JOURNAL OF COMPUTATIONAL PHYSICS **145,** 61–88 (1998)
ARTICLE NO. CP986030

Adaptive Triangulation of Evolving, Closed, or Open Surfaces by the Advancing-Front Method

S. Kwak and C. Pozrikidis

Department of Applied Mechanics and Engineering Sciences, University of California, San Diego, La Jolla, California 92093-0411
E-mail: costas@ames.ucsd.edu

Received January 20, 1998; revised April 28, 1998

The advancing-front method is adapted to describe the evolution of open or closed three-dimensional surfaces in terms of an unstructured grid consisting of quadratic triangular elements. In science and engineering applications, the surface may be identified with a material interface, a free boundary, or a moving front. In the numerical method, the geometrical properties of the surface and the coordinates of the triangle vertices are computed either in terms of available analytical expressions, or by means of interpolation through an underlying coarse grid. The shape and size of the curved triangular elements are determined by the maximum magnitude of the mean or directional local surface curvature. Two algorithms are implemented: the first one for simply connected open surfaces bounded by closed lines, and for closed surface with plane symmetry; and the second for closed surfaces. In the case of open surfaces, the discretization front advances from a one-dimensional boundary grid that traces the bounding curve. The boundary grid is generated either by a one-dimensional version of the advancing-front method, or by requiring a set of criteria based on local line representation with circular arcs. In the case of closed surfaces, the discretization front emanates from the point of the maximum mean curvature. In both cases, the algorithm also interpolates to generate the values of surface geometrical or physical variables such as temperature or concentration of a surface-active agent. The method was tested by following the motion of several passive and active surfaces evolving under the action of specified fields of flow, while performing occasional regridding to ensure adequate spatial resolution. In one test, the large deformation of a viscous drop subjected to an infinite simple shear flow at vanishing Reynolds number was computed into the regime where a cigar-like shape is established, thereby extending previous numerical computations for small and moderate deformations and reproducing experimentally observed shapes. Overall, the adaptive-front method emerges as an important tool in numerical studies of free boundaries or moving fronts and should be useful in a broad range of applications.

0021-9991/98 $25.00

1. INTRODUCTION

The use of unstructured grids based on triangulations to describe the shapes of two-dimensional contours and three-dimensional iso-scalar surfaces, material surfaces, and fluid interfaces has been gaining increasing popularity in recent years [1–11]. Compared to a structured grid that is defined in terms of global surface curvilinear coordinates, the unstructured grid has several advantages: The local curvilinear coordinates over each triangle are non-singular, whereas the global curvilinear coordinates may have singular points; the triangle shape and size may be controlled effectively to enhance the spatial resolution at regions of particular interest; and the discretization is amenable to the meritorious finite-volume and spectral-element formulations for solving integral or differential equations over an evolving surface. Examples are the Fredholm integral equations arising from boundary-integral formulations of potential or Stokes flow, and the convection-diffusion differential equation for the transport of an insoluble surfactant.

The need for adaptive surface triangulation becomes evident by inspecting Fig. 1, where we present the shape of a file of liquid drops moving though a circular tube under the action of a pressure-driven Poiseuille flow [3]. At the initial instant, each drop has a spherical shape that is readily described by elementary triangulation based on the subdivision of an octahedron. But as a drop starts deforming, a dimple develops at the back, and the interface is no longer represented with adequate resolution. In this simulation, the marker points move with the component of the velocity of the fluid normal to the interface and with a certain tangential velocity. There is a wealth of other applications where a reliable description of evolving open or closed surfaces and the accurate computation of their geometrical properties, including the mean curvature, is imperative. Examples include the flow suspensions of deformable capsules such as red blood cells, the evolution of three-dimensional vortex sheets, and the stretching and folding of interfaces in laminar or turbulent flows.

To prevent grid distortion, the tangential velocity of the marker points may be adjusted according to certain criteria involving, for example, the rate of change of the length of the edges of the triangle or the angles subtended between them [3, 7, 8]. One difficulty with this approach is that a satisfactory set of criteria involving the triangle size, skewness,

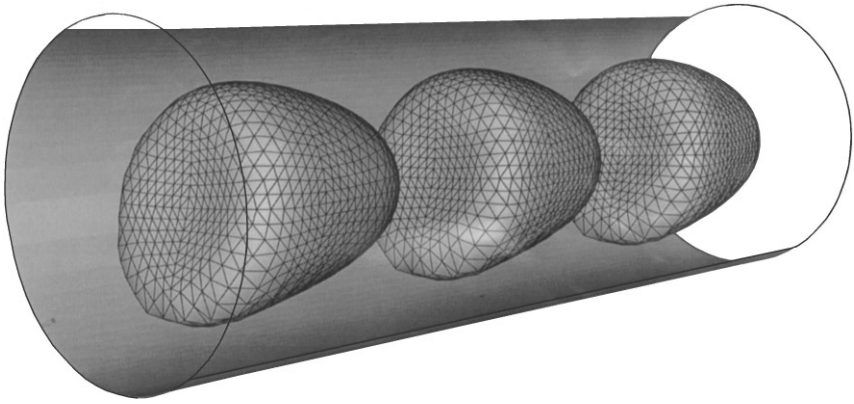


FIG. 1. Triangulation of the interfaces of liquid drops traveling through an ambient fluid in pressure-driven tube flow [3].

and the surface curvatures is difficult to devise. Even if this were possible, the associated minimization or optimization problem, whose solution produces the tangential velocity of the marker points, places a heavy burden on the numerical method [8]. Nakahashi and Deiwert [12] developed a method for the optimal distribution of the grid points defining a global structured grid based on variational principles and a grid-spring analogy. More recently, Cristini *et al*. [9] developed a procedure for regridding a three-dimensional surface based on the idea of allowing the triangles to relax to an equilibrium configuration. The marker points are connected by, and move under the influence of, damped massless springs; optimal marker point connectivity is maintained by local reconnection; and a specified local density of triangles is maintained by adding and subtracting marker points at regions where the elastic tensions are large.

In this work, we develop a procedure for the dynamical regridding of an evolving three-dimensional open or closed surface based on the advancing-front method. Originally, the advancing-front method was developed, as a part of a finite-element procedure, for triangulating a region in a plane [13, 14]. In recent years, the method was extended to handle three dimensional stationary surfaces with applications in aerodynamics [15–18]. The highlights of the method in three dimensions are: adaptive triangulation according to a measure of the local surface curvature; reasonably uniform distribution of triangle sizes; effective control of triangle skewness; and reduced user intervention. To the authors' knowledge, the method has not been implemented to handle the changing shapes of evolving surfaces by means of regridding, although a step in that direction was recently made. Löhner [18] developed an algorithm for grid refinement that incorporated interpolation of geometrical or other surface variables from a crude to a refined grid.

In the first part of this work, we discuss an implementation of the advancing-front method for a grid consisting of curved *quadratic* triangular elements; previous work considered planar *linear* triangular elements. The interpolation of geometrical and other surface variables from the coarse to the refined grid is somewhat similar to that developed by Löhner [18], but there are several differences. We present two general algorithms applicable to two distinct classes of problems involving: (a) open surfaces bounded by single closed lines, as depicted in Fig. 2a; or (b) closed surfaces whose exterior is a singly or multiply connected domain, as depicted in Fig. 2b. The algorithm for open surfaces is also applicable to closed surfaces that are symmetric with respect to a plane. In that case, duplication by plane reflection produces the whole of the surface. The procedure for open surfaces incorporates an algorithm for the discretization of a closed curve, which is done in two independent ways. The two methods are fit as stand-alone routines in problems involving the motion of lines in two or three dimensions. Analogous algorithms for the optimal distribution points along closed or periodic planar lines are discussed and reviewed in Refs. [19, 20].

In the second part of this work, we test and confirm the suitability of the advancing-front method for describing surfaces and fluid interfaces evolving under the action of a specified field of flow. In test simulations, we consider passive interfaces advected by an imposed flow, and active interfaces whose shape affects the development of the flow. In all cases, the grid points move with the velocity of the fluid, and occasional regridding is performed to ensure adequate resolution. In the most demanding tests, we combine the regridding method with a boundary element method for Stokes flow to describe the large deformation of a liquid drop immersed in simple shear flow, and illustrate the development of slender shapes.

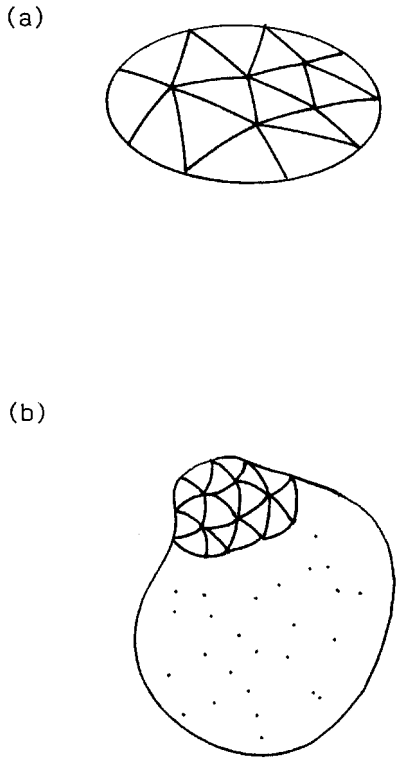


FIG. 2. (a) Illustration of an open surface bounded by a closed curve and represented by a collection of curved triangles. (b) A closed surface represented by a collection of curved triangles.

2. ADVANCING-FRONT METHOD FOR OPEN SURFACES

Consider a three-dimensional line C defined in terms of an arbitrary distribution of N_c marker points bounding an open surface, as shown in Fig. 3a. The line segments between two successive points are edges of curved triangular elements that describe the surface. As a first step toward implementing the advancing-front method, we redistribute the contour marker points with two objectives: Achieve a reasonably uniform distribution of edge lengths so that adjacent triangles have comparable sizes; and resolve regions of high curvature.

We implemented and tested two independent contour-point redistribution algorithms, henceforth called bounding curve discretization algorithms, as will be described in the following two subsections. Best results were obtained when the algorithms were applied repeatedly and in alternating fashion a number of times, but each algorithm alone produced acceptable distributions, with the first algorithm being the best performer. In the second algorithm, the geometry of the surface does not play a role in the discretization of the contour; that is, the method is oblivious to the curve being the boundary of a surface.

