *with $d = \lambda(1 + c)$, verifies*

$$\|\hat{u}_h - \tilde{u}_h\|_{L^\infty(\Omega)} \le K\,H\,\|\tilde{\omega}_h\|_\infty \left[\left(\frac{1}{c}\right)^{p+1} + \left(\frac{2}{c+1}\right)^{q+1}\right], \tag{30}$$

where K is a constant that depends only on c and on the aspect ratio of triangulation T_h.

The proof of Theorem 3.11 is a technical adaptation of that of Lemma 3.7.

4. Convergence of the modified Algorithm A. In this section we state that Algorithm A is still convergent if Algorithm B is used to compute the discrete velocities with enough accuracy.

We call Algorithm A′$(p,q;d)$ the modification of Algorithm A which corresponds to the use of Algorithm B$(p,q;d)$ to compute the discrete velocities. As in Algorithm A, we denote by $\hat{X}_j^n$, $\hat{\omega}_h^n$, and $\hat{u}_h^n$ the Lagrangian mesh points, the discrete vorticity, and the discrete velocity computed by Algorithm A′, respectively. Note that Algorithm B only computes the velocities of points that lie in the necessarily bounded computational domain Ω. For that reason, we assume that Algorithm A′ updates a triangulation $\hat{T}_h^n$ of supp $\hat{\omega}_h^n$, instead of a triangulation of the whole $\boldsymbol{R}^2$, as does Algorithm A. To give a convergence result for Algorithm A′, consider a triangulation T_h of a polygonal domain D_0 that approximates supp ω_0. We use the discrete l^∞-norm

$$\|Y\|_{\infty,h} = \max_{1\le j\le M} |Y_j| \tag{31}$$

for any sequence $Y = \{Y_j\}_{j=1}^M$ defined on the nodes $\alpha_1, \ldots, \alpha_M$ of D_0. We also use the sequences X^n and $\hat{X}^n$, defined by

$$X^n = \{X_j^n\}_{j=1}^M, \qquad \hat{X}^n = \{\hat{X}_j^n\}_{j=1}^M.$$

The following theorem states the convergence of Algorithm A′.

THEOREM 4.1. *Assume that $\omega_0 \in C^2(\boldsymbol{R}^2)$ with compact support. Assume also that the family of initial triangulations $\{\hat{T}_h^0\}_{h>0}$ is regular.*

Given $c > 1$, $\sigma > 1$, $h > 0$, choose two integer numbers p_h and q_h in such a way that

$$\left(\frac{1}{c}\right)^{p_h+1} + \left(\frac{2}{c+1}\right)^{q_h+1} \leq h^{\sigma}. \tag{32}$$

Then, Algorithm A′ *is convergent in the uniform norm with accuracy of order* $\min(2, \sigma)$. *More precisely, given $T > 0$, there exist a separation parameter d_T and two positive constants h_T and Δ_T that depend only on ω_0, T, c and on the aspect ratio $\bar{\gamma}$ of the initial triangulations, such that if $0 < h < h_T$ and $0 < \Delta t < \Delta_T$, then the discrete solution of Euler equations computed by* Algorithm A′(p_h,q_h;d_T) *verifies*

$$\begin{cases} \max\limits_{0\leq t_n\leq T} \|\omega(\cdot, t_n) - \hat{\omega}_h^n\|_{\infty} \leq C\,(\Delta t^2 + h^2 + h^{\sigma}), \\ \max\limits_{0\leq t_n\leq T} \|u(\cdot, t_n) - \hat{u}_h^n\|_{\infty} \leq C\,(\Delta t^2 + h^2 + h^{\sigma}), \\ \max\limits_{0\leq t_n\leq T} \|X(t_n; 0, \cdot) - \hat{X}^n\|_{\infty,h} \leq C\,(\Delta t^2 + h^2 + h^{\sigma}), \end{cases} \tag{33}$$

where the constant C depends only on ω_0, T, c and on the aspect ratio $\bar{\gamma}$ of the initial triangulations.

A sketch of the proof of this result is given in the Appendix. Note that to ensure the second-order accuracy of Algorithm A′, we must take $\sigma = 2$. If we assume $p \simeq q$, (32) will be verified if

$$p \simeq 2\,|\log_{\lambda} h|, \quad \text{where } \lambda = \frac{2}{c+1}. \tag{34}$$

In this case, the computational complexity of our algorithm is of order

$$N\,(\log N)^2. \tag{35}$$

However, if p and q are large enough to have

$$\left[\left(\frac{1}{c}\right)^{p+1} + \left(\frac{2}{c+1}\right)^{q+1}\right] \sim \varepsilon,$$

where ε is the precision of the computer actually used, then the computational complexity of Algorithm B will be of order

$$N\,(|\log \varepsilon|)^2, \tag{36}$$

just as is that of Greengard and Rokhlin. However, in practical cases h is always much bigger than ε, so that (35) applies. The fact that Algorithm A′ is still convergent with second-order accuracy if (32) holds allows us in practice to save an amount of computational work of order higher than N.

We should remark also that in practice Algorithm A′ must be combined with regridding techniques to improve its stability and long time accuracy. It is interesting to observe that Theorem 4.1 is still true if Delaunay regridding is used to construct the triangulations, independent of the time-stepping strategy used. This happens because of the special properties of this kind of regridding; i.e., it gives the triangulation with the largest aspect ratio of all possible triangulations lying on a given cloud of nodes. Also, it keeps constant the discrete l^{∞}-norm defined by (31) and the continuous L^{∞}-norm of piecewise affine functions. This allows us to reproduce step by step the proof of Theorem 4.1.

