

# ALGORITHMS FOR AREA PRESERVING FLOWS

CATHERINE KUBLIK, SELIM ESEDOĞLU AND JEFFREY A. FESSLER

**Abstract.** We propose efficient and accurate algorithms for computing certain area preserving geometric motions of curves in the plane, such as area preserving motion by curvature. These schemes are based on a new class of diffusion generated motion algorithms using signed distance functions. In particular, they alternate two very simple and fast operations, namely convolution with the Gaussian kernel and construction of the distance function, to generate the desired geometric flow in an unconditionally stable manner. We present applications of these area preserving flows to large scale simulations of coarsening.

**Key words.** Diffusion generated motion, level set, area preserving mean curvature motion, coarsening

**AMS subject classifications.** 65C20

**1. Introduction.** Computing the motion of interfaces, e.g., curves in the plane or surfaces in space, is an essential component of many applications. For instance, in image processing and computer vision, many popular variational models for segmentation and reconstruction involve initializing a curve or surface and then evolving it towards features (e.g., edges) that are of interest. Typically, the resulting motion is by a normal speed that includes geometric terms such as curvature of the interface and its derivatives. There are many numerical schemes for simulating geometric motions of interfaces, including front tracking, phase-field and level set methods. Front tracking methods [4] involve an explicit discretization of the interface and approximate its motion by moving marker particles located on the interface. These methods are well suited for two dimensional motions of curves that do not cross, but become very difficult to implement whenever topological changes occur, particularly in dimensions higher than two. On the other hand, phase-field [24, 27] and level set methods [28] represent the interface implicitly and are therefore able to naturally handle topological changes.

Motion by mean curvature has been extensively studied in the mathematics literature [15, 17, 19] and in applications such as crystal growth and image processing [1, 25]. Under this geometric flow each point  $x$  on a curve  $\Gamma$  moves with normal velocity  $v_N = \kappa(x)$ , where  $\kappa(x)$  is the mean curvature of the curve at  $x \in \Gamma$ . It is also called the Euclidean curve shortening flow since the Euclidean perimeter of a curve shrinks as quickly as possible when evolving according to this motion. A variant of this motion is the geometric flow that decreases the total perimeter of a collection of curves as quickly as possible while preserving the total enclosed area: it is referred to as area (or volume) preserving mean curvature motion. This motion finds applications in image processing [10, 16] and arises physically as a limit of a nonlocal model describing the phase separation of binary alloys [3, 18, 30], and as a limit of the Lifshitz-Slyozov-Wagner (LSW) mean field model with kinetic drag when the size of the particles are small compared to value of the kinetic drag [8]. The LSW model was introduced by Lifshitz, Slyosov and Wagner [20, 43] to study the general phenomenon of Ostwald ripening [26], which describes the dissolution/precipitation dynamics of precipitate crystals in a saturated solution: small precipitate crystals dissolve and redeposit themselves on the surface of larger crystals, thus creating a coarsening of the system size since small particles shrink and eventually vanish while larger ones grow. The area preserving mean curvature flow exhibits a similar coarsening behav-

ior (see Section 4). Mathematically, the area preserving mean curvature motion is characterized by its normal velocity  $v_N = \kappa - \bar{\kappa}$ , where  $\kappa$  is the mean curvature and  $\bar{\kappa}$  denotes the average mean curvature of the collection of curves evolving under this motion.

Several schemes based on the level set method [10, 29, 45] and diffusion generated motion [38] have been proposed for approximating this motion. In this paper, we describe new schemes for area preserving flows based on a new class of algorithms that generate the desired interfacial motion by alternating two simple and efficient steps: construction of the signed distance function, and convolution with a kernel (usually a Gaussian kernel). The resulting schemes are unconditionally stable, and have low, namely  $O(N \log N)$ , per time step cost, where  $N$  is the total number of grid points. In addition, we present applications of these efficient area preserving algorithms to large scale simulations of area preserving curvature motion. The paper is organized as follows. In Section 2, we give a brief description of previous diffusion generated motion based algorithms, and describe our new area preserving schemes in Section 3. We present applications of our new algorithms in Sections 4 and conclude in Section 5.