2.1. *Contour Discretization by the AFM*

In this algorithm, we employ a variation of the one-dimensional version of the advancing-front method developed by Nakahashi and Sharov [16]. The main idea is to proceed from the point of highest contour curvature toward regions of lower curvature, while continuously monitoring the magnitude of the curvature and the ratio of arc lengths of successively

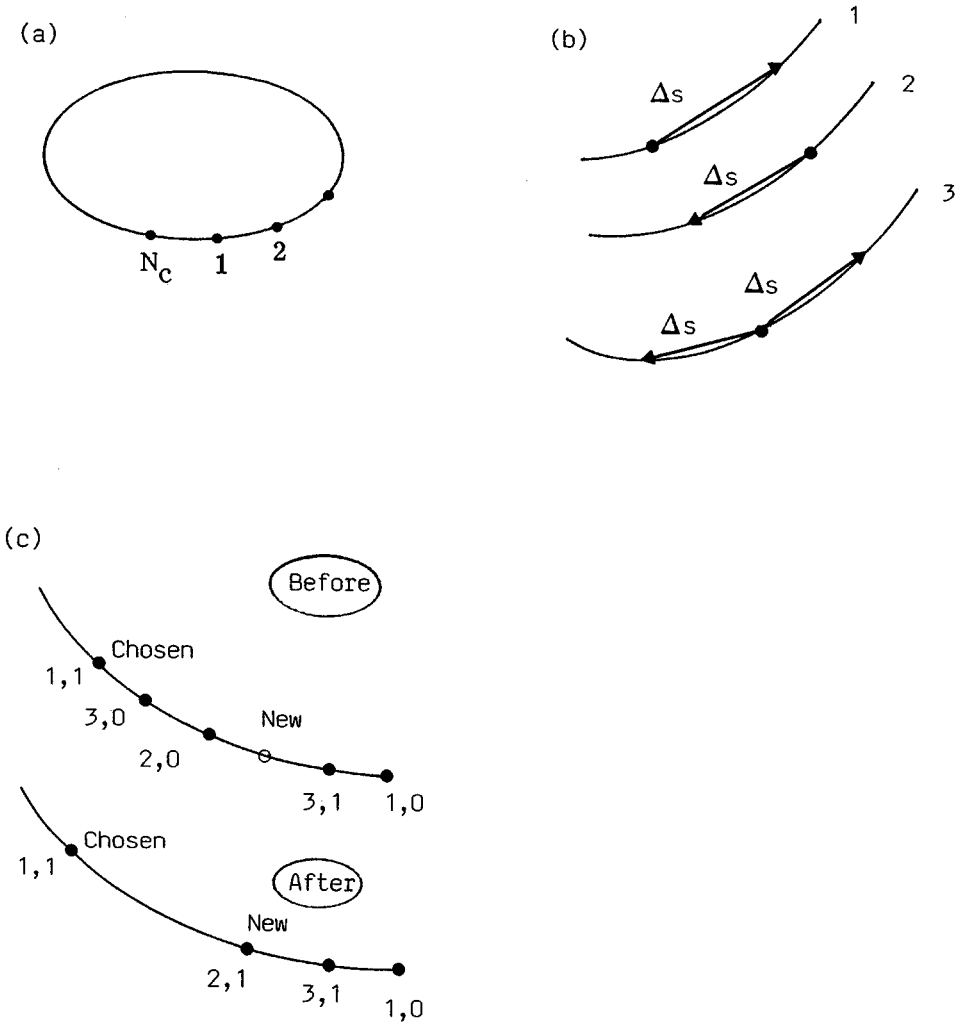


FIG. 3. (a) A closed line is represented by N_c marker points. (b) Introduction of one or two new nodes along a line; the labels 1, 2, 3 correspond to the discretization direction index of the point marked with a filled circle. (c) Introduction of a new node on a curve with pre-distributed marker points, before and after; the pair of integers next to a point state the discretization direction index and the usage index.

generated segments. A large number of line segments are generated at regions of high curvature, and fewer segments are generated at regions of lower curvature. The steps are as follows:

(1) Number the N_c successive bounding curve nodes, moving in a desirable direction according to the designated side of the triangulated surface as the *upper* side, as shown in Fig. 3a. The point 1 coincides with the point $N_c + 1$.

(2) Describe the Cartesian coordinates of the contour in a parametric manner with respect to the reduced polygonal arc length ξ, as $\boldsymbol{x}(\xi)$; $\xi = l_P / l_{P,Tot}$ where l_P is the length of the polygonal line connecting successive marker points starting from the first point, and $l_{P,Tot}$ is the perimeter of the N_c-sided three-dimensional polygon.

In our implementation, we carried out the interpolation in terms of periodic cubic splines and found that, in some instances, the cubic spline method without smoothing causes

significant fluctuations in the computed curvature, thus undermining the effectiveness of the method. An alternative approach is to abandon the global representation and use instead a local representation based on circular arcs passing through trios of adjacent marker points [21].

(3) Assign to each node of the bounding curve the value of either the local curvature of the contour κ_c, or of the maximum local directional curvature of the surface that is bounded by the contour, κ_{Max}. The first choice is appropriate when the curve is a stand-alone line; the second choice is appropriate when the curve is the boundary of an open surface. The objective in both cases is to introduce a proper length scale that can be used to control the node spacing.

The local curvature of the contour κ_c, can be computed from the formula $\kappa_c h^2 \boldsymbol{n} = -\partial^2 \boldsymbol{x}/\partial \xi^2$, where $\boldsymbol{n}$ is the unit vector normal to the line, and h is the line metric, $h = |\partial \boldsymbol{x}/\partial \xi|$. A simpler method identifies κ_c with the curvature of the circular arc that passes through the node of interest and two adjacent nodes on either side, as described in Ref. [21]. The computation of the maximum local directional curvature of the surface, κ_{Max}, will be discussed in Section 3.

(4) Label each node with a *discretization direction index* that takes the value of 1 to indicate discretization only toward increasing node numbers, 2 to indicate discretization only toward decreasing node numbers, or 3 to indicate discretization in both directions. The use of this index prevents the discretization from exceeding the designated beginning and end of the line in parametric space, and eliminates the risk of accidentally deleting nodes at the step (6)(g) to be discussed shortly. Initially, all nodes are labeled 3, except for node 1 that is labeled 1, and node $N_c + 1$ that is labeled 2.

(5) Label each node with a *usage index* that takes the value of 1 to indicate that the node has been chosen previously as a starting point for the discretization, and 0 otherwise. Initially, all nodes are labeled with 0.

(6) The core of the algorithm is the curve discretization stage involving several substeps, as follows. Steps (6)(b), (e) attempt to control the arc lengths of the line segments and produce distributions with smooth variations.

(a) Among the nodes whose usage index is equal to 0, choose the one with the highest magnitude of the curvature. This node will be called the *chosen* node.

(b) Compute the arc length $\Delta s = 2\sin(\alpha/2)/|\kappa_c|$ or $\Delta s = 2\sin(\alpha/2)/|\kappa_{Max}|$, where κ_c and κ_{Max} are, respectively, the curvature of the contour and the maximum curvature of the surface at the chosen node, and α is a specified angle that is a free parameter of the numerical method. If the chosen node is node 1 or $N_c + 1$, we skip steps (6)(c)–(e) and proceed to step (6)(f). Otherwise, we reduce or amplify Δs to bring it within a specified window $(\Delta s_{min}, \Delta s_{max})$.

(c) If the discretization direction index of the chosen node has the value of 1 or 3, search the points on the left until the discretization direction label changes to 2. Let the number of the point with the discretization direction label 2 be p. If the discretization direction label of the chosen node is 2, perform a search on the right until the label changes to 1. Let the number of the point with the discretization direction label 1 be q. In some cases only one of p or q may be found, but this does not present a difficulty.

(d) Calculate the distance Δs_1 between nodes p and $p+1$ and the distance Δs_2 between nodes q and $q-1$. Compare Δs_1 and Δs_2, choose the one with the smaller magnitude, and call it Δs_{Ref}. Accommodations must be made for cases where one of p or q has not been found.

(e) If $1/\phi < \Delta s/\Delta s_{Ref} < \phi$, where ϕ is a specified constant, keep Δs. Otherwise replace it with the value of Δs_{Ref}.

(f) Introduce a *new* node; the chosen node and the new node are the end-points of a *new* segment with approximate arc length equal to Δs, placed on the appropriate side of the chosen node according to the discretization index of the chosen node, as illustrated in Fig. 3(b). If the value of the discretization index is equal to 3, then the segment is placed on both sides. The coordinates of the new node are found by interpolation from the cubic-spline representation.

(g) Examine whether one or more pre-existing nodes exist between the *chosen* and the *new* mode. If the discretization direction index of the pre-existing node is equal to 1 or 2, replace the new node with the pre-existing node that lies closest to the chosen node. If, on the other hand, the discretization direction index of the pre-existing node is equal to 3, remove this node from all lists once and for all. An example is illustrated in Fig. 3c.

(h) Update the node list, the discretization-direction list, and the usage list. A new node located on the left or on the right of the chosen node receives, respectively, the discretization-direction index 1 or 2. If a pre-existing node has replaced the new node in step (6)(g), then it receives the usage label 1; otherwise it receives the usage label 0. The chosen node label is switched to 1; this node will not be used again as a starting point. An example is illustrated in Fig. 3c.

(i) With the node lists updated, reparametrize the curve and recompute the coefficients of the cubic spline.

(j) If all nodes have a usage list index of 1, stop the computation; otherwise return to step (6)(a) and repeat the process.

After the contour discretization has been completed, the ratios of successive line segment arc lengths are re-examined. Steps (6)(b) and (6)(e) impose restrictions on the variations of segment lengths, but the resulting distributions may violate the required criteria because of the replacement of a new node with a pre-existing node in step (6)(g). Pronounced variations become prevalent for contours with complex shapes. If the testing associated with a specified parameter ϕ is not satisfied everywhere on the curve, then the angle α is modified, and steps (1)–(6) are repeated until all length segment ratios fall within the range $(1/\phi, \phi)$. Setting ϕ equal to 1 produces evenly spaced points, where the arc length between two points is proportional to the minimum of the absolute value of the radius of directional curvature. In our implementation, the modification of α was done automatically by specifying factors for increasing or decreasing its value.

The variables and parameters of the method just described are summarized in Table I. The number of contour points placed around the contour is an implicit function of the four numerical parameters α, Δs_{min}, Δs_{max}, and ϕ, with the precise functional form depending on the complexity of the contour shape. Several examples of point distributions are presented in Fig. 4. Figure 4a shows the trace, in the azimuthal plane of left-and-right symmetry, of the interface of a liquid drop moving along a cylindrical tube in pressure-driven flow, computed from a dynamic simulation using the boundary-element method starting from the spherical shape [3]. The upper and lower panels show, respectively, the point distribution before and after the application of the advancing front method with $\alpha = 0.395\pi$, $\phi = 1.4$, $\Delta s_{min} = 2\sin(\alpha/2)/|\kappa_c|$, and $\Delta s_{Max} = 20\Delta s_{Min}$. Two notable features are the dimpled shape of the interface at the rear of the drop, with regions of negative directional curvature, and the conical shape of the front, with regions of large positive curvature.

TABLE I
Variables and Parameters for Contour-Discretization by the Advancing-Front Method

	Variable or parameter	Function
1	Discretization direction Index	Indicates direction of discretization.
2	Usage index	Marks a previously examined node.
3	α	Specified angle to control point spacing according to curvature.
4	$(\Delta s_{min}, \Delta s_{max})$	Specified window of point spacing.
5	ϕ	Ratio of successive segment lengths lies within the window $(1/\phi, \phi)$.

Inspecting the upper panel, we notice that the clustering of the marker points at the rear has produced an unacceptable point distribution, and this was a reason for halting the simulation. The redistributed marker points shown at the lower panel describe the shape of the interface in an economical fashion. In particular, the arc length between successive marker points varies smoothly even though the magnitude of the curvature shows pronounced variations.