5. Numerical tests. In this section we report some numerical results that show the numerical performances of Algorithms A′ and B. Specifically, we observe in practical cases that the use of Algorithm B to compute the discrete velocities effectively takes an amount of operations of order $N \log^2 N$, as stated in Theorem 3.11. We also observe that the numerical order of convergence of Algorithm A′ agrees with the theoretical order stated in Theorem 4.1, even for long integration times. Moreover, our experiments show that the use of Delaunay regridding allows us to keep the errors in Algorithm A′ at levels very close to the initial ones, for very long time intervals, without introducing numerical diffusion. Finally, we observe that Algorithm A′ compares advantageously for long times with desingularized vortex methods of the same convergence order and even with high-order vortex–blob methods with regridding. In our tests we have considered two steady solutions (u, ω) of Euler equations (1). These solutions have been used by Beale and Majda and Perlman, among others, in the numerical study of the accuracy of vortex–blob methods (cf. Beale and Majda [5]), Perlman [20]). In both cases, the vorticity is a smooth radially symmetric function, $\omega = \omega(r)$, with support in the unit circle. Our test cases now follow.

Test Case 1.

$$\omega(x) = \begin{cases} (1 - |x|^2)^7 & \text{if } |x| \le 1, \\ 0 & \text{if } |x| > 1. \end{cases} \tag{37}$$

Test Case 2.

$$\omega(x) = \begin{cases} (1 - r)^2 (1 - 2r) (1 + 4r) & \text{if } |x| \le 1, \\ 0 & \text{if } |x| > 1. \end{cases} \tag{38}$$

The corresponding velocity field is given by the expression

$$u(x) = \frac{1}{r^2} (-y, x) \int_0^r s\, \omega(s)\, ds. \tag{39}$$

The streamlines of these rotating flows are circles. In both cases, the angular velocity varies in the radial direction, so that a large tangential stretching takes place as time goes on. Thus, the corresponding flow maps produce a large geometrical distortion. Because of these properties, these flows seem to be appropriate to test the performances of our algorithm for long integration times.

In Test Case 1 (TC1), the vorticity ω is of class $C^6(\boldsymbol{R}^2)$ and has constant sign. The larger tangential stretching takes place in the interval $r \in [0.3, 0.6]$, approximately. Particles with maximum speed, situated approximately on $|x| = 0.4$, complete one rotation at time $t = 4\pi$, while those on $|x| = 1$ complete one rotation at time $t = 32\pi$. In Test Case 2 TC2, the vorticity is less smooth and is only of class C^1. Moreover, ω changes sign and the velocity profile is not monotone for $r < 1$. Thus, this problem is a more severe test than TC1. In our convergence results, a basic hypothesis is that the angles of the triangles of the triangulations are bounded from below by a constant independent of the grid size. In our computations we have used initial triangulations whose angles are always bigger than $\pi/4$. This is done by uniformly spacing the vertices in the radial and angular directions. Thus, for these triangulations the grid size h may be defined as $h = 1/M$, where M is the number of radial subdivisions. In Fig. 1 we have drawn one such initial triangulation corresponding to $h = 1/10$, constructed with the help of MODULEF finite element library (cf. Bernadou et al. [6]).

5.1. Fast computation of velocity. Our first set of tests focused on the analysis of the practical performances of Algorithm B. To do this, we computed the initial discrete velocity $\hat{u}_h^0$ in TC1 by direct calculation of $K * \hat{\omega}_h^0$ and also more approximately by means of Algorithm B,

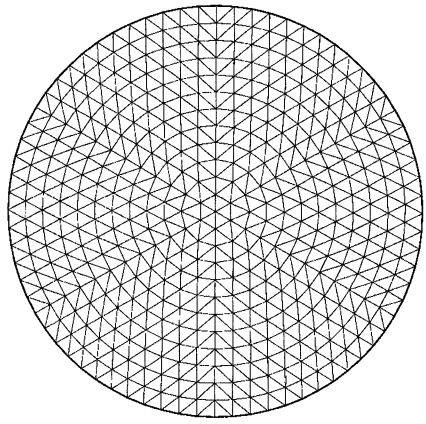

FIG. 1. *Initial triangulation for Algorithm* A′ *with grid size* $h = 1/10$. *Triangulation of the unit circle with* 469 *grid points and* 864 *triangles.*

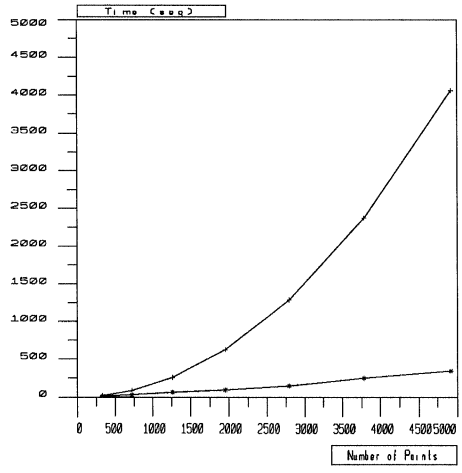

FIG. 2. *Computing time required by direct calculation (line marked by* * *symbols) and by Algorithm* B *(line marked by* + *symbols) to compute the initial discrete velocity. The first one grows quadratically, while the second one grows almost linearly.*

allowing an error of order h^3 in this last calculation. In Fig. 2 we represent a comparison between the number of operations taken by both calculations. We may observe that the number of operations taken by Algorithm B grows almost linearly, while the one taken by the direct calculation is very closely quadratic. Thus, we save a large amount of computational work for small h.

5.2. Convergence order. Our second set of tests was performed in order to analyze the agreement between the theoretical and the numerical orders of convergence of Algorithm A′, including the use of Delaunay regridding. To do this, we set a quality test on triangulation $\hat{T}_h^n$; i.e., if the smallest angle becomes smaller than a preset minimum value, Delaunay regridding is performed. In Figs. 3 and 4 we represent, respectively, the grids corresponding to $h = 1/12$ at time $t = 6.8$ before and after using Delaunay regridding. The minimum angle was set to be one-tenth of the smallest initial angle. A large improvement of the quality of the grid may be observed.