**2. Diffusion generated motion.** The algorithms proposed in this paper have been motivated by the threshold dynamics idea of Merriman, Bence and Osher (MBO) [22, 23, 24]. The MBO algorithm is obtained by time splitting the well-known Allen-Cahn phase-field equation for motion by mean curvature. The resulting scheme alternates two steps, namely convolution and thresholding. More precisely, consider a set  $\Sigma$  in the domain  $\Omega \subset \mathbb{R}^2$ , with boundary  $\partial\Sigma$  evolving with normal speed  $v_N = \kappa$ , where  $\kappa(x)$  is the mean curvature of the interface  $\partial\Sigma$  at the point  $x \in \partial\Sigma$ . Then  $\partial\Sigma$  is the  $\frac{1}{2}$  level set of the characteristic function  $\chi_\Sigma$  of the set  $\Sigma$ :

$$\partial\Sigma =: \left\{ x : \chi_\Sigma(x) = \frac{1}{2} \right\}.$$

Given an initial set  $\Sigma_0$  defined through its characteristic function  $\chi_0$ , and a time step size  $\delta t > 0$ , the MBO scheme generates a discrete sequence  $\{\Sigma_j\}_{j \in \mathbb{N}}$  at subsequent times  $j(\delta t)$  in the following way: from the set  $\Sigma_{j-1}$ , obtain the set  $\Sigma_j$  by alternating between the following two operations:

1. *Diffusion step.* From the set  $\Sigma_{j-1}$  defined through its characteristic function  $\chi_{j-1}$  at  $(j-1)\delta t$ , solve the following initial value problem for a length of time  $\delta t$ :

$$\begin{cases} u_t = \Delta u \\ u(x, 0) = \chi_{j-1}(x), \end{cases}$$

which is equivalent to forming the function

$$\mathcal{L}(x) = (G_t * \chi_{j-1})(x),$$

where  $G_t$  is the  $n$ -dimensional Gaussian kernel given by

$$G_t(x) = \frac{1}{(4\pi t)^{\frac{n}{2}}} e^{-\frac{|x|^2}{4t}}. \quad (2.1)$$

2. *Sharpening step.*

$$\chi_j(x) = \begin{cases} 0 & \text{if } \mathcal{L}(x) < \frac{1}{2}, \\ 1 & \text{else.} \end{cases}$$

The location of the interface  $\partial\Sigma_j$  is given by the level set  $\{x \in \Omega : \chi_j(x) = \frac{1}{2}\}$ . The time discretization  $\{\partial\Sigma_j\}_{j \in \mathbb{N}}$  has been proven to converge to motion by mean curvature in the limit  $\delta t \rightarrow 0^+$  (see [2, 13, 21]). One of the main advantages of this algorithm is its unconditional stability. In fact, the scheme remains monotone (preserves the order of sets) for all choices of  $\delta t$ , independent of the spatial resolution. In addition, its computational complexity is low ( $O(N \log N)$  due to the FFT used in the convolution step), which makes it computationally more attractive than standard level set techniques that involve the solution of a nonlinear and degenerate PDE [28]. Nevertheless, there exists some semi-implicit schemes for level set methods described by Smereka in [40]. Several generalizations of the basic MBO scheme have been proposed for generating more complicated interfacial motions including anisotropic curvature motion, motion by curvature plus constant, and motion of multiple junctions [12, 21, 32, 33, 34, 35, 36, 37].

Despite its computational advantages, the MBO scheme inherits a major drawback from its construction using characteristic functions: inaccuracy on uniform grids. Indeed, characteristic functions cannot resolve the location of the interface better than the spatial grid size. Consequently, unless the grid size is refined concurrently with the time step size, the approximate motion generated by the scheme gets stuck. It is therefore necessary to discretize the scheme with methods that can provide subgrid accuracy in the location of the interface. This was done by Ruuth in [33] through the efficient use of an unequally spaced FFT. Such an adaptive strategy is especially needed for simulating high order motions.

To address this issue, a variant of the MBO scheme has recently been proposed by Esedoğlu, Ruuth and Tsai in [11]. The new algorithms differ from the original by representing the interface through its signed distance function instead of its characteristic function, thereby replacing the thresholding step with a *redistancing step*. The new schemes therefore simulate the motion of an interface by alternately diffusing and redistancing the signed distance function to the interface. The diffusion step consists of convolving the distance function with an appropriate kernel, usually chosen to be the Gaussian kernel, and the redistancing step simply consists of constructing the signed distance function to the interface from the previously diffused distance function. In these algorithms, the computationally very efficient thresholding step is replaced by the construction of the signed distance function, for which there are also very efficient ( $O(N \log N)$ ) algorithms (e.g., fast marching, fast sweeping, etc. [6, 31, 39, 41, 42]), hence maintaining the complexity of the original diffusion generated motion scheme. This new class of diffusion generated motion algorithms thus provides an efficient and highly accurate technique for simulating a wide range of interfacial motions.

**3. Proposed schemes.** In this section we introduce our area preserving schemes, and present systematic studies of their numerical convergence and accuracy.