Figures 4b and 4c illustrate the performance of the method for three-dimensional lines. Before discretization, the point distribution in Fig. 4b is smooth everywhere except near the two regions of high curvature. The advancing front method accommodates the strong

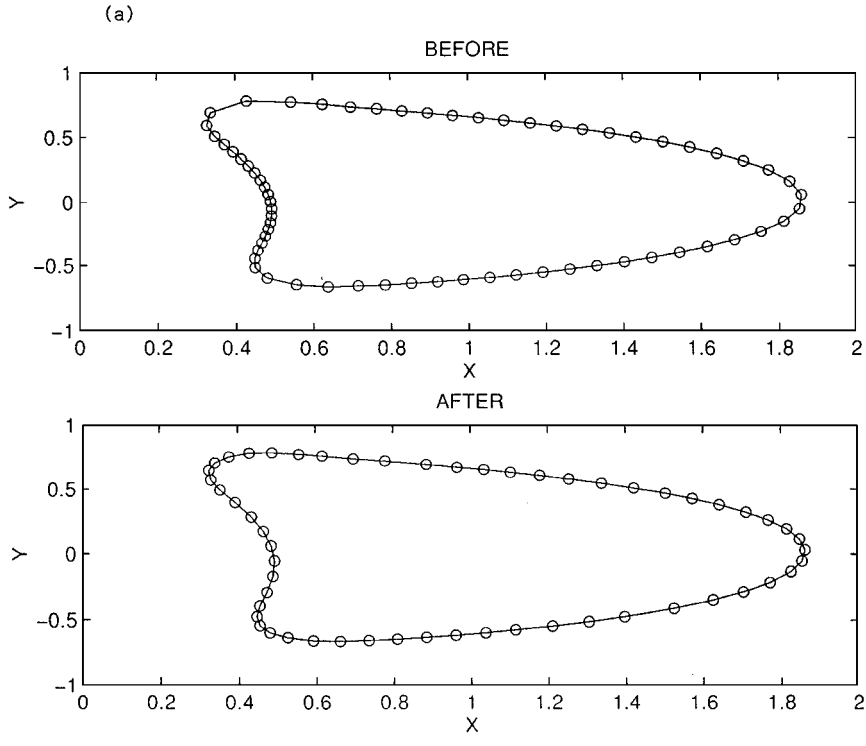


FIG. 4. (a) Illustrations of improved distribution of marker points along the trace of a slipper-shaped drop in the mid-plane [3]. (b), (c) Improved distributions of marker points along three-dimensional curves with large variations in curvatures.

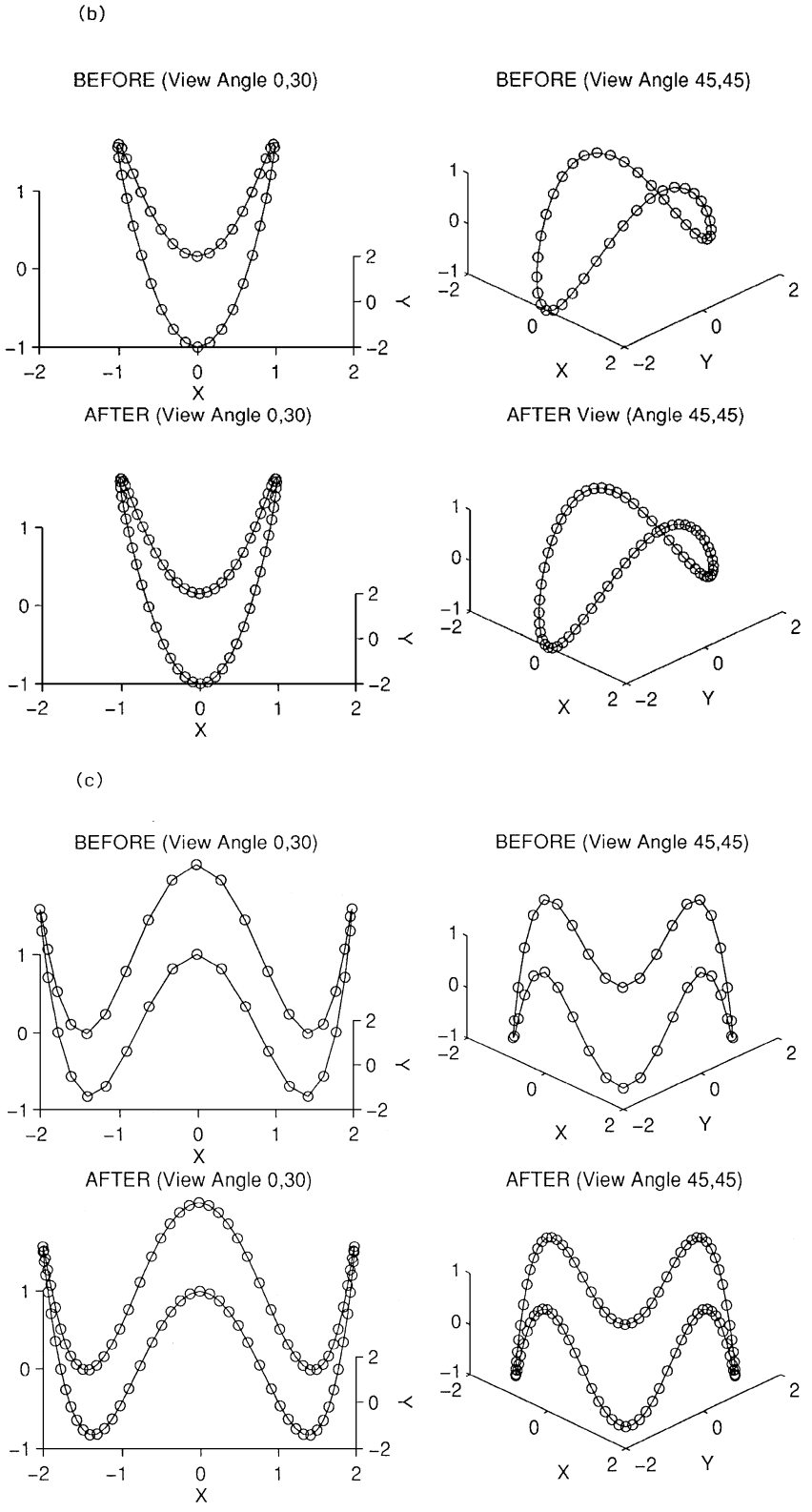


FIG. 4—*Continued*

variation in curvature by allocating a larger number of point relative to the lower curvature region, while maintaining a smooth variation in arc length between adjacent nodes. Figure 4c demonstrates the effectiveness of the method under quite demanding conditions.

2.2. *Contour Discretization by Local Representation with Circular Arcs*

We developed and implemented an alternative contour discretization algorithm based on local interpolation that involves the following three tests:

(1) Compute the circular arc that passes through all trios of successive points using the method discussed in [21]. If the angle subtended by the arc center and the first and third point is higher than a present threshold, then discard the middle point and introduce two new points at even intervals along the arc.

(2) Examine the arc length between two successive nodes. If it is greater than a present threshold, introduce a new node mid-way between the two nodes. The new node is placed on the blended forward-backward arc.

(3) Re-examine the arc length between two successive nodes. If it is less than a present threshold, consolidate the two nodes into one node located on the blended forward-backward arc. Consolidation is allowed only if the resulting point distribution does not violate the first two criteria.

This method is much faster and easier to implement than the advancing-front method described earlier, but may produce imperfect distributions where the ratios of the arc lengths of successive pairs of points are unacceptably large or small, even though regions of high curvature are described with good resolution. If the objective is to simply discretize the line, pronounced arc length variations may be tolerated. But if the contour points are the vertices of triangles defining a surface, large variations will undermine the quality of the triangulation.

2.3. *Open Surface Discretization*

Like the contour advancing-front discretization method described in Subsection 2.1, the surface advancing-front discretization method uses the curvature of the surface to generate flat triangular elements each defined by three nodes, advancing from regions of highest curvature to regions of lower curvature. Ideally, the distance between two vertices of a triangle should be inversely proportional to the magnitude of the directional curvature of the surface in the direction of these two points. After the primary flat triangles have been defined, six-node quadratic triangles that share the nodes of the flat triangles are generated by surface interpolation. The *surface definition data* consist of the coordinates of the marker points defining the curved triangles, and a connectivity list associating the triangle and vertex numbers.

The method proceeds according to the steps outlined below. The geometrical variables necessary in the individual steps are computed either from available analytical expressions, or by interpolation through an underlying coarse grid of quadratic triangular elements using the method discussed in Subsection 2.4.

(1) Describe the contour bordering the open surface with a number of nodes using the methods described in Subsections 2.1 and 2.2, compute the distances between successive marker points, and introduce a front list containing the numbers of the nodes that define the edges and the length of the edges.

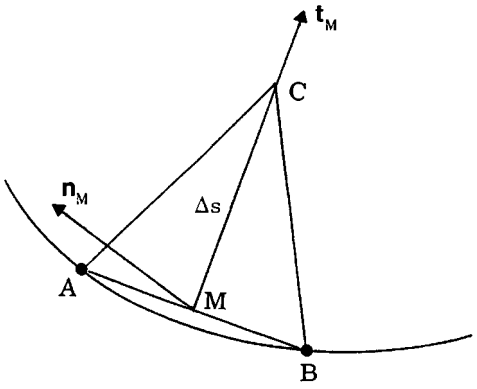


FIG. 5. Introduction of a new marker point C on a surface, appended at a distance Δs from a node in the direction of tangent vector at point M.

(2) Choose the smallest segment on the front list, label the end-points of this segment as point A and point B, and compute the unit vectors normal to the surface at the locations of the end-points, $\boldsymbol{n}_A$ and $\boldsymbol{n}_B$. Introduce the mid-point M on the line segment AB, and approximate the unit vector normal to the surface, $\boldsymbol{n}_M$, and the unit vector tangent to the surface and perpendicular to the segment AB at the point M, $\boldsymbol{t}_M$, with the expressions

$$\boldsymbol{n}_M \cong \frac{1}{2}(\boldsymbol{n}_A + \boldsymbol{n}_B), \qquad \boldsymbol{t}_M \cong \frac{1}{|\boldsymbol{x}_B - \boldsymbol{x}_A|}\boldsymbol{n}_M \times (\boldsymbol{x}_B - \boldsymbol{x}_A) \tag{1}$$

as depicted in Fig. 5.

(3) Compute the maximum magnitude of the normal surface curvature at the location of the point M, denoted as $|\kappa_{max}|$. Let ξ and η be a set of two surface curvilinear coordinates. The normal curvature in the direction of the generally non-unit tangential vector $\boldsymbol{t}(\lambda) = \boldsymbol{t}_\xi + \lambda \boldsymbol{t}_\eta$, where $\boldsymbol{t}_\xi$ and $\boldsymbol{t}_\eta$ are the unit tangential vectors along the ξ and η axes, and λ is a free parameter, is given by [22]

$$\kappa(\lambda) = -\frac{a + 2b\lambda + c\lambda^2}{A + 2B\lambda + C\lambda^2}. \tag{2}$$

The coefficients A, B, and C define the *first fundamental form* of the surface, and the coefficients a, b, and c define the *second fundamental form* of the surface. Both sets of coefficient are computed either from available analytical expressions or from the parametric representation of the curved triangles that define an underlying coarse grid. Differentiating the right-hand side of Eq. (2) with respect to λ, and setting the derivative equal to zero yields a quadratic equation for λ whose solution indicates the directions of the minimum and maximum normal curvature. Once these directions are available, the maximum of the directional curvature follows from a simple computation.

(4) To define the third vertex of the triangle, designated as point C in Fig. 5, we first compute the preliminary triangle height Δs from the expression $\Delta s = 2\sin(\alpha/2)/|\kappa_{Max}|$. The numerical parameter α determines the skewness of the triangle. To prevent the formation of too skewed a triangle, we further require that

$$\frac{1}{\phi}\Delta s_{AB} < \frac{\Delta s}{\delta} < \phi \Delta s_{AB}, \tag{3}$$

where δ is a numerical parameter set equal to $\sqrt{3}/2$. If the computed Δs lies outside this window, then we set it equal to the upper or lower limit.