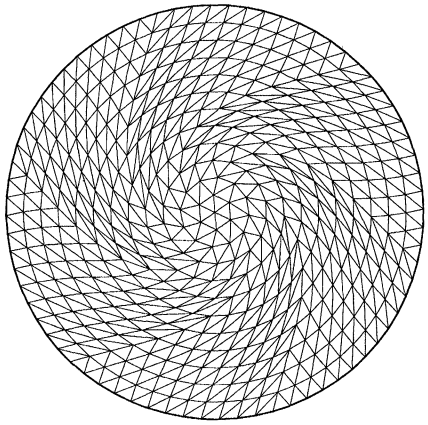

FIG. 3. *Triangulation corresponding to* $h = 1/12$ *at* $t = 6.8$ *constructed by Algorithm* A′ *without regridding.*

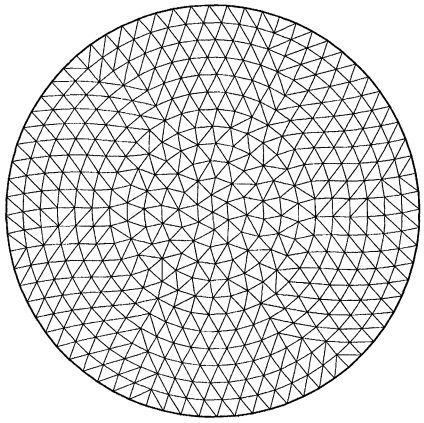

FIG. 4. *Triangulation corresponding to* $h = 1/12$ *at* $t = 6.8$ *constructed by Algorithm* A′ *after applying Delaunay regridding.*

To measure the errors in all test cases we used a discrete l^∞-norm, because our convergence results are stated in uniform norms. To describe this discrete norm, let us denote by Λ_h the set of nodes α_j of the initial triangulation that lies inside the unit circle C. The discrete norm of an error function E defined on Λ_h is defined as follows:

$$\|E\|_{\infty,h} = \max_{\alpha_j \in \Lambda_h} |E_j|. \tag{40}$$

The errors in velocity, $e_u(t, h)$, and vorticity, $e_\omega(t, h)$, have been normalized by the exact L^∞-norm of u and ω, respectively, in C:

$$e_u(t_n, h) = \frac{\|u - \hat{u}_h^n\|_{\infty,h}}{\|u\|_{L^\infty(C)}}, \quad e_\omega(t_n, h) = \frac{\|\omega - \hat{\omega}_h^n\|_{\infty,h}}{\|\omega\|_{L^\infty(C)}}. \tag{41}$$

The numerical order of convergence in velocity, for instance, was computed by comparing the errors corresponding to two different values of h as follows:

$$p_u(t_n; h_1, h_2) = \frac{\log [e_u(t_n, h_1)/e_u(t_n, h_2)]}{\log(h_1/h_2)}. \tag{42}$$

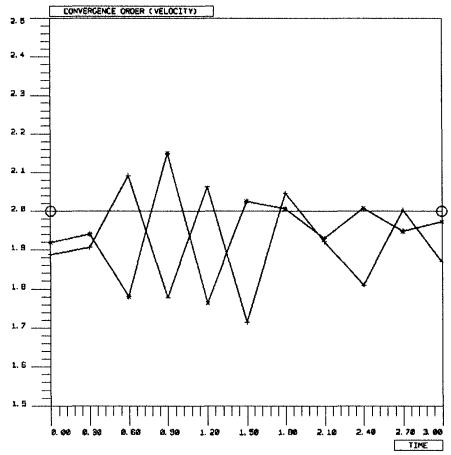

FIG. 5. *Numerical convergence order in velocity of Algorithm* A′, *applied to* TC1, *for a short time interval. The lines marked by* * *and* + *symbols correspond, respectively, to* $h_1 = 1/12$, $h_2 = 1/16$ *and to* $h_1 = 1/16$, $h_2 = 1/20$ *to compute the numerical convergence order. The corresponding theoretical convergence order is* $p = 2$ *(line marked by 0 symbols).*

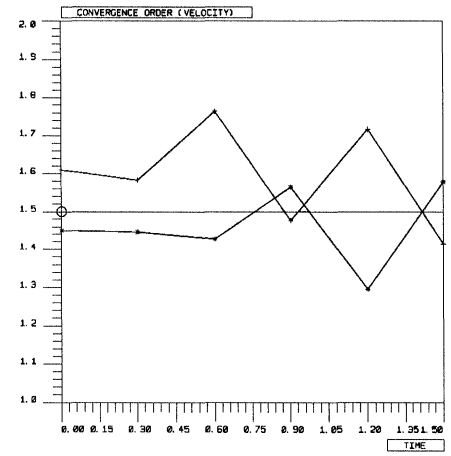

FIG. 6. *Numerical convergence order in velocity of Algorithm* A′, *applied to* TC1, *for a short time interval. The lines marked by* * *and* + *symbols correspond, respectively, to* $h_1 = 1/12$, $h_2 = 1/16$ *and to* $h_1 = 1/16$, $h_2 = 1/20$ *to compute the numerical convergence order. The corresponding theoretical convergence order is* $p = 1.5$ *(line marked by 0 symbols).*

In Figs. 5 and 6 we represent the behaviour of the numerical convergence order in velocity for TC1 for short times. Figure 5 corresponds to a theoretical convergence order $p = 2$, obtained by choosing $\sigma = 2$ in Algorithm A′. Figure 6 corresponds to $p = 1.5$, obtained by choosing $\sigma = 1.5$. The sharp oscillations observed in these curves occur when Delaunay regridding is performed. In both cases there is a good agreement between the computed convergence order and the theoretical one. Furthermore, the curves corresponding to smaller values of h are globally closer to the theoretical convergence orders. Note that in all cases the computed orders oscillate around the theoretical value.

It is interesting to observe that the convergence order of Algorithm A′ may be preset to any value $p \in]1, 2]$ by simply choosing the parameter $\sigma = p$ at the beginning of the run. However, taking σ smaller than 2 would produce a waste of computational work, as the computational complexity of Algorithm A′ is of order $N \log^2 N$ for any value of σ. In what follows we shall always take $\sigma = 2$.