**3.1. Diffusion generated motion using signed distance functions.** For a given open set  $\Sigma \in \mathbb{R}^2$ , we denote by  $d$  the signed distance function to its boundary  $\partial\Sigma$

$$d(x) := \begin{cases} \inf_{y \in \Sigma^c} |x - y| & \text{if } x \in \Sigma, \\ -\inf_{y \in \Sigma} |x - y| & \text{if } x \in \Sigma^c, \end{cases}$$

where  $d$  is positive in the interior of  $\Sigma$  and negative outside its closure. The signed distance function dynamics [11] for motions with normal speed of the form

$$v_N = \kappa + S,$$

where  $\kappa$  is the curvature and  $S$  is a given function mapping  $\mathbb{R}^2$  to  $\mathbb{R}$ , now reads as follows: from an initial set  $\Sigma_0$  defined through its signed distance function  $d_0(x)$  and a time step size  $\delta t > 0$ , generate a time discrete approximation  $\{\partial\Sigma_j\}_{j \in \mathbb{N}}$  at times  $j(\delta t)$  by alternating between the following two operations:

1. *Diffusion step.* From the set  $\Sigma_{j-1}$ , defined through its signed distance function  $d_{j-1}$  at time  $(j-1)\delta t$ , form the function

$$\mathcal{L}(x) = (G_t * d_{j-1})(x) + S_{j-1}(x)(\delta t),$$

where  $G_t$  is the  $n$ -dimensional Gaussian kernel given in (2.1).

2. *Redistancing step.* Construct  $d_j$ , the signed distance function to the zero level set of  $\mathcal{L}$ , defining the new set  $\Sigma_j$  (and thus its boundary  $\partial\Sigma_j$ )

$$d_j(x) = \mathbf{Redist}(\mathcal{L}(x)),$$

where **Redist** denotes the construction of the signed distance function to the zero level set of  $\mathcal{L}$ ; this operation is commonly used in level set techniques [28], and unlike thresholding does not require making a binary decision at each grid point about whether the point is inside or outside the zero level set, thus allowing subgrid resolution to be maintained.

**3.2. Algorithms for area preserving flows.** Building on the distance function dynamics for curvature motions, we propose new and efficient algorithms for area preserving flows in two dimensions. These algorithms generate interfacial motions with normal velocities

$$v_N = \kappa - \bar{\kappa} + S, \tag{3.1}$$

where  $\kappa$  denotes the curvature,  $\bar{\kappa}$  the average curvature and  $S = S(x, t)$  is an additional normal speed term that may depend on space and time. Such terms arise for example in computer vision applications from data fitting terms in variational models [10]. The core of our algorithm is the scheme for area preserving curvature motion which evolves interfaces with normal velocity  $v_N = \kappa - \bar{\kappa}$ . Under this motion,  $n$  disjoint curves  $\Gamma_i$  will evolve to decrease their total length while maintaining the total enclosed area constant. In our algorithms, we use the fact that both  $\kappa$  and  $\bar{\kappa}$  can be calculated very easily using the signed distance function to the interface.

Let us now recall a few well known properties of the signed distance function that hold more generally in  $\mathbb{R}^n$  (see e.g. [9, 14]). Consider the set  $\Sigma \subset \Omega$  defined through its signed distance function  $d$ , and let  $\partial\Sigma$  be its boundary. The first property of  $d$  is based on the fact that the normals to a smooth interface do not focus immediately, so that the signed distance function is smooth in a tubular neighborhood  $T$  of  $\partial\Sigma$ , and linear with slope one along the normals:

$$|\nabla d| = 1 \text{ for all } x \in T, \text{ with boundary condition } d|_{x \in \partial\Sigma} = 0. \tag{3.2}$$

The second property is that the Laplacian of the signed distance function  $d$  at a point  $x$  gives, up to a multiplicative constant depending on the dimension, the mean curvature of the isosurface of  $d$  passing through  $x$ :

$$\Delta d(x) = (n-1)H(x), \tag{3.3}$$

where  $H(x)$  denotes the curvature of the level set  $\{\xi : d(\xi) = d(x)\}$ , and  $n$  is the dimension. In two dimensions, we will denote  $\kappa(x)$  the curvature of the curve

$\partial\Sigma = \{\xi : d(\xi) = 0\}$ , so that equation (3.3) simplifies to

$$\Delta d(x) = \kappa(x). \quad (3.4)$$

Before moving on to the expression of the average curvature  $\bar{\kappa}$  in terms of the signed distance function  $d$ , we need to recall some simple definitions and properties. The average curvature of a curve  $C$  is defined as

$$\bar{\kappa} = \frac{1}{|C|} \int_C \kappa ds, \quad (3.5)$$

where  $|C|$  denotes the length of the curve  $C$ . For a two dimensional simply connected set  $\Sigma$  of genus  $p$ , the average curvature of its boundary  $\partial\Sigma$  can be expressed as

$$\bar{\kappa} = \frac{2\pi(1-p)}{|\partial\Sigma|},$$

where  $|\partial\Sigma|$  is the total length of the boundary of  $\Sigma$ . The genus number  $p$  can be interpreted as the number of ‘‘holes’’ in the set  $\Sigma$ . If the set  $\Sigma$  is made up of  $K$  connected components  $\Sigma = \bigcup_{j=1}^K \Sigma_j$ , with  $\Sigma_j$  being a set of genus  $p_j$ , then the average curvature of  $\partial\Sigma$  becomes

$$\bar{\kappa} = \frac{2\pi \left( K - \sum_{j=1}^K p_j \right)}{|\partial\Sigma|}, \quad (3.6)$$