Next, we introduce the new point C' located a distance Δs away from the point M in the direction of $\boldsymbol{t}_M$, as shown in Fig. 5. Because of the surface curvature, the point C' will not necessarily lie in the surface. To prevent this tangential departure, we apply a modification of the iterative procedure developed by Nakahashi and Sharov [16] to project C' onto the surface. When an underlying coarse grid is available, the procedure involves finding the grid point in the unrefined surface that is closest to the point C', designated by the subscript s, and computing a sequence of points that successively approach the surface, using the recursive formula

$$\boldsymbol{x}_C^{(i+1)} = \boldsymbol{x}_C^{(i)} + \frac{1}{2}\boldsymbol{n}_S\left(\boldsymbol{n}_S \cdot \left(\boldsymbol{x}_S - \boldsymbol{x}_C^{(i)}\right)\right), \tag{4}$$

where $\boldsymbol{n}$ and $\boldsymbol{x}$ are, respectively, the unit normal vector and the position vector. This procedure may fail to project the point C' on the surface at regions of high curvature, especially when the surface definition data are sparse. To prevent this pitfall, we further adjust the position of C' using the interpolation method described in Subsection 2.4. At the final position, we compute the unit vector normal to the surface.

(5) In this stage, we decide whether the point C' will be introduced as a new node, or will be replaced by an existing node. For this purpose, we search for pre-existing nodes within the radius $\gamma\Delta s$ around point C', and form a *local point list*; γ is a numerical parameter whose value was set equal to 0.75. The point C' is also appended at the end of the local point list, provided that it does not lie within a pre-existing triangle.

The points in the local point list are then sorted according to distance from the middle point M, closest to farthest, and successive points P from the local point list are used to form the triangle ABP. A triangle is acceptable if it does not contain any other front-list nodes, and the bisector PM does not pierce through existing elements or intersect their edges. If, at any time, a point from the local list satisfies all conditions, then it is adopted as node C, and the search terminates even though all points in the local point list may have not been examined. If no points from the local point list satisfies all conditions, then the value of γ is increased by a preset factor, and this step is repeated until a suitable point is found.

(6) Having introduced the new element ABC, we update the *connectivity list*, the *front list*, and the *node list*. In the connectivity list, the three vertices of the new element are recorded under the element number. In the front list, the current front AB as well as any edges of element ABC that are already listed is removed, whereas those edges that have not been previously recorded are added. Likewise, if point C has not already been introduced, it is added to the *point list*.

(7) Steps (2) through (6) are repeated until the front list has been exhausted.

(8) After the whole of the surface has been triangulated, a connectivity improvement method called *edge swapping* is applied to reduce the element skeweness. The method changes the common edges of neighboring elements based on the Delaunay circumcircle criteria for flat domains. We found that, even though the surface is curved, a sufficiently dense grid allows the successful application of the method implemented as described in the next paragraph.

The circumcircle of a flat triangle is the circle passing through the three vertices. The center of this circle, called the circumcenter, may lie inside or outside the triangle. As the triangle becomes increasingly skewed, the circumcenter moves farther away from the

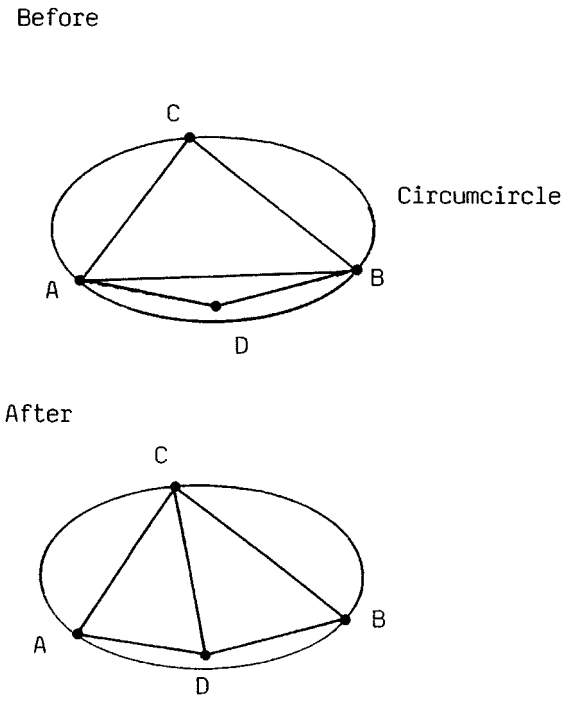


FIG. 6. Illustration of edge swapping between neighboring triangular elements to give better shaped elements. Swapping occurs when the circumcircle of triangle ABC contains a vertex D of the neighboring triangle ADB.

geometrical center of the triangle. As an example, let two adjacent triangles ABC and ADB share the vertices A and B, as shown in Fig. 6. If the circumcircle of triangle ABC contains a point D that lies outside this triangle, then the common edge AB is discarded, and a new edge CD is introduced to create the new elements ADC and BCD from the previous elements ABC and ADB. If the circumcircle does not contain such a point D, then the triangle remains unchanged. The edge swapping technique maximizes the minimum angles of both triangles by reducing the radii of the circumcircles. Edge swapping does not introduce new marker points, neither does it change the total number of elements. It should be noted, however, that in some cases edge swapping might lead to worse representations of a discretized surface, and it is thus not a panacea.

(9) The flat triangular elements are transformed into six node quadratic triangular elements by projecting the mid-points of the edges onto the surface. The unit normal vectors of these points are computed by analytical evaluation by means of interpolation through a coarse grid. The connectivity and the point lists are updated to include the mid-points.

The variables and parameters of the numerical method are summarized in Table II. The effectiveness of the method was tested and confirmed by performing a number of tests to be discussed in Section 4.

2.4. *Grid-to-Grid Interpolation*

The accurate computation of the position vector, the unit normal vector, and the directional surface curvature are necessary for the successful implementation of the AFM. When the surface represents a fluid interface containing, for example, a surfactant, regridding must

TABLE II
Variables and Parameters for Surface Discretization by the Advancing-Front Method

	Variable or parameter	Function
1	Connectivity list	Explained in text.
2	Front list	Explained in text.
3	Point list	Explained in text.
4	α	Specified angle to control edge length according to curvature.
5	γ	Search parameter for admitting a new node.
6	ϕ	Parameter to control a triangle height.
7	δ	Parameter to control element aspect ratio.

be accompanied by interpolation for the surfactant concentration, and the interpolation should be sufficiently accurate so that unphysical Marangoni stresses due to surface-tension gradients are not introduced. It was mentioned earlier that the interpolated geometrical and physical variables can be obtained either by means of analytical evaluation or by interpolation through an underlying coarse grid. Interpolation is the only alternative when performing dynamic regridding in the course of a simulation, except at the initial instant.

The interpolation method projects a new marker point onto the underlying old grid, and interpolates for the surface variables through the element that hosts the projection. The performance of the method is sensitive to the identification of the host element.

Two methods were considered for identifying the host element. Let C be the marker point of interest where interpolation is to be performed. The first method finds the host element for point C by first projecting it in the plane that is defined by the three vertex nodes of each triangle, and then determining whether or not the projection lies within the triangle. The interpolation is carried out using the isoparametric representation; the host element is mapped from the physical space to the local parametric ξ, η space, and the interpolation is carried out as discussed in Ref. [21]. In particular, in order to find the appropriate values of ξ and η, we solve a system of two quadratic equations that define the projection of the marker point. The implementation of this method is straightforward, but difficulties arise when the projection lies too close to an element edge, whereupon the search method may find either no host element or two host elements. To prevent this occurrence, conditions can be specified for the proper identification of the host element, but this complicates the numerical method. As an alternative, we implemented a simpler but more laborious method that exploits the flexibility of the AFM.

First, we find the marker point on the old grid that is closest to the point C, and identify all elements that share this marker point. Second, we partition all elements in the list into a network of sub-triangles based on the local parametric variables ξ and η. In the present implementation, increments of 0.01 were used for $\Delta\xi$ and $\Delta\eta$. Third, we compute the Cartesian coordinates of every subgrid point based on the local parametric representation. Fourth, we temporarily store the subgrid point that is closest to the point C as point C''. The same procedure is followed for the rest of the elements listed in the first step, and the temporary point C'' is continuously updated when a subgrid point closer to point C has been found. Once all elements on the list have been checked, the point C is replaced by the point C'', and the desired surface variable at the new point C is computed by interpolation.

It should be noted that a minor adjustment in the position of marker point during the triangulation does not impair the effectiveness of grid generation, neither does it violate the imposed restrictions. On the contrary, the method effectively prevents errors caused by the iterative projection discussed in step (7) of the surface discretization, and minimizes possible surface distortions caused by continuous re-triangulation.

To test the effectiveness of the method, each component of the interpolated unit vectors normal to a biconcave disk were graphed over the entire surface. Since, the number of elements and marker points vary from the new grid to the old grid, the graphical representation is a proper way of examining the accuracy of the method. Figures 7a and 7b show the x component of the normal vector distribution over the flat side of the original and re-triangulated biconcave disk. Considering the moderate number of elements, and keeping in mind that the normal vector is a sensitive function of the surface geometry, the results appear satisfactory.

3. ADVANCING-FRONT METHOD FOR CLOSED SURFACES

The method for triangulating a closed surface is similar in many respects to that for an open surface discussed in Section 2. The main difference lies in the generation of the initial front list before the surface discretization. In the case of an open surface, the front list is generated by discretizing the boundary curve. In the case of a closed surface, the initial front list is generated by introducing a triangular element at the region of highest magnitude of mean or directional curvature.

We begin by determining the marker point with the highest magnitude of the mean curvature κ_m or directional curvature κ_c, named point A. Among all marker points defining the elements that share the marker point A, we find the one with the highest magnitude or mean or directional curvature, excluding point A, named point B. In the third step, we compute the unit vector directed from point A to point B and the edge distance $\Delta s = 2\sin(\alpha/2)/\kappa_m$ or $\Delta s = 2\sin(\alpha/2)/\kappa_c$, and determine a new location for point B. The normal vector at the new location is computed by interpolation, as discussed in Subsection 2.4. The third point C defining the triangle ABC is determined by following steps (3)–(6) of the surface discretization method. The initial front list for the surface discretization contains the nodes and length defining the edges of the triangular element ABC.

Once the front list has been established, steps (2)–(9) described in the surface discretization method are applied to discretize the rest of the surface. The effectiveness of the method was tested and confirmed by performing a number of tests, as will be discussed in Section 4.

4. APPLICATIONS

The algorithms described in the preceding sections were applied to describe the shapes of several stationary or evolving, open or closed surfaces. Sample results, along with a critical discussion of the performance of the method, are now presented.