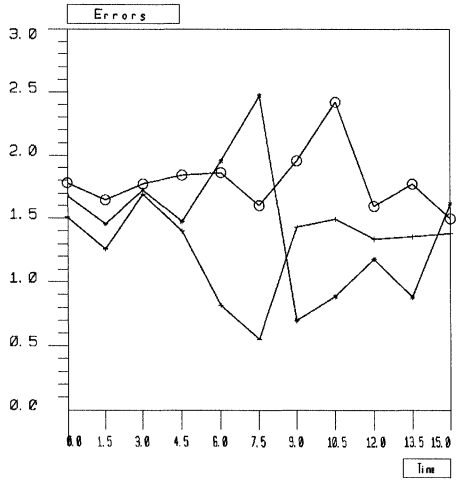

FIG. 7. *Time evolution of numerical convergence order in vorticity of Algorithm* A′, *applied to* TC2, *for the time interval* [0, 15]. *The lines marked by* +, *, *and* 0 *symbols correspond, respectively, to* $h_1 = 1/12$, $h_2 = 1/16$; $h_1 = 1/16$, $h_2 = 1/20$; *and to* $h_1 = 1/16$, $h_2 = 1/20$ *to compute the convergence order. A good agreement with the theoretical prediction* $p = 2$, *which improves for smaller values of* h, *is observed for the whole time interval.*

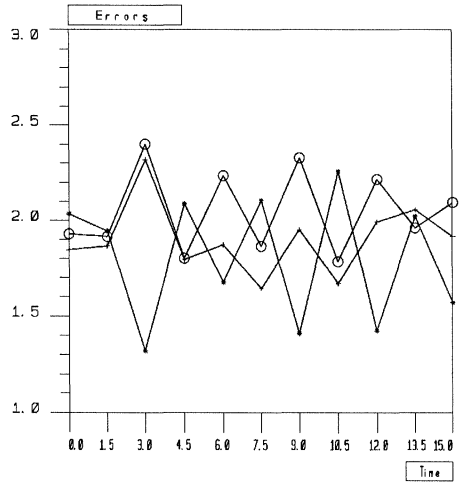

FIG. 8. *Time evolution of numerical convergence order in velocity of Algorithm* A′, *applied to* TC2, *for the time interval* [0, 15]. *The lines marked by* +, *, *and* 0 *symbols correspond, respectively, to* $h_1 = 1/12$, $h_2 = 1/16$; $h_1 = 1/16$, $h_2 = 1/20$; *and to* $h_1 = 1/16$, $h_2 = 1/20$ *to compute the convergence order. A good agreement with the theoretical prediction* $p = 2$, *which improves for smaller values of* h, *is observed for the whole time interval.*

Figures 7 and 8 show the numerical convergence order in vorticity and velocity for TC2 during a relatively long integration time. The sharp oscillations of the former test are still observed. However, again the curves corresponding to smaller values of h are closer to the theoretical order $p = 2$. Note that the numerical orders in velocity are closer to $p = 2$ than those in vorticity. This is probably due to the higher regularity of the velocity field. Note also that the agreement holds even for long times.

5.3. Behaviour for long integration times. Our third set of numerical experiments deals with the analysis of the long time behaviour of errors due to Algorithm A′.

At first, we tested the effect of introducing Delaunay regridding in Algorithm A′. In Fig. 9 we represent the time evolution of the percent relative errors in velocity corresponding

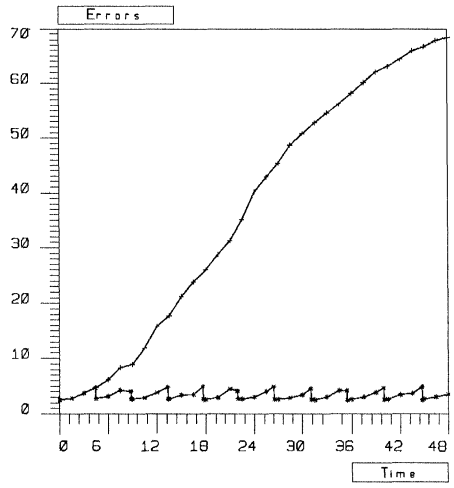

FIG. 9. *Time evolution of relative errors in vorticity of Algorithm* A′ *with and without using Delaunay regridding (marked by* * *and* + *symbols, respectively). The second one grows exponentially while the first one keeps close to the initial error.*

to Algorithm A without regridding and to Algorithm A′ using Delaunay regridding, applied to TC1 with $h = 1/12$. We made a run in the time interval [0, 50], which appears as quite a long time for the test case considered. Indeed, in that interval the fastest points in supp ω completed more than three rotations, while the slowest ones did not complete half a rotation. Note that Algorithm A is still defined when the triangulation $\hat{T}_h^n$ becomes degenerated. Indeed, in this case it is still possible to compute directly the exact velocity $\hat{T}_h^n$. We may observe that in this case the error grows exponentially until attempting relative values of more than 50% at $t \simeq 50$. When Delaunay regridding is used, there is a fall of error each time it is effectively performed. This is due to the diminishing of the local grid size that renders the linear interpolations on each triangle more accurate. After this, there is a slow exponential increase of errors, which falls again the next time Delaunay regridding is used. Note that regridding is not needed very often. The error at $t = 50$ is approximately only two times the initial one.

Figures 10 and 11 also show the behaviour of errors in velocity and vorticity corresponding to TC1 and TC2, respectively, with $h = 1/12$, during the time interval [0, 100]. This is a very long time interval for both cases, as at time $t = 100$ the unit circle has been dramatically deformed by both flows. The curves present sharp oscillations due not only to the use of Delaunay regridding, but also to the low smoothness of the discrete l^∞-norm used. However, we remark at first that in both cases the errors in velocity and vorticity remain almost constant. Also, in both cases the errors in velocity are substantially smaller than those in vorticity. Again, this is very probably due to the higher smoothness of the velocity fields. Note also that although in TC2 the vorticity is not smooth enough to ensure the convergence of Algorithm A′, in practice second-order convergence is attempted. A possible reason for this fact is that the singularities of the vorticity lie on the curve $r = 1$, while we solve Euler equations only inside the unit circle. Also, the fact that the errors corresponding to TC2 are close to those corresponding to TC1 is probably due to the radial distribution of the nodes in the triangulation used.