From expression (3.6), we see that the only quantity left to compute is the perimeter of the set  $\Sigma$ ,

$$\begin{aligned} |\partial\Sigma| &= \int_{\partial\Sigma} ds \\ &= \int_{\partial\Sigma} |\nabla d|^2 ds \quad (\text{since by definition of the signed distance function, } |\nabla d| = 1) \\ &= \int_{\partial\Sigma} \nabla d \cdot \nu ds \quad (\text{where } \nu = |\nabla d| \text{ is the outward unit normal}) \\ &= \int_{\Sigma} \Delta d dx. \quad (\text{by the divergence Theorem}) \end{aligned}$$

Thus we have

$$|\partial\Sigma| = \int_{\Sigma} \Delta d dx, \quad (3.7)$$

which provides a simple relation between the length of the boundary of a set and its signed distance function. In our computations, we use equations (3.6) and (3.7) to compute the average curvature of a set. Notice that all the computations are done using only the signed distance function to the interface. Note also that under area preserving curvature motion, the boundaries of sets will evolve to decrease their total length while maintaining the total area at its initial value. Consequently, the final state of the evolution is a disk with area equal to the initial total area. In particular, disks preserve their circular symmetry under this motion.

The complete algorithm for the general motion with normal speed  $v_N = \kappa - \bar{\kappa} + S$  builds on the area preserving curvature motion scheme by simply shifting the location

of the redistancing process by a constant determined by  $S(x, t)$ . For clarity in the description of the algorithms, we define

$$\#\{d > 0\} := \text{number of connected components of } \{x : d(x) > 0\}.$$

**ALGORITHM 1.** *Given the initial set  $\Sigma_0$  defined through its signed distance function  $d_0(x)$  and a time step  $\delta t > 0$ , generate the sets  $\Sigma_j$  via their signed distance functions  $d_j(x)$  at the subsequent discrete times  $t = j(\delta t)$  by alternating between the following two steps:*

1. *Using  $G_t$  in (2.1), form*

$$\begin{aligned} \mathcal{L}(x) = & (d_{j-1} * G_{\delta t})(x) - \frac{2\pi (\#\{d_{j-1} > 0\} - \#\{d_{j-1} < 0\} + 1)}{\int_{d_{j-1} > 0} \Delta d_{j-1}(x) dx} (\delta t) \\ & + S_{j-1}(x)(\delta t). \end{aligned}$$

2. *Construct the signed distance function  $d_j$  using*

$$d_j(x) = \mathbf{Redist}(\mathcal{L}(x)).$$

A second-order in time version of Algorithm 1 can be obtained as follows

**ALGORITHM 2. *Multi-step, second-order in time version.*** *Given the initial set  $\Sigma_0$  defined through its signed distance function  $d_0(x)$  and a time step  $\delta t > 0$ , generate the sets  $\Sigma_j$  via their signed distance function  $d_j(x)$  at the subsequent discrete times  $t = j(\delta t)$  by alternating between the following two steps:*

1. *Using  $G_t$  in (2.1), form*

$$\begin{aligned} \mathcal{L}_1(x) = & (d_{j-2} * G_{2\delta t})(x) + S_{j-2}(x)(2\delta t) \\ & - \frac{2\pi (\#\{d_{j-2} > 0\} - \#\{d_{j-2} < 0\} + 1)}{\int_{d_{j-2} > 0} \Delta d_{j-2}(x) dx} (2\delta t) \end{aligned}$$

$$\begin{aligned} \mathcal{L}_2(x) = & (d_{j-1} * G_{\delta t})(x) + S_{j-1}(x)(\delta t) \\ & - \frac{2\pi (\#\{d_{j-1} > 0\} - \#\{d_{j-1} < 0\} + 1)}{\int_{d_{j-1} > 0} \Delta d_{j-1}(x) dx} (\delta t). \end{aligned}$$

2. *Construct the signed distance function  $d_j$  using*

$$d_j(x) = \mathbf{Redist} \left( \frac{1}{3} (4\mathcal{L}_2(x) - \mathcal{L}_1(x)) \right).$$

The overall computational complexity of the general algorithm is  $O(N \log N)$ . Indeed, counting connected grid components can be performed in  $O(N)$  operations where  $N$  is the total number of grid points. The convolution is done in  $O(N \log N)$  operations using the FFT. Also there exist algorithms, such as fast marching and fast sweeping, that construct the signed distance function in  $O(N \log N)$  operations [6, 31, 39, 41, 42]. All the other terms used in the scheme, including the integral, can be done in  $O(N)$  operations. Thus, we see that the per time step cost of the complete algorithm is  $O(N \log N)$ . In addition, due to its unconditional stability, there is no restriction on

the time step size. Note that the number of reinitialization operations involved in Algorithm 2 can be kept at one per time step.