4.1. *Stationary Surfaces*

Figures 8a–8c show the triangulated surface of a biconcave disk whose shape is similar to that of an undeformed red blood cell, and Figs. 8d–8f show the triangulated

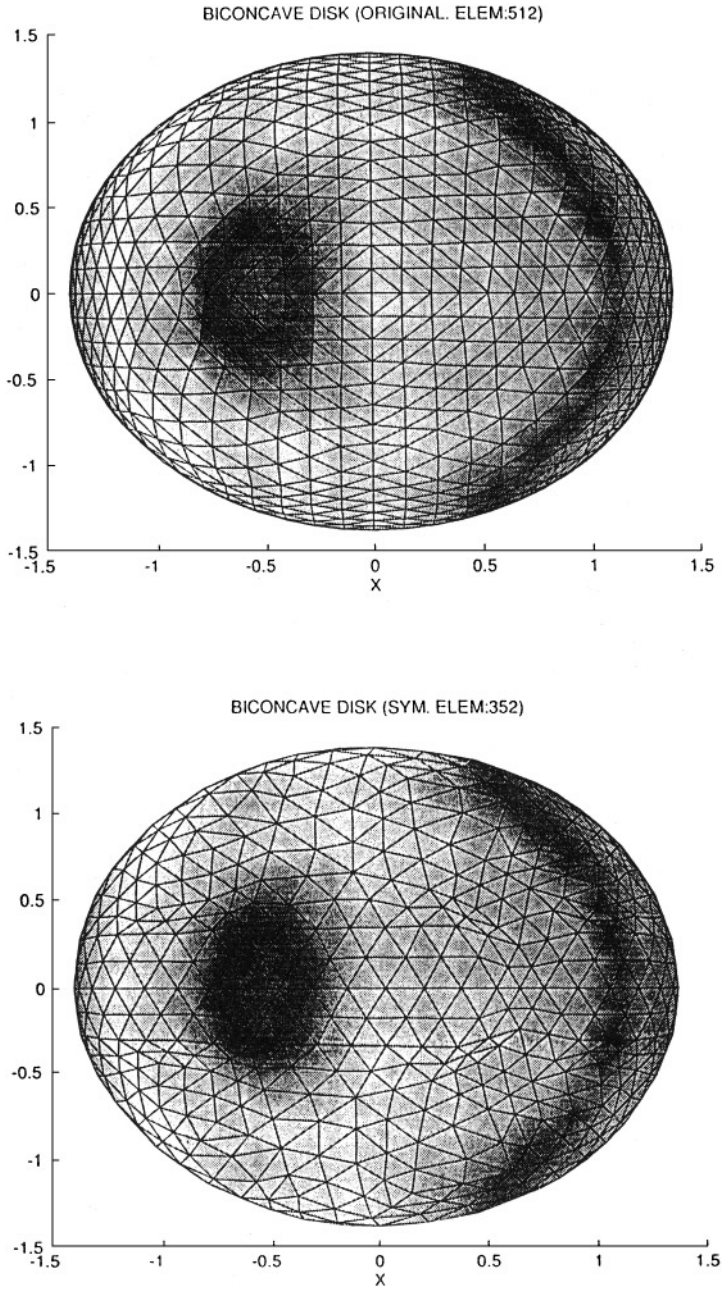
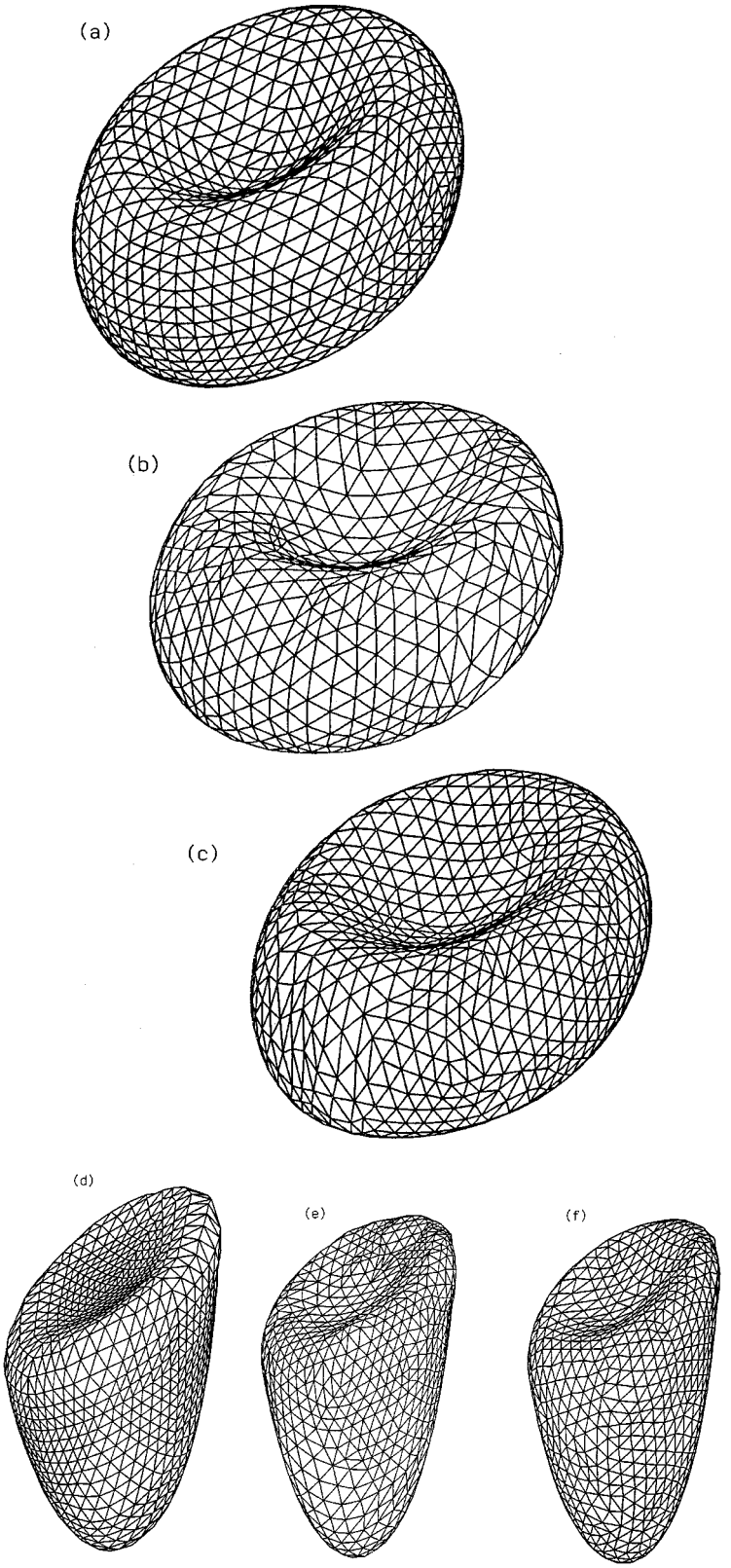


FIG. 7. Validation of surface interpolation. Gray-scale comparison of the x-component of the unit normal vector on (a) the regularly gridded of a biconcave disk and (b) the re-gridded surface with much fewer elements.

FIG. 8. Discretization of the biconcave disk and slipper-shaped drop by various methods. (a) Triangulation of a biconcave disk by projection of marker points descending from an icosahedron. (b), (c) Open-surface and closed-surface discretizations by utilizing the underlying grid shown in (a). (d) Magnification of a single slipper shaped drop shown on Fig. 1. (e), (f) Open-surface and closed-surface discretizations by utilizing the underlying grid shown in (d).


(a)
(b)
(c)
(d)
(e)
(f)

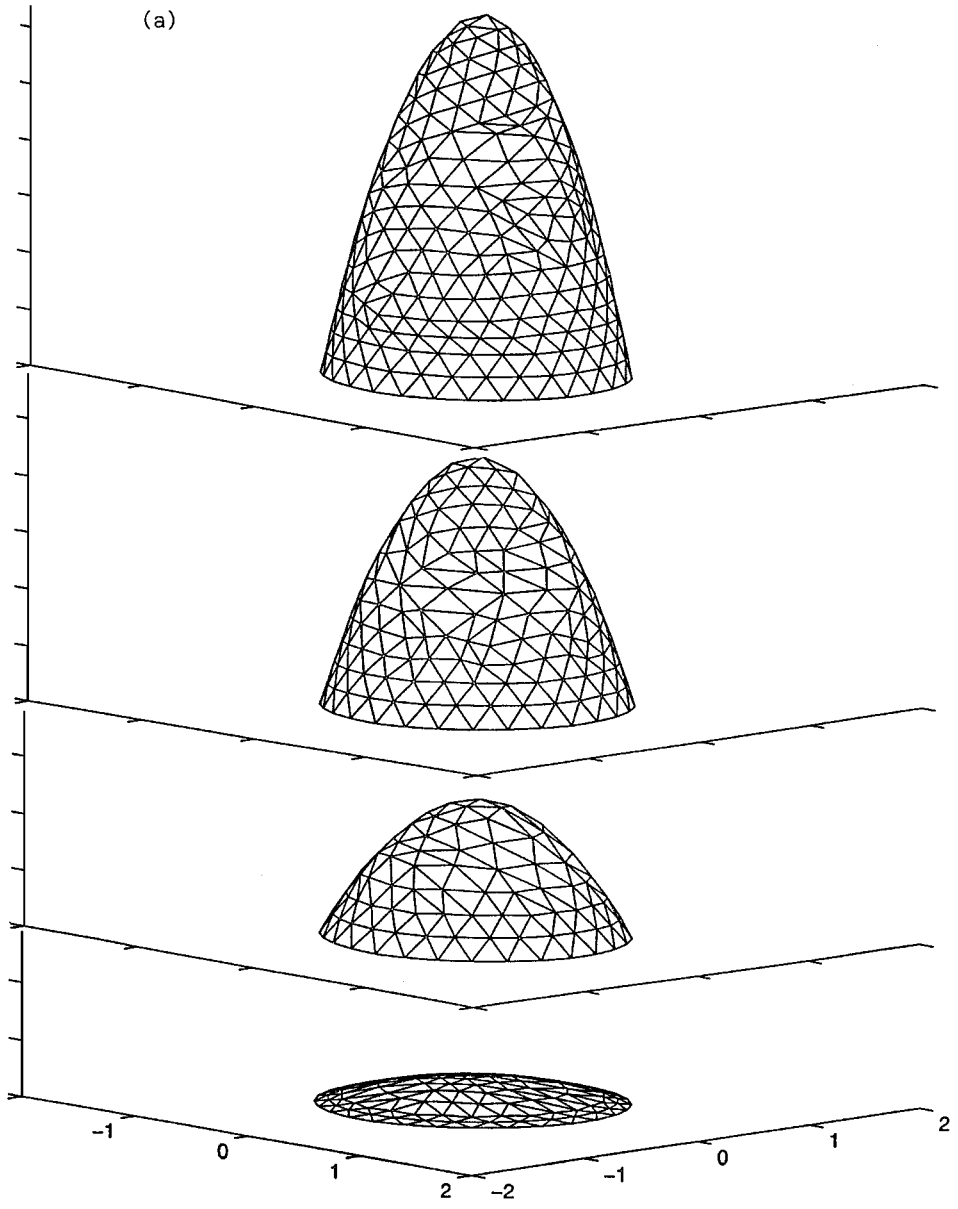


FIG. 9. (a), (b) Evolution of an initially flat disk whose center is located at the streamline of maximum velocity of an unbounded parabolic flow, at successive times.

surface of the slipper-like droplet mentioned in the Introduction. Both shapes have a reflection symmetry with respect to a mid-plane. The triangulations were produced using either the open-surface or the closed-surface discretization algorithm. In the first case, the surface generated on the upper half-space was reflected to the bottom half-space to give a closed surface. Thus, whereas the open-surface algorithm produces a shape that respects the symmetry of the shape, the closed-surface algorithm may generate triangles whose edges cross the mid-plane. Regridding was effected by interpolation through the original grid