Figure 12 represents the triangulation for TC1 at time $t = 99$, the last time Delaunay regridding is used. Observe the good quality of the grid, which suggests that our run could continue for longer times with similar error levels.

Globally, these tests show that Algorithm A′ with the use of Delaunay regridding is stable and accurate, with second-order accuracy, even for very long integration times.

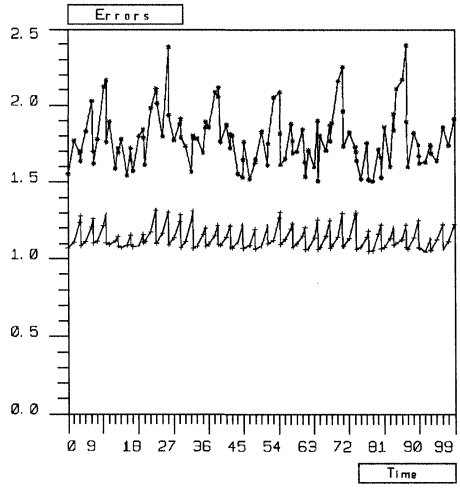

FIG. 10. *Time evolution of errors in velocity (line marked by + symbols) and vorticity (line marked by * symbols) for Algorithm* A′, *applied to* TC1 *with* $h = 1/12$, *in the time interval* [0, 100]. *Both errors remain close to the initial values for the whole time interval.*

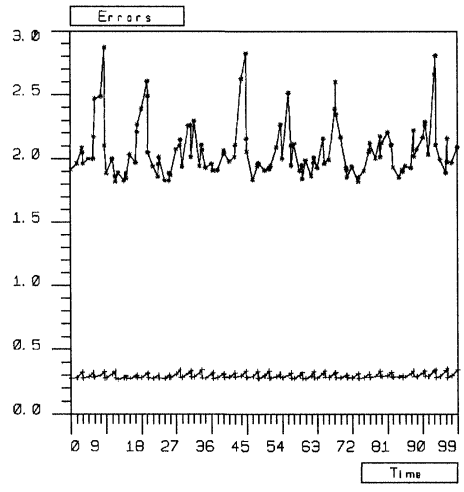

FIG. 11. *Time evolution of errors in velocity (line marked by + symbols) and vorticity (line marked by * symbols) for Algorithm* A′, *applied to* TC2 *with* $h = 1/12$, *in the time interval* [0, 100]. *Both errors remain close to the initial values for the whole time interval.*

5.4. Comparison to a desingularized vortex method. Our next experiment is to compare the performances of a vortex method on a fixed uniform grid with those of Algorithm A′ using Delaunay regridding.

As the vortex method we have used the desingularized point vortex method (DPVM) introduced in Hou [15]. To describe it, let us consider a uniform grid of size h of $\boldsymbol{R}^2$ with nodes $\{\beta_j\}_{j\in N}$. Denote $\omega_j = \omega_0(\beta_j)$. Then, DPVM computes the discrete velocity at point β_k and time t by

$$\hat{u}_h(\beta_k, t) = \sum_{\beta_l \in \operatorname{supp}\omega_0} K(\beta_k - \beta_l)\,(\omega_l - \omega_k) - \omega_k \int_{\Omega_h(t)} K(\beta_k - y)\,dy, \tag{43}$$

where $\Omega_h(t)$ is a polygonal approximation of $\operatorname{supp}\omega(\cdot, t)$.

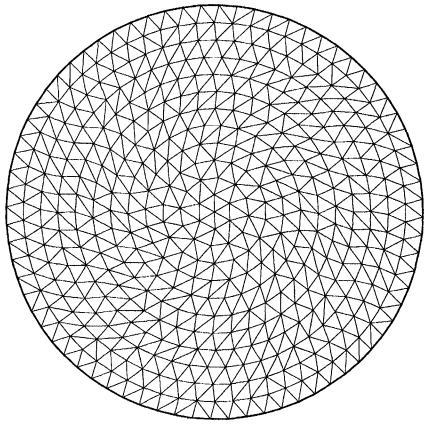

FIG. 12. *Triangulation at time* $t = 99$ *corresponding to Algorithm* A′, *applied to* TC2 *with* $h = 1/12$, *in the time interval* [0, 100].

This is a stable modification of the point vortex method, uniformly convergent with second-order accuracy. Because of that, it seems to be a good method to compare with ours.

In our experiments we have run both algorithms for grids of size $h = 1/12$ and $h = 1/16$. We always take the unit circle to be the set $\Omega_h(t)$. This allows us to compute exactly the integral expression in (43). Also, we solved the equation of characteristics for the DPVM with the Adams–Bashforth second-order scheme, just as in Algorithm A′.

In our tests, if no regridding techniques are introduced, the DPVM produces a large increase of errors in a relative time interval. For instance, for TC1 the relative errors in velocity take values of approximately 80% by time $t = 40$. As we pointed out in the Introduction, some convenient regridding technique is needed to obtain accurate solutions for long integration times.

Beale and Majda introduced in 1985 a simple, but efficient, regridding technique in the context of a vortex–blob method (VBM). VBMs are based upon the discretization of vorticity as a sum of smooth functions with small supports, called *blobs*. Given a smooth cutoff function Ψ (i.e., an approximation of the Dirac delta at the origin), the vorticity at a fixed time t is approximated by

$$\omega(x,t) \simeq \omega_h(x,t) = \sum_j \Psi_\delta(x - X_j(t))\,\omega_j\,h^2, \tag{44}$$

where

$$\Psi_\delta(x) = \frac{1}{\delta^2}\,\Psi\left(\frac{x}{\delta}\right).$$

With a discretization of the kind of (44), it is possible to compute the vorticity at any prescribed point. The regridding technique of Beale and Majda consists of reinterpolating the vorticity at the nodes of a uniform grid, whenever the current local grid size is large enough. However, as reported by Beale and Majda, the grid size of the reinterpolating grid should decrease progressively to maintain reasonable error levels.