In the case of pure area preserving curvature motion (i.e., for  $S = 0$ ), in the interest of even better accuracy, we propose slightly modified versions of Algorithms 1 and 2 that consist in exactly matching, at each time step  $j$  ( $\delta t$ ), the current area with the initial area. This strategy is similar to the one used by Ruuth and Wetton [38] in the case of threshold dynamics for area preserving curvature motion. For clarity of the exposition, we define the area operator  $\mathcal{A} : C^2(\mathbb{R}^2, \mathbb{R}) \rightarrow \mathbb{R}$  applied to a smooth function  $\phi$  to be

$$\mathcal{A}(\phi) := \int_{\phi(x) > 0} dx = |\{x : \phi(x) > 0\}|,$$

which computes the area of the zero super level set of  $\phi$ . We modify Algorithms 1 and 2 to use Newton's method at each time step to find  $\lambda_j^*$  the zero of the function  $\lambda_j \mapsto a(\lambda_j) - a_0$ , where  $a_0$  is the initial area, and  $a(\lambda_j)$  the current area at time step  $j$  defined as

$$a(\lambda_j) = \mathcal{A}(d_{j-1} * G_{\delta t} - \lambda_j).$$

Our expression for the average curvature used in Algorithms 1 and 2 is taken here to be the initial guess for the Newton iteration:

$$\lambda_j^0 = \frac{2\pi (\#\{d_{j-1} > 0\} - \#\{d_{j-1} < 0\} + 1)}{\int_{d_{j-1} > 0} \Delta d_{j-1}(x) dx} (\delta t),$$

The signed distance function  $d_j$  is then constructed using

$$d_j = \mathbf{Redist}(d_{j-1} * G_{\delta t} - \lambda_j^*).$$

To modify Algorithm 2, we proceed in the same way, namely at each time step  $j$  ( $\delta t$ ), we compute  $\lambda_{j,1}^*$  and  $\lambda_{j,2}^*$  coming from the result of a Newton iteration using  $d_{j-2}$  with time step  $2\delta t$  and  $d_{j-1}$  with time step  $\delta t$  respectively. The final update in this case becomes

$$d_j = \mathbf{Redist}\left(\frac{1}{3} (4 (d_{j-1} * G_{\delta t} - \lambda_{j,2}^*) - (d_{j-2} * G_{\delta t} - \lambda_{j,1}^*))\right). \quad (3.8)$$

The more accurate version of Algorithm 1 in the case of pure area preserving curvature motion then reads as follows

**ALGORITHM 3. Area Preserving Curvature Motion.** Given the initial set  $\Sigma_0$  with area  $a_0$  defined through its signed distance function  $d_0(x)$ , a time step  $\delta t > 0$  and a tolerance  $\eta > 0$ , generate the sets  $\Sigma_j$  via their signed distance function  $d_j(x)$  at the subsequent discrete times  $t = j(\delta t)$  by alternating between the following two steps:

1. Using Newton's method with initial guess

$$\lambda_j^0 = \frac{2\pi (\#\{d_{j-1} > 0\} - \#\{d_{j-1} < 0\} + 1)}{\int_{d_{j-1} > 0} \Delta d_{j-1}(x) dx} (\delta t),$$

find  $\lambda_j^*$  such that  $|\mathcal{A}(d_{j-1} * G_{\delta t} - \lambda_j^*) - a_0| < \eta$  and form

$$\mathcal{L}(x) = (d_{j-1} * G_{\delta t})(x) - \lambda_j^*.$$

2. Construct the signed distance function  $d_j$  using

$$d_j(x) = \mathbf{Redist}(\mathcal{L}(x)).$$

The multi-step version of Algorithm 3 can be obtained by following the same pattern as Algorithm 2 and using the update described in equation (3.8).

**ALGORITHM 4. Area Preserving Curvature Motion: second-order in time version.** Given  $\Sigma_0$  with area  $a_0$  having distance function  $d_0(x)$ ,  $\delta t > 0$  and  $\eta > 0$ , generate  $\Sigma_j$  at times  $t = j(\delta t)$  by alternating between the following two steps:

1. Using Newton's method with initial guesses

$$\lambda_{j,1}^0 = \frac{2\pi (\#\{d_{j-2} > 0\} - \#\{d_{j-2} < 0\} + 1)}{\int_{d_{j-2} > 0} \Delta d_{j-2}(x) dx} (\delta t)$$

and

$$\lambda_{j,2}^0 = \frac{2\pi (\#\{d_{j-1} > 0\} - \#\{d_{j-1} < 0\} + 1)}{\int_{d_{j-1} > 0} \Delta d_{j-1}(x) dx} (\delta t),$$

find  $\lambda_{j,1}^*$  and  $\lambda_{j,2}^*$  s.t.  $|\mathcal{A}(d_{j-2} * G_{\delta t} - \lambda_{j,1}^*) - a_0| < \eta$  and  $|\mathcal{A}(d_{j-1} * G_{\delta t} - \lambda_{j,2}^*) - a_0| < \eta$ , and form

$$\mathcal{L}_1(x) = (d_{j-2} * G_{\delta t})(x) - \lambda_{j,1}^*$$

$$\mathcal{L}_2(x) = (d_{j-1} * G_{\delta t})(x) - \lambda_{j,2}^*.$$

2. Construct the signed distance function  $d_j$  using

$$d_j(x) = \mathbf{Redist}\left(\frac{1}{3}(4\mathcal{L}_2(x) - \mathcal{L}_1(x))\right).$$

Note that although Algorithms 3 and 4 preserve the area more accurately than Algorithms 1 and 2, they will be inapplicable in situations where the total area may change due to the presence of a non-zero forcing term  $S$ .



**3.3. Numerical convergence study.** In this section we describe some convergence studies done with the algorithms for area preserving curvature motion introduced in the previous section. In the computations presented below, we used a second order accurate procedure to construct the signed distance function in a tubular neighborhood of the interface. For details on more sophisticated algorithms for constructing signed distance functions, we refer the reader to [5, 6, 31, 39, 41, 44].

We start by investigating the convergence of Algorithms 1 and 2 in the case  $S = 0$ , namely for pure area preserving curvature motion. The results are displayed in Tables 3.1 and 3.2. The initial condition for these tests is an ellipse  $\mathcal{E}$  with major axis  $a = 0.45$  and minor axis  $b = 0.2$ , and the computational domain is  $[0, 1]^2$ . The evolution is computed over the time interval  $[0, 0.01]$ . The major and minor axes of the final curve (which is no longer an ellipse) are 0.4 and 0.22 respectively. At the final time  $T = 0.01$  we measure the quantity  $\int_{\Sigma(T)} (x^2 + y^2) d\Omega$ , where  $\Sigma(t) \in \mathbb{R}^2$  evolves under area preserving curvature motion with initial condition  $\Sigma(0) = \mathcal{E}$ . We compare this quantity with the exact quantity  $\int_{\Sigma_e(T)} (x^2 + y^2) d\Omega$ , where the exact evolution  $\Sigma_e(t)$  is computed using a front tracking technique with a very fine discretization of the parameterized curve. We also display the error in area and its associated convergence rate.

Resolution	# of Time Steps	Relative error in $\int_{\Sigma(T)} (x^2 + y^2) d\Omega$ (in %)	Order	Error in area	Order
$33 \times 33$	20	0.0167	–	0.000921	–
$65 \times 65$	40	0.3218	–4.27	0.000561	0.72
$129 \times 129$	80	0.2046	0.65	0.000271	1.05
$257 \times 257$	160	0.1002	1.03	0.00012	1.18
$513 \times 513$	320	0.0487	1.04	0.000055	1.12
$1025 \times 1025$	640	0.0240	1.02	0.0000263	1.06
$2049 \times 2049$	1280	0.0119	1.01	0.0000128	1.03

TABLE 3.1

*Convergence of Algorithm 1 for  $S = 0$ . The initial condition is an ellipse with major axis  $a = 0.45$  and minor axis  $b = 0.2$  on  $[0, 1]^2$ . The evolution was computed for  $t \in [0, 0.01]$ .*

Resolution	# of Time Steps	Relative error in $\int_{\Sigma(T)} (x^2 + y^2) d\Omega$ (in %)	Order	Error in area	Order
$33 \times 33$	20	0.8098	–	0.00164	–
$65 \times 65$	40	0.4919	0.72	0.000710	1.20
$129 \times 129$	80	0.0951	2.37	0.000150	2.24
$257 \times 257$	160	0.00738	3.69	0.0000210	2.84
$513 \times 513$	320	0.000928	2.99	0.00000434	2.27
$1025 \times 1025$	640	0.000196	2.25	0.00000109	1.99
$2049 \times 2049$	1280	0.00000915	4.42	0.000000265	2.04

TABLE 3.2

*Convergence of Algorithm 2 for  $S = 0$ . The initial condition is an ellipse with major axis  $a = 0.45$  and minor axis  $b = 0.2$  on  $[0, 1]^2$ . The evolution was computed for  $t \in [0, 0.01]$ .*

As can be seen in Table 3.1, Algorithm 1 settles into a clearly first order convergence rate. Table 3.2 shows the convergence rate for Algorithm 2, which on this example turns out to be significantly higher than second order, perhaps due to some

cancellation of errors. In any case, Algorithm 2 achieves very high accuracy even on very modest sized grids.