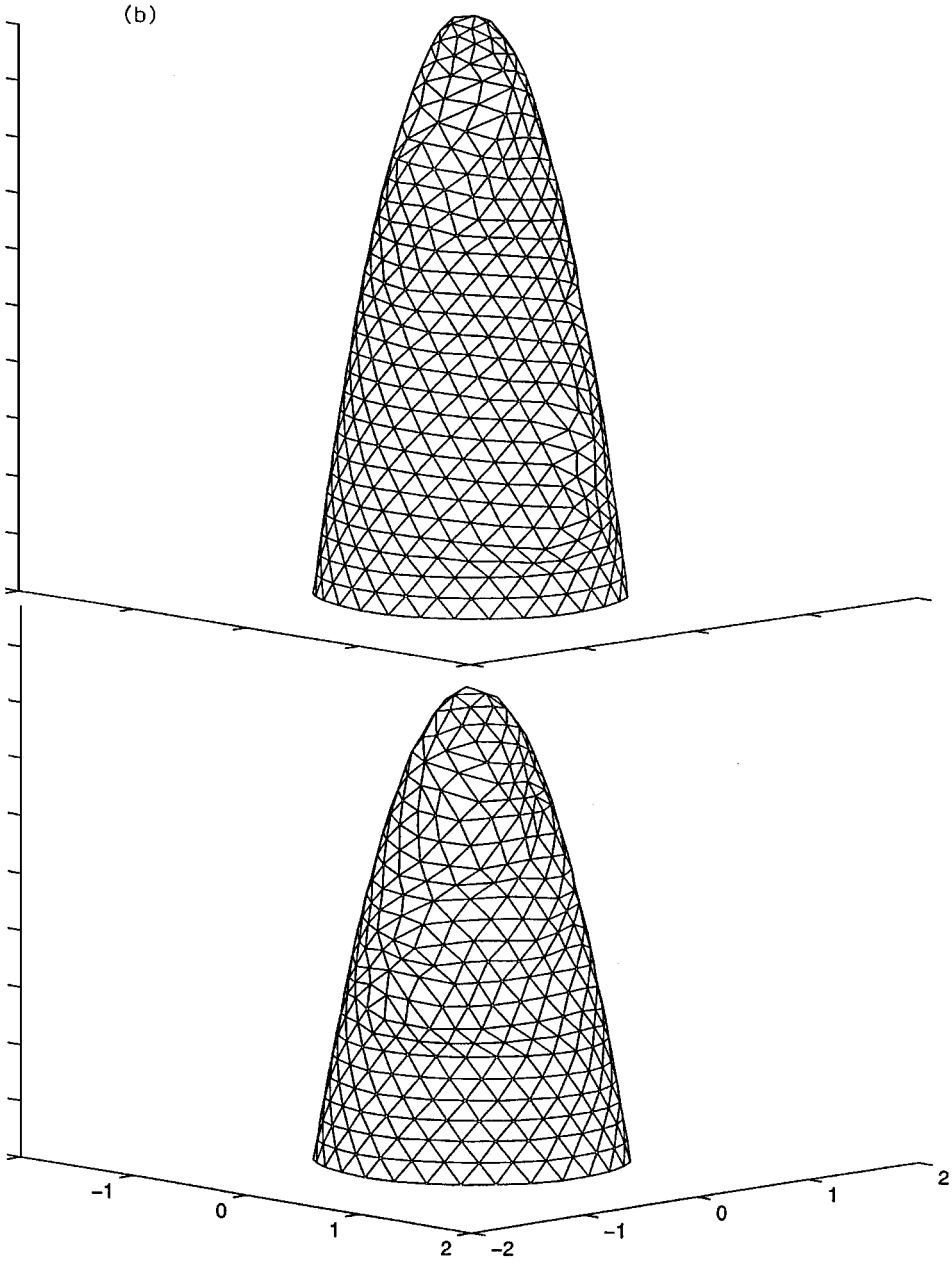


FIG. 9—*Continued*

that was generated by analytical expressions or supplied from the results of a dynamical simulation [3].

For the open-surface triangulations shown in Figs. 8b and 8e, we used $\alpha = 0.315$ and 0.375, respectively, for the biconcave disk and the slipper shape. In both cases, the redefined grid is comparable in quality, or superior to the original one. The improvement is particularly evident in the case of the slipper shape where the congested elements in the dimpled area of

the original grid are replaced by fewer elements, and the poorly represented high curvature region around the mid-plane is described by a denser concentration of elements. For the closed-surface triangulations shown in Figs. 8c and 8f, we used $\alpha = 0.17$ for both the biconcave disk and the slipper shape, and obtained triangulations that are comparable in quality with those resulting from the open-surface triangulation. The CPU time necessary for these triangulations is on order of a few minutes on a SUN SPARCstation 20. The closed-surface discretization is faster than the open-surface discretization. The difference is due to additional work necessary for checking the line segment variation in the curve discretization portion of the method.

4.2. *Evolving Surfaces*

In a more demanding series of tests, we used the open-surface discretization algorithm to adaptively describe the evolution of several types of surfaces that are convected either passively or actively under the action of a specified flow. The marker points are material point particles moving with the fluid velocity. In the case of an active surface, the velocity depends on the instantaneous shape of the surface or, more precisely, on the instantaneous relative position of the marker points.

Passively deforming disk in parabolic flow. Figures 9a–9b show the evolution of a flat circular disk convected passively in an unbounded rectilinear parabolic flow. At the initial instant, the disk is placed perpendicular to the streamlines of the unidirectional flow, and its center lies on the streamline with the maximum velocity. The numerical parameters were kept at the fixed values $\alpha = 0.523$ and $\phi = 1.4$ throughout this computation. Regridding was performed at the times corresponding to the shapes displayed. As the disk is stretched, while remaining axisymmetric, the number of elements is increased in a monotonic fashion so that the surface is described with adequate resolution at every stage.

Figure 10 shows the evolution of a disk whose center lies off the axis of the parabolic flow. These results were obtained with $\alpha = 0.807$ and $\phi = 1.4$. In this case, the disk deforms in a non-axisymmetric fashion and maximum mean curvature develops off the axis. The quality of the adaptive triangulation is comparable to that of the axisymmetric deformation.

Deformation of a viscous drop in simple shear flow. In the most interesting and computationally intensive series of tests, we combined the advancing-front method with a boundary integral method for Stokes flow to compute the large deformation of a liquid drop with viscosity $\lambda\mu$ suspended in an infinite ambient fluid with viscosity μ, under the action of a simple shear flow directed along the x axis with velocity $\boldsymbol{u} = (Gy, 0, 0)$; G is the constant shear rate, and the interface has a constant surface tension γ. The boundary element method is described in detail in Refs. [1–4]. Briefly, the numerical procedure involves solving an integral equation for the three components of the velocity over the interface, where the solution is assumed to vary in a quadratic fashion over the curved six-node triangles. The mean curvature of the interface involved in the integral equation is computed analytically from the local parametric representation.

Several previous experimental, analytical, and numerical studies have shown that when λ is less than approximately 4, there is a critical capillary number $Ca = \mu G a/\gamma$ above which

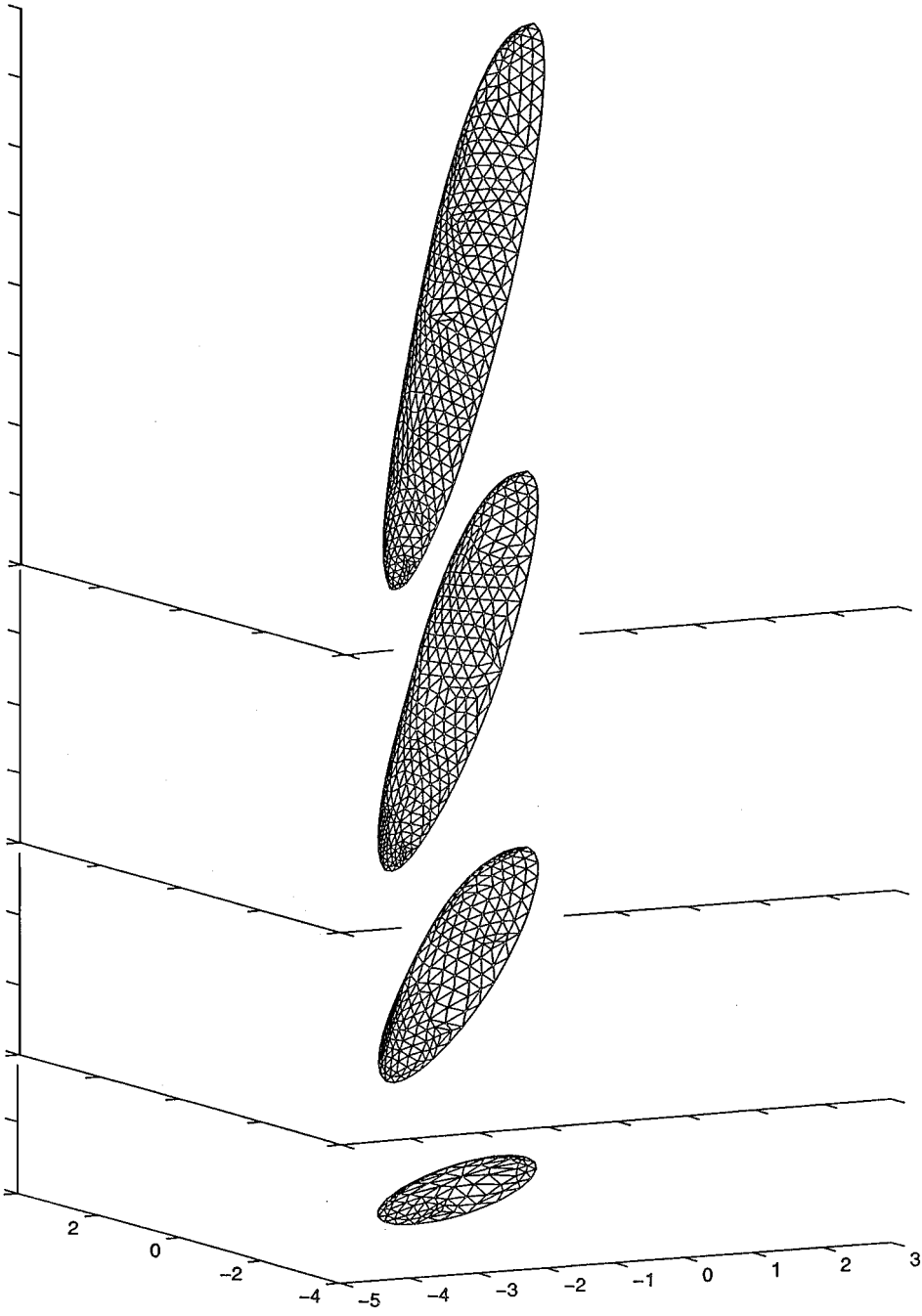


FIG. 10. Evolution of an initially flat disk whose center is located off the streamline of maximum velocity of an unbounded parabolic flow, at successive times.

the drop continues to deform without reaching a steady shape, and this leads to disintegration or breakup at long times; a is the equivalent drop radius. The process of continued deformation and disintegration was described in qualitative terms by Rumscheidt and Mason [23]. When λ is higher than about 4, the drop deforms and reaches a steady state irrespectively of the value of the capillary number.