In DPVM the vorticity is discretized as a sum of Dirac masses:

$$\omega(x,t) \simeq \omega_h(x,t) = \sum_j \delta(x - X_j(t))\,\omega_j\,h^2. \tag{45}$$

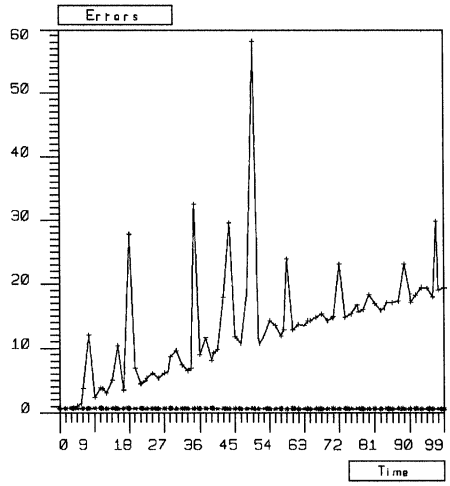

FIG. 13. *Comparison of errors in velocity between the* DPVM *(line marked by + symbols) and Algorithm* A′ *(line marked by * symbols), applied to* TC1, *in the time interval* [0, 100]. *A progressive increase is observed in the first one, while the second remains almost constant.*

Consequently, the regridding technique of Beale and Majda cannot be directly applied here. However, it is possible to use this technique after approximating $\hat{\omega}_h^n$ by a sum of blobs as in (44). In practice, we have used a fourth-order cutoff function Ψ_δ:

$$\Psi_\delta(x) = \frac{1}{\pi\,\delta^2}\left[2\exp\left(-\frac{r^2}{\delta^2}\right) - \frac{1}{2}\exp\left(-\frac{r^2}{2\,\delta^2}\right)\right].$$

This cutoff function was also introduced by Beale and Majda in 1985. For smooth functions f, the error $f - \Psi_\delta \star f$ is of order δ^4. We have taken δ of order h, so this accuracy seems to be enough, as the DPVM is of order h^2.

The regridding strategy that we have used consists of reinterpolating the vorticity when the smallest angle of the deformed grid is smaller than a preset limit value. Each regridding has been set to produce an increase in the amount of the grid points of approximately 15%.

Figures 13 and 14 compare the behaviour of errors due to the DPVM and to the finite element vortex method (FEVM) of Algorithm A′. We represent the errors in velocity and trajectories corresponding to TC1 during the time interval [0, 100] with initial grid size $h = 1/16$.

Sharp variations of errors corresponding to the DPVM are observed, probably due to the additional error introduced in the reinterpolation associated to the regridding steps. The errors corresponding to the DPVM increase faster than those corresponding to the FEVM. By time $t = 0$, the errors corresponding to both methods take very similar values. By time $t = 100$, the later errors are nearly 200 times smaller than the former ones in velocity and nearly 20 times smaller in trajectories. This different growth rate is probably a consequence of the introduction of numerical diffusion in the regridding steps.

We may conclude that the FEVM solves more accurately our TC1 for long times, without introducing numerical diffusion. We should point out that the computational work needed by one time-step with the FEVM is nearly 100 times bigger than the one needed by one time-step with the DPVM. However, this work remains constant in time for the FEVM, while that

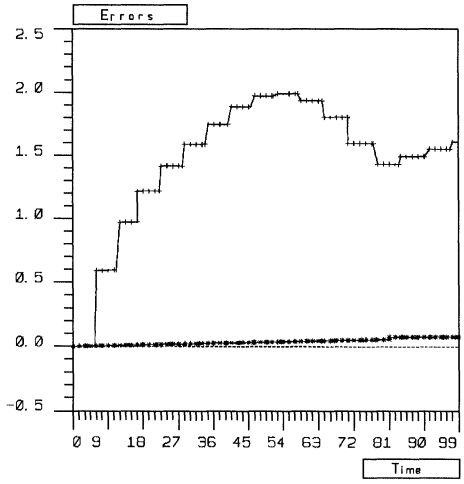

FIG. 14. *Comparison of errors in trajectories of grid points between the* DPVM *(line marked by + symbols) and Algorithm* A′ *(line marked by * symbols), applied to* TC1, *in the time interval* [0, 100].

needed by the DPVM increases each time regridding is performed. Thus, for long enough time intervals, both computational efforts will be of the same order.

5.5. Comparison to high-order VBMs. We finally compared the FEVM with the VBM with cutoff functions of orders $m = 4$ and $m = 6$. Specifically, we used those introduced by Beale and Majda, corresponding to

$p = 4$,

$$\Psi_\delta^{(4)}(x) = \frac{1}{\pi\,\delta^2}\left[2\exp\left(-\frac{r^2}{\delta^2}\right) - \frac{1}{2}\exp\left(-\frac{r^2}{2\,\delta^2}\right)\right],$$

$p = 6$,

$$\Psi_\delta^{(6)}(x) = \frac{1}{\pi\,\delta^2}\left[\frac{8}{3}\exp\left(-\frac{r^2}{\delta^2}\right) - \exp\left(-\frac{r^2}{2\,\delta^2}\right) + \frac{1}{12}\exp\left(-\frac{r^2}{4\,\delta^2}\right)\right].$$

To ensure the convergence of the method, we took the blob size to be $\delta = h^q$, with $0 < q < 1$. Thus, for smooth enough initial vorticity, the convergence order of the method is $p = mq$.

In practice, we took $q = 0.95$ in all our experiments, as this value seems to be quasioptimal, as reported by Perlman. The streamline equation has been solved with a fourth-order Runge–Kutta method with very small time-step. We have tested our code for TC1 at $t = 1$. Our estimations of computed convergence orders are given in Table 1. They are in very good agreement with those reported by Perlman.

We have used the regridding technique described in the preceding subsection, with some minor modifications. Indeed, many possible criteria which can be used to apply regridding are equivalent in practice for the rotating steady solutions we are considering. Either regridding when the smallest angle of the mesh is smaller than a given tolerance, or when the current grid size is long enough, is equivalent to regridding a certain fixed number of time-steps. In any

TABLE 1
Convergence order of the VBM *for* TC1, *estimated at time* $t = 1$, *used to compare to the* FEVM.