We now present the convergence of Algorithms 3 and 4 with the same initial condition (same ellipse). The results are displayed in Tables 3.3 and 3.4 respectively. We repeat that these algorithms can only be used to simulate pure area preserving motion by curvature. In both convergence tests, the area was preserved up to an error of  $10^{-14}$  or less. Table 3.3 displays the convergence of Algorithm 3 and Table 3.4 the convergence of Algorithm 4.

Resolution	# of Time Steps	Relative error in $\int_{\Sigma(T)} (x^2 + y^2) d\Omega$ (in %)	Order
$33 \times 33$	20	0.6256	–
$65 \times 65$	40	0.0588	3.41
$129 \times 129$	80	0.0213	1.47
$257 \times 257$	160	0.0194	0.13
$513 \times 513$	320	0.0117	0.73
$1025 \times 1025$	640	0.00631	0.89
$2049 \times 2049$	1280	0.00327	0.95

TABLE 3.3

*Convergence of Algorithm 3. The initial condition is an ellipse with major axis  $a = 0.45$  and minor axis  $b = 0.2$  on  $[0, 1]^2$ . The evolution was computed for  $t \in [0, 0.01]$ .*

Resolution	# of Time Steps	Relative error in $\int_{\Sigma(T)} (x^2 + y^2) d\Omega$ (in %)	Order
$33 \times 33$	20	0.3040	–
$65 \times 65$	40	0.0112	4.76
$129 \times 129$	80	0.00735	0.61
$257 \times 257$	160	0.00689	0.09
$513 \times 513$	320	0.00197	1.81
$1025 \times 1025$	640	0.000488	2.01
$2049 \times 2049$	1280	0.000117	2.06

TABLE 3.4

*Convergence of Algorithm 4. The initial condition is an ellipse with major axis  $a = 0.45$  and minor axis  $b = 0.2$  on  $[0, 1]^2$ . The evolution was computed for  $t \in [0, 0.01]$ .*

In addition to the convergence studies described above, we tested our area preserving curvature motion algorithm on an initial configuration containing three circles with radii 0.15, 0.2 and 0.22 on  $[0, 1]^2$ . Since circles remain circles under this motion, we monitored the evolution of each of the three radii for  $t \in [0, 0.1]$ , and compared it with the exact evolution obtained by numerical integration of the coupled ODEs for the radii. The circles were placed far apart initially so that no collision occurred during the evolution. The result is shown in Figure 3.2. Figure 3.1 shows the three circles in the initial condition (thick line) and the final curves at time  $t = 0.1$  (fine line). Note that area preserving curvature motion is global and hence couples the evolving circles to each other.

Evolution of three circles under area preserving curvature motion

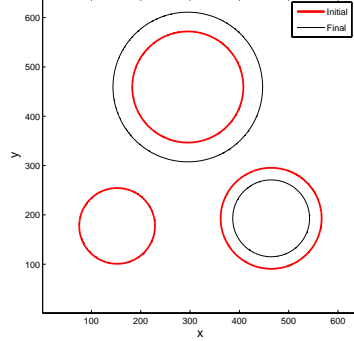


FIGURE 3.1. Evolution of three circles under area preserving curvature motion for  $t \in [0, 0.1]$ . The initial condition is shown by the thick curves, and the final curves at  $t = 0.1$  are displayed by the thin curves. In this configuration, the largest circle grew while the other two shrunk (the smallest one actually disappeared).