The deformation of the drop is typically described in terms of the Taylor deformation parameter $D = (L - B)/(L + B)$, where L and B are the maximum and minimum drop dimensions in the xy plane. The orientation of the drop can be expressed in terms of the angle θ that the maximum axis of deformation forms with the x axis. Figures 11a and 11b show graphs of D and θ at steady state for $\lambda = 1$ computed from the present simulations with the open-surface triangulation followed by reflection, along with results presented in previous experimental and computational investigators. At low capillary numbers the present results coincide with those obtained by the previous numerical studies of Rallison [24] and Kennedy *et al.* [1], and are in excellent agreement with the experimental results of Rumscheidt and Mason [23]. As the capillary number is raised, the present results stay close to the experimental results almost all the way up to the critical capillary number of 0.42+ above which steady stapes are not established. Prior numerical results

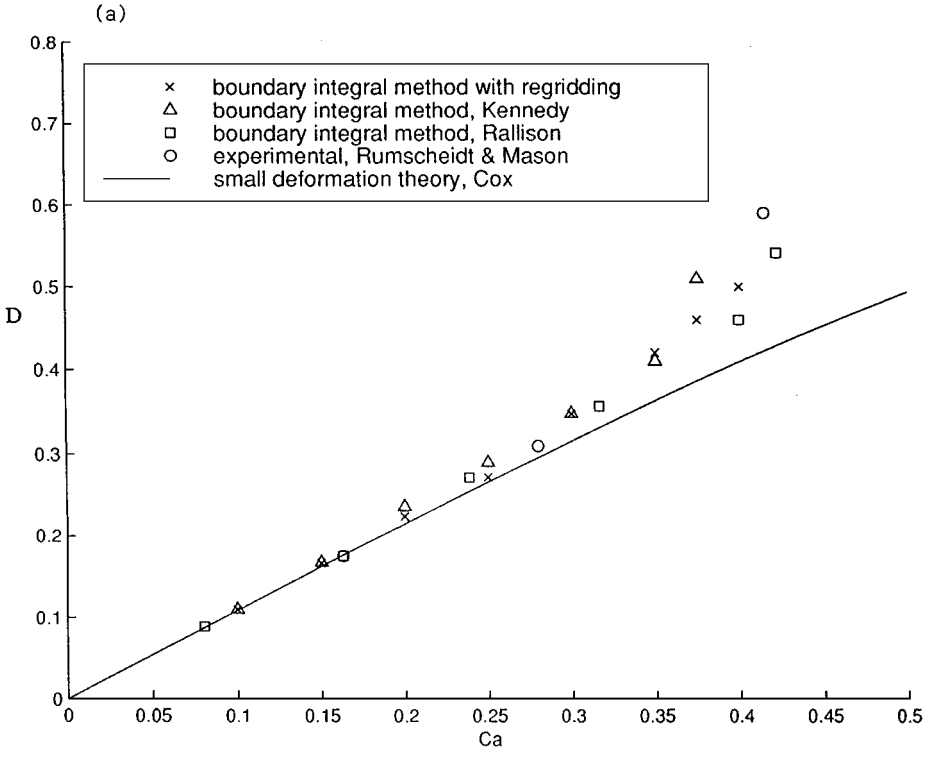


FIG. 11. Deformation of a viscous drop in simple shear flow. Comparison of (a) the steady-state drop deformation parameter D, and (b) the steady-state drop orientation angle θ measured in degress for viscosity $\lambda = 1.0$; $\times$, boundary integral computations; $\triangle$, boundary integral computations of Kennedy *et al.* [1], $\square$, boundary integral computation of Rallison [24]; $\bigcirc$, experimental results by Rumscheidt and Mason [23]; the solid line represents the asymptotic results by Cox [25] for small deformation.

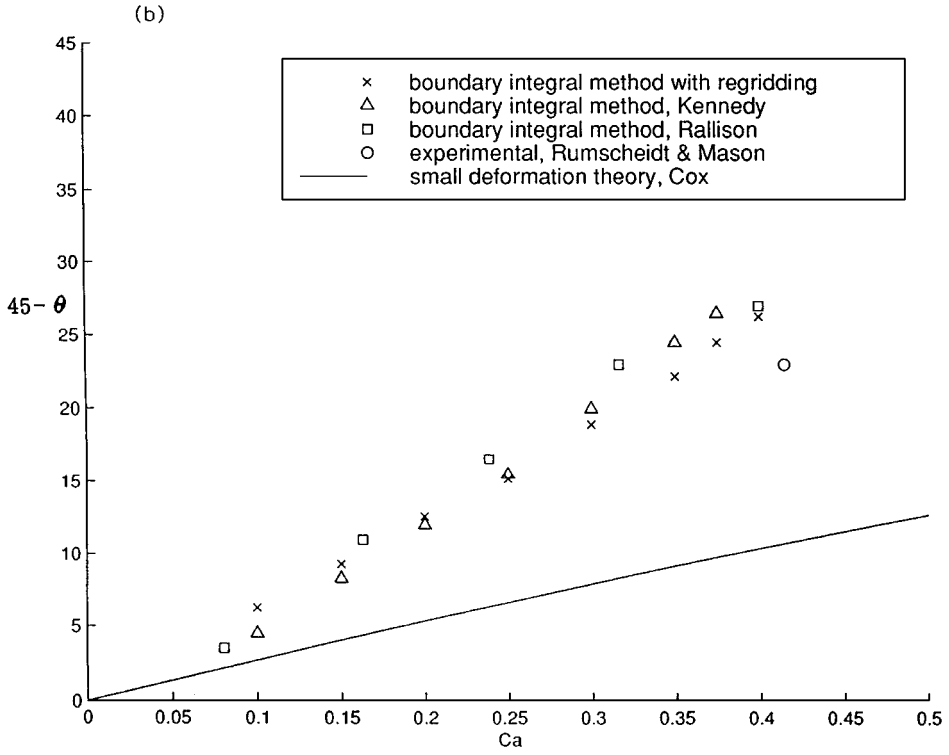


FIG. 11—*Continued*

show some deviations that may be attributed to numerical inaccuracies due to insufficient triangulation.

Figures 12 and 13 illustrate sequences of deforming shapes following the impulsive application of the simple shear flow, starting from the spherical shape. The first case corresponds to $\lambda = 1$, $Ca = 0.45$, and the second case to $\lambda = 0.08$, $Ca = 0.55$. In both cases, the capillary number is supercritical; that is, the drop continues to elongate without reaching a steady state. In the first case, we used a time step of $0.01/G$, and carried out the simulation up to time $8.0/G$; in the second case, we used a time step of $0.02/G$, and carried out the simulation up to time $5.6/G$. Each simulation required a total CPU time on a SUN SPARCstation 20 on the order of 5 days.

During the early stages of the deformation, the shapes of the drops shown in Figs. 12 and 13 show similar behaviors, but differences arise when the deformation becomes large. In the first case, the drop develops a cylindrical shape with bulbous ends; in the second case, a capillary Rayleigh instability develops during the final stages of the deformation, causing the drop to break up into two pieces. The capillary instability in the first case seems to be delayed by the appreciable viscosity of the drop fluid, or else is suppressed by the ambient shear flow. In the classification of Rumscheidt and Mason [23], the drop shown in Fig. 12 shows a B-2 type of deformation, whereas the drop shown on Fig. 13 shows a B-1 type of burst. Rumscheidt and Mason [23] recorded B-1 burst for $\lambda = 1$, and B-2 burst for $\lambda = 0.7$, which seems to contradict the trends observed in our simulations. It is

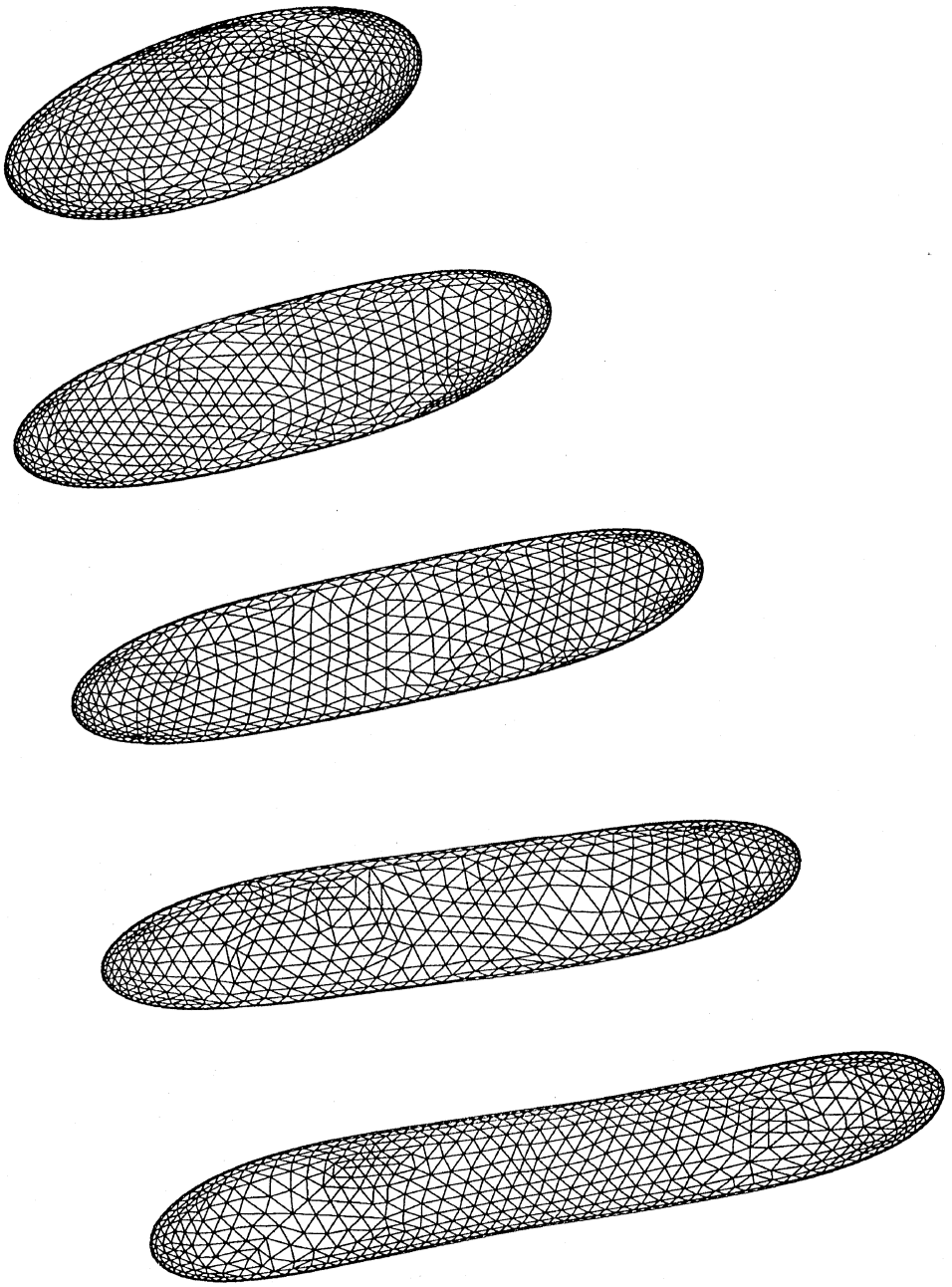


FIG. 12. (a) Illustration of a continuously deforming drop for $\lambda = 1.0$ and $Ca = 0.45$ at times $Gt = 1.6$, 3.4, 4.6, 6.2, 7.8; (b) Plane view and three-dimensional perspectives at time $Gt = 7.8$.

known, however, that the capillary number is an important parameter in determining the long time behavior.

In the simulations described earlier, regridding by the AFM was done after a fixed number of times steps, typically on the order of 20. In principle, triangulation should be enabled when the quality of the grid—measured by the minimum internal angle of a triangle, by

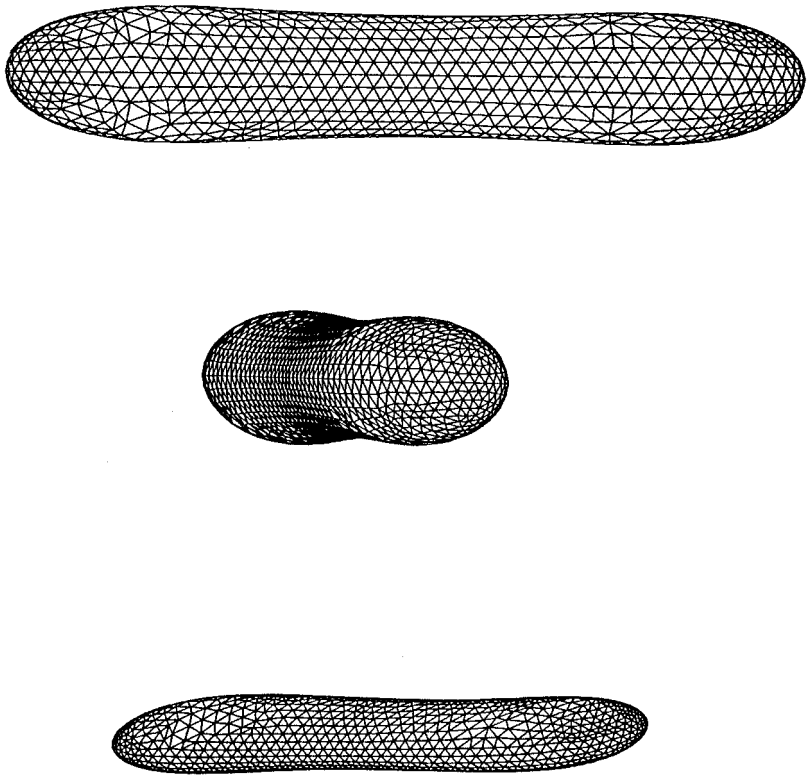


FIG. 12—*Continued*

the angle subtended by the three points that define each side of a triangle, or a combination thereof—falls below a specified level. In practice, in order to avoid capillary instabilities, the time step is adjusted to scale with the minimum triangle size, and triangulation after a fixed number of time steps is appropriate.