Values of m and h	$h = 0.2$	$h = 0.1$	$h = 0.05$	Theoretical orders
$m = 4$	2.53	3.32	3.60	3.80
$m = 6$	2.78	4.44	5.16	5.70

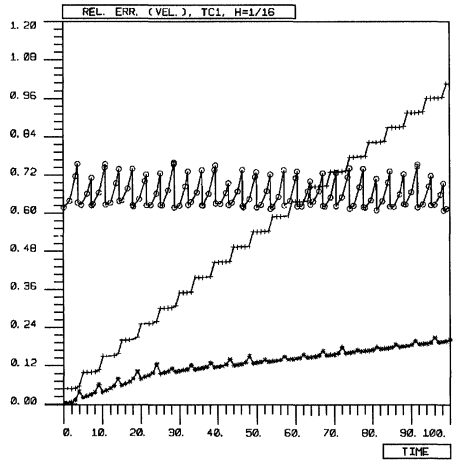

FIG. 15. *Comparison of errors in velocity between Algorithm* A′ *(line marked by 0 symbols) and the* VBM *with* $m = 4$ *(line marked by* + *symbols) and* $m = 6$ *(line marked by* * *symbols) for* TC1 *and* $h = 1/16$ *in the time interval* [0, 100].

case, the actual tolerance value must be tuned with care to avoid an excessive increase in errors. If regridding is applied too often, we shall progressively introduce high levels of numerical diffusion, but if the grid is excessively distorted when regridding, then the accumulated errors will produce an unrecoverable loss of accuracy.

In Figs. 15 and 16 we represent the relative errors in velocity for the FEVM and for the VBM with $m = 4$ and $m = 6$, corresponding to TC1 with $h = 1/16$ and TC2 with $h = 1/12$. We may observe that regridding is applied in all cases an almost constant number of time-steps. In all cases errors are kept almost constant for a short time interval whenever regridding is applied and experience a fast increase when the grid becomes progressively distorted. For $m = 6$, this increase is very fast, and this is probably the reason why regridding produces a decrease in errors. For $m = 4$, the errors do not grow as fast, and regridding produces an increase in them. Also, almost linear growth rates of errors are observed in all cases. These rates are smaller for $m = 6$ than for $m = 4$. For the FEVM, the growth of errors as the grid is distorted is the fastest of all cases considered. Thus, the loss of quality of the grid more dramatically affects the accuracy of the FEVM than that of the VBM. However, applying Delaunay regridding in the FEVM diminishes the errors to values close to the initial ones, counterbalancing almost completely the former increase.

In TC1, which corresponds to a smooth solution, the FEVM presents a better performance at time $t = 100$ than the VBM with $m = 4$, while the VBM with $m = 6$ yields a higher accuracy than the FEVM. However, in TC2, which corresponds to a less smooth vorticity, the performance of the FEVM at $t = 100$ improves that of the VBM with $m = 4$ and also that of the VBM with $m = 6$. We should also remark that the growth rates of errors for the BVM are

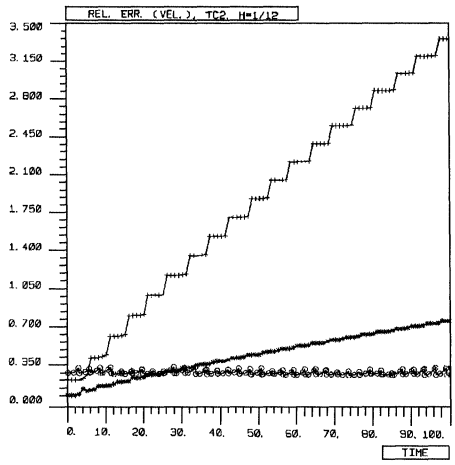

FIG. 16. *Comparison of errors in velocity between Algorithm* A′ *(line marked by 0 symbols) and the* VBM *(line marked by + symbols) with* $m = 4$ *and* $m = 6$ *(line marked by * symbols) for* TC2 *and* $h = 1/12$ *in the time interval* [0, 100].

in all cases larger than those corresponding to the FEVM. Thus, the comparison is very likely to be even more favourable for the FEVM for later integration times.

Finally, we must say that we may not expect to solve two-dimensional Euler equations with any initial condition, simply by using Algorithm A′ combined with Delaunay regridding. It seems clear that, in general, the other regridding rules that we mentioned in §2 are needed to obtain accurate results. The results presented in this paper must be understood in the sense that our FEVM, due to its geometrical adaptability, improves the accuracy of classical vortex methods, without introducing numerical diffusion.

Appendix: Proof of Theorem 4.1.

Proof. Our proof is an adaptation of the convergence proof for Algorithm A given by Chacon and Hou. The essentials of the proof are as follows. Let us define

$$T^* = \max\left\{t_n : 0 \le t_n \le T,\ \max_{0\le k\le n} \|X^k - \hat{X}^k\|_{\infty,h} \le h^{1+\rho}\right\}, \quad \text{where } \rho = \frac{1}{2}[\min(2,\sigma)-1].$$

We prove that there exists a separation parameter d_T, depending only on ω_0, T, c, and $\bar{\gamma}$, such that estimates (33) hold in the time interval $[0, T^*]$. Then, we conclude that there exist two positive numbers h_T and Δ_T such that if $0 < h < h_T$ and $0 < \Delta t < \Delta_T$, then there must be $T^* \ge T$.

The main innovation in our analysis is that we obtain uniform-in-time estimates for the error in the computation of discrete velocities by Algorithm B. As we shall see, this essentially happens because our algorithm preserves the uniform norm of the discrete solution.

Indeed, if $0 \le t_n \le T^*$, following Chacon and Hou we state that if h is small enough, then all triangulations $\{\hat{T}_h^n\}_{0\le t_n\le T^*}$ are nondegenerated. Moreover, all aspect ratios of these triangulations are uniformly bounded from above by a constant γ_T independent of h. Let us define the separation parameter

$$d_T = \lambda_T\,(1 + c),$$

where λ_T is the parameter associated to γ_T given by Theorem 3.11. Note that d_T depends only on ω_0, T, c, and the aspect ratio of the initial triangulations.

We prove now that there exist two positive constants r and C such that

$$\operatorname{supp} \hat{\omega}_h^n \subset B(0, r), \quad 0 \le t_n \le T^*, \tag{46}$$

and

$$\max_{0 \le j \le M} |\left[\hat{u}_h^n - (K \star \hat{\omega}_h^n)\right](\hat{X}_j^n)| \le C\, r\, h^\sigma, \quad 0 \le t_n \le T^*. \tag{47}$$

Indeed, assume

$$\operatorname{supp} \hat{w}_h^{n-1} \subset B(0, r_{n-1}), \quad \operatorname{supp} \hat{\omega}_h^n \subset B(0, r_n)$$

for some positive numbers $r_{n-1} \le r_n$.

Let us denote by $|\cdot|$ the Euclidean norm on $\boldsymbol{R}^2$ and by $\|\cdot\|_\infty$ the uniform norm on $\boldsymbol{R}^2$. Theorem 3.11 yields

$$\begin{aligned} |\hat{u}_h^n(\hat{X}_j^n)| &\le |\left[\hat{u}_h^n - (K \star \hat{\omega}_h^n)\right](\hat{X}_j^n)| + |(K \star \hat{\omega}_h^n)(\hat{X}_j^n)| \\ &\le C_1\, r_n\, \|\hat{\omega}_h^n\|_\infty\, h^\sigma + \int_{B(0,r_n)} |K(\hat{X}_j^n - y)|\, \hat{\omega}_h^n(y) dy \le C_2\, r_n\, \|\omega_0\|_\infty. \end{aligned} \tag{48}$$

Note that the last inequality here follows because Algorithm A′ conserves in time the uniform norm of the discrete vorticity. Consequently,

$$|\hat{X}_j^{n+1}| \le |\hat{X}_j^n| + \frac{\Delta t}{2}\left[3\,|\hat{u}_h^n(\hat{X}_j^n)| + |\hat{u}^{n-1}(\hat{X}_j^{n-1})|\right] \le |\hat{X}_j^n| + C_3\, \Delta t, \quad j = 1, \ldots, M.$$

Thus,

$$r_n \le r = r_0\, \exp(C_3\, T), \quad 0 \le t_n \le T^*.$$

The remainder of the proof is a technical refinement of that of Chacon and Hou. We shall omit it here, as it does not introduce any essential innovation. □

Acknowledgments. The authors wish to thank Macarena Gómez Marmol for her valuable help in obtaining the graphic output.

REFERENCES

[1] C. Anderson, *Observations on vorticity creation boundary conditions*, in Mathematical Aspects of Vortex Dynamics, R. Caflisch ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1988, pp. 144–159.

[2] C. Anderson and C. Greengard, *On vortex methods*, SIAM J. Numer. Anal., 22 (1985), pp. 413–439.

[3] J. T. Beale and A. Majda, *Vortex methods* I: *Convergence in three dimensions*, Math. Comp., 32 (1982), pp. 1–27.

[4] ———, *Vortex methods* II: *High order accuracy in two and three dimensions*, Math. Comp., 32 (1982), pp. 29–52.

[5] ———, *High order accurate vortex methods with explicit vorticity kernels*, J. Comp. Phys., 58 (1985), pp. 188–208.

[6] M. Bernadou et al., *MODULEF A Modular Library of Finite Elements*. INRIA, Rocquencourt, France, 1986.

[7] R. Bowyer, *Computing Dirichlet tessellations*, Comput. J., 24 (1981) pp. 162–166.

[8] T. F. Buttke, *Fast vortex methods in three dimensions*, in Vortex Dynamics and Vortex Methods, Lectures in Appl. Math., Vol. 28, K. E. Gustafsson and J. A. Sethian, eds., American Mathematical Society, Providence, RI, 1991, pp. 51–66.

[9] ———, *A fast adaptive method for patches of constant vorticity in two dimensions*, J. Comput. Phys., 89 (1990) p. 161.

[10] T. CHACÓN REBOLLO AND T. Y. HOU, *A Lagrangian algorithm for the 2D Euler's equations*, Comm. Pure Appl. Math., 43, (1990), pp. 785–767.
[11] P. CIARLET, *The Finite Element Method for Elliptic Problems*, North-Holland, Amsterdam, 1978.
[12] L. GREENGARD AND V. ROKHLIN, *A fast algorithm for particle simulation*, J. Comput. Phys., 73 (1987), p. 325.
[13] L. GREENGARD, *Rapid evaluation of potential fields in particle systems*, in ACM Distinguished Dissertations, MIT Press, Cambridge, London, 1988.
[14] T. Y. HOU, *A survey of convergence analysis for vortex methods*, in Vortex Dynamics and Vortex Methods, Lectures in Appl. Math., Vol. 28, K. E. Gustafsson and J. A. Sethian, eds., American Mathematical Society, Providence, RI, 1991, pp. 327–340.
[15] ———, *A new desingularization for vortex methods*, Math. Comp., 58 (1992), pp. 103–117.
[16] T. Y. HOU AND J. LOWENGRUB, *The convergence of the point-vortex method for the 3-D Euler equations*, SIAM J. Numer. Anal., 27 (1990), pp. 1387–1404.
[17] T. KATO, *On the classical solution of the two-dimensional nonstationary Euler equations*, Arch. Rational Mech. Anal., 25 (1967), pp. 188–200.
[18] A. LEONARD, *Computing three-dimensional incompressible flows with vortex elements*, Ann. Rev. Fluid Mech., 17 (1985), pp. 523–559.
[19] A. MAJDA, *Vortex Dynamics: Numerical analysis, scientific computing and mathematical analysis*, in Proceedings 1st ICIAM, Paris, 1987.
[20] M. PERLMAN, *On the accuracy of vortex methods*, J. Comput. Phys., 59 (1985), pp. 200–223.