Repetitive triangulation with a fixed value of the discretization parameter α leads to an excessive number of triangles at a relatively early stage of the motion, and these could not be accommodated by the available computational resources. To circumvent this difficulty, tests were performed on a slowly deforming drop for a short period of time to understand the behavior of the performance of the AFM in more quantitative terms. Figure 14 shows a graph of the total number of elements versus the normalized maximum curvature of the interface for several values of the parameter α. With these results as a point of departure, the following empirical polynomial equation was adopted to control parameter α at the time step i,

$$\alpha^{(i)} = \alpha^{(i-1)} + 2\psi\pi(\Delta D - \Delta D^2 - \Delta D^3)/180,$$

where $\Delta D = D^{(i)} - D^{(i-1)}$, and ψ is a relaxation factor. All simulations were performed with an initial value for α of 0.28, and the coefficient ψ was chosen to lie in the range (0, 2) depending on the value of the capillary number.

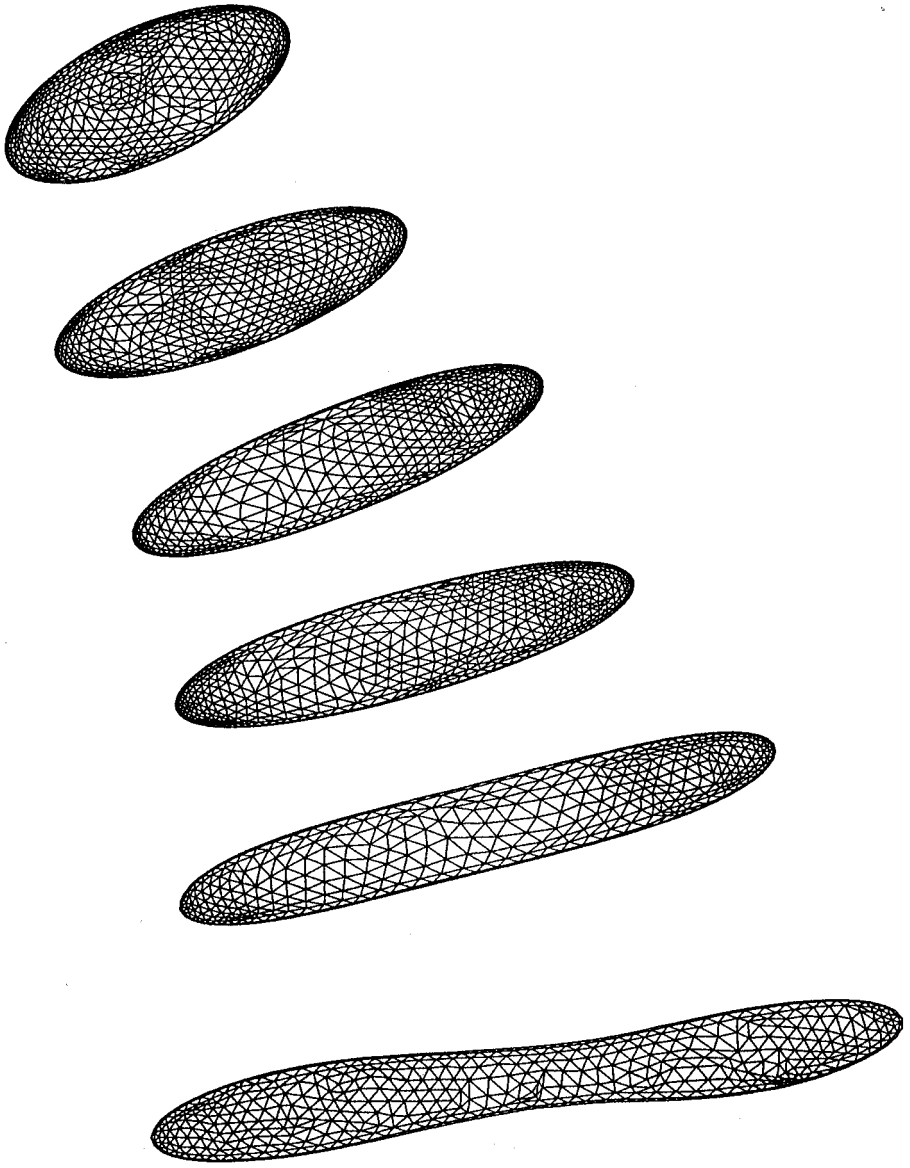


FIG. 13. Illustration of a continuously deforming drop for $\lambda = 0.08$ and $Ca = 0.55$ at times $Gt = 0.8$, 1.6, 2.4, 3.2, 4.0, 4.2.

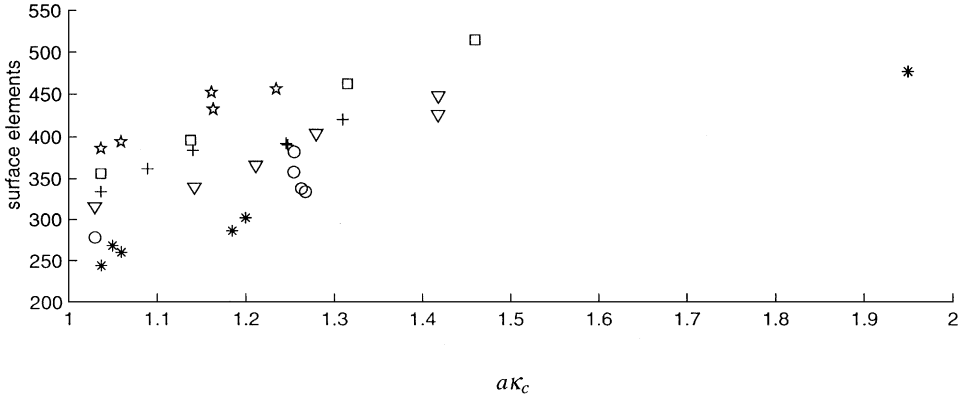


FIG. 14. Sensitivity of the Advancing-Front Method to changes in surface geometry for a drop undergoing small deformations; ☆, $\alpha = 0.253$; □, $\alpha = 0.263$; +, $\alpha = 0.274$; ∇, $\alpha = 0.287$; ○, $\alpha = 0.300$; ∗, $\alpha = 0.315$.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the advice and guidance of Chad Coulliette and Steve Yon. Support for this work was provided by the *National Science Foundation, and the SUN Microsystems Corporation.* Acknowledgment is made to the Donors of the Petroleum Research Fund, administered by the *American Chemical Society*, for partial support of this research.

REFERENCES

1. M. R. Kennedy, C. Pozrikidis, and R. Skalak, Motion and deformation of liquid drops and the rheology of dilute emulsions in simple flow, *Comput. & Fluids* **23**, 251 (1994).
2. S. Ramanujan and C. Pozrikidis, Deformation of liquid capsules enclosed by elastic membranes in simple shear flow: large deformations and the effect of capsule viscosity, *J. Fluid Mech.* **361**, 117 (1998).
3. C. Coulliette and C. Pozrikidis, Motion of an array of drops through a cylindrical tube, *J. Fluid Mech.* **358**, 1 (1998).
4. S. Yon and C. Pozrikidis, A finite-volume/boundary-element method for flow past interfaces in the presence of surfactants, with application to shear flow past a viscous drop, *Comput. & Fluids*, in press.
5. F. X. Giraldo, Lagrange–Galerkin methods on spherical geodesic grids, *J. Comput. Phys.* **136**, 197 (1997).
6. D. J. Mavriplis, Unstructured grid techniques, *Ann. Rev. Fluid Mech.* **29**, 473 (1997).
7. M. Loewenberg and E. J. Hinch, Numerical simulation of a concentrated suspension in shear flow, *J. Fluid Mech.* **321**, 395 (1996).
8. A. Z. Zinchenko, M. A. Rother, and R. H. Davis, A novel boundary-integral algorithm for viscous interaction of deformable drops, *Phys. Fluids* **9**, 1493 (1997).
9. V. Cristini, J. Blawzdziewicz, and M. Loewenberg, Drop breakup in three-dimensional viscous flows, *Phys. Fluids* **10**, 1781 (1998).
10. S. Yon and C. Pozrikidis, Deformation of a liquid drop or gas bubble adhering to a plane wall: Significance of the drop viscosity and the effect of an insoluble surfactant, submitted for publication.
11. B. Hamann, I. Trotts, and G. Farin, On approximating contous of the piecewise trilinear interpolant using triangular rational-quadratic Bézier patches, *IEEE Trans. Vis. Comput. Graph.* **3**, 215 (1997).
12. K. Nakahashi and G. S. Deiwert, Self-adaptive-grid method with application to airfoil flow, *AIAA J.* **25**, 513 (1987).
13. S. H. Lo, A new mesh generation scheme for arbitrary planar domains, *Int. J. Numer. Methods Eng.* **21**, 1403 (1985).
14. J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz, Adaptive remeshing for compressible flow computations, *J. Comput. Phys.* **72**, 449 (1987).
15. E. Hinton, N. V. R. Rao, and M. Ozakca, Mesh generation with adaptive finite element analysis, *Adv. Eng. Software*, **13**(5/6), 238 (1991).
16. K. Nakahashi and D. Sharov, *Direct Surface Triangulation Using the Advancing Front Method*, AIAA 95-1686-CP, 1995.
17. P. L. George and E. Seveno, The advancing front mesh generation method revisited, *Int. J. Numer. Methods Eng.* **37**, 3605 (1994).
18. R. Löhner, Regridding surface triangulations, *J. Comput. Phys.* **126**, 1 (1996).
19. C. Pozrikidis and J. J. L. Higdon, Nonlinear Kelvin–Helmholtz instability of a finite vortrex layer, *J. Fluid Mech.* **157**, 225 (1985).
20. B. Hamann and J.-L. Chen, Data point selection for piecewise linear curve approximation, *Comput. Aided Geom. Design.* **11**, 289 (1994).
21. C. Pozrikidis, *Numerical Computation in Science and Engineering* (Oxford Univ. Press, New York, 1998).
22. C. Pozrikidis, *Introduction to Theoretical and Computational Fluid Dynamics* (Oxford Univ. Press, New York, 1997).

23. F. D. Rumscheidt and S. G. Mason, Particle motions in sheared suspensions. XII. Deformation and burst of fluid drops in shear and hyperbolic flow, *J. Colloid Sci.* **16**, 238 (1961).
24. J. M. Rallison, The deformation of small viscous drops and bubbles in shear flows, *Ann. Rev. Fluid Mech.* **16**, 45 (1984).
25. R. G. Cox, The deformation of a drop in a general time-dependent fluid flow, *J. Fluid Mech.* **37**, 601 (1969).