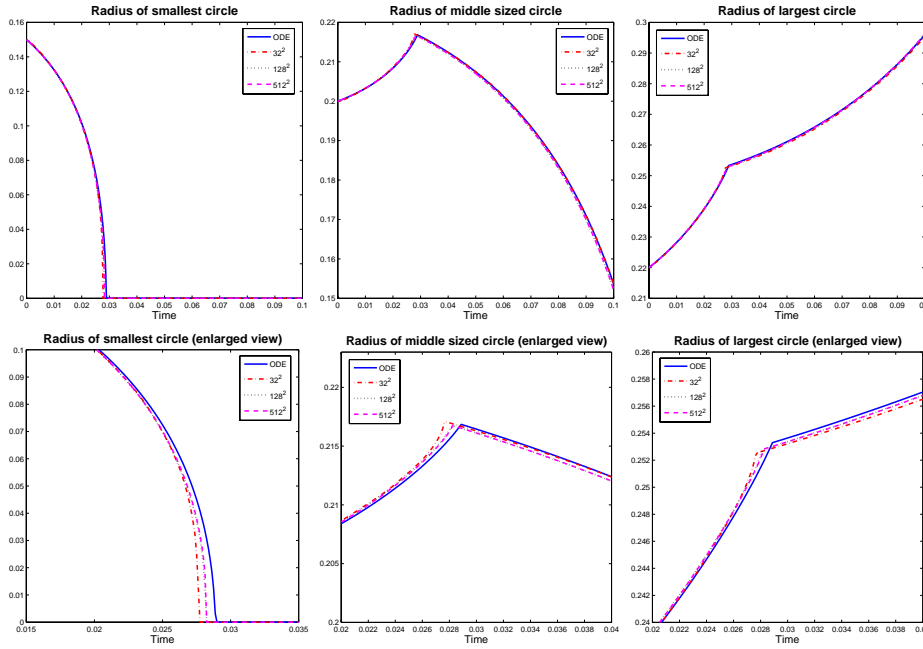


FIGURE 3.2. Comparison between the evolutions of the exact and computed radii obtained from three initial circles taken as initial condition and evolved under area preserving curvature motion. The initial three circles have the following radii: 0.15, 0.2 and 0.22. In each of the plot, we superimpose the exact evolution (in bold) and the evolution of the radii computed on a  $32^2$ ,  $128^2$  and  $512^2$  grid. The top row shows the entire evolution, while the bottom row shows an enlarged view of the plots where the graph is not differentiable (corresponding to the disappearance of the smallest circle). This computation was performed with Algorithm 4.

**4. Application: large scale simulations of area preserving curvature motion.** In this section, we demonstrate the capacity of our proposed algorithms to handle large scale simulations with very good accuracy.

**4.1. Curve shortening at various area fractions.** Geometrically in two dimensions, the area preserving curvature flow describes the shortening of a curve (or interface) separating two phases, while maintaining the area of each phase equal to their respective initial area. A natural question therefore arises: at what rate does the total length of the curve decrease? Scaling arguments [7] suggest that the total length  $L$  decreases as a power law in time according to

$$L(t) \sim t^{-\frac{1}{2}}.$$

In a recent paper, Dai [7] obtained a rigorous result for the rate of decrease of  $L$  in the case of a dilute mixture. Specifically, he showed that for a collection of non-intersecting and convex plane curves, the total length  $L(t)$  cannot decrease faster than  $t^{-\frac{1}{2}}$  in a time average sense. In this simplified case, there is no coalescence and the only singularity is the disappearance of curves. In the general case however, collisions of curves will occur causing singularities in the curvature to appear at the times of first intersections. In fact, at the points of intersections, the curvature will be infinite, leading to an immediate smoothing and a fast decay of the sum of the lengths pertaining to the merging curves. Figure 4.1 illustrates this point on a simple example of two curves intersecting each other during their evolution under area preserving curvature motion.

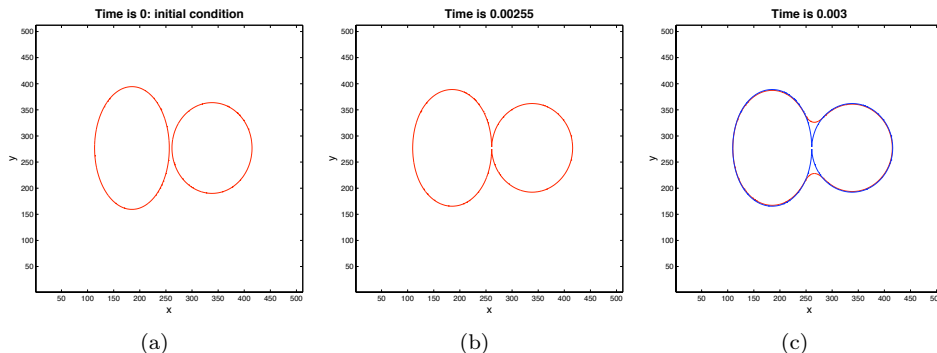


FIGURE 4.1. Coalescence of two curves during their evolution under area preserving curvature motion. Figure 4.1(a) shows the initial condition and Figure 4.1(b) displays the new curve just as the two previous ones collided. The curvature of the new curve is very large at the point of intersection. Figure 4.1(c) shows the new curve shortly after the collision occurred superimposed with the curve shown in Figure 4.1(b).

In this context, we refer to the phase enclosed by the curves as Phase 1. Phase 2 denotes its complement. As a demonstration of the proposed algorithms, we present some simulations of area preserving curvature motion on very large collections of closed curves (or droplets). We also measure certain statistics related to the configurations of droplets during their evolution. Our algorithms achieve good accuracy with droplets as small as 15 pixels in diameter, allowing us to carry out simulations with very large number of initial droplets. Indeed, we present simulations with initial conditions containing up to 25000 droplets on a  $4097^2$  grid. Our computations con-

sidered various area fractions of Phase 1 ranging from 10% to 50%, which equivalently considered area fractions of Phase 2 ranging from 50% to 90%.

**4.2. Numerical results.** In this section, we present the results of our simulations. We construct the initial data by generating random sets of points from a uniform distribution, and placing a disk centered at each of the points with radius  $r_d$  chosen from a uniform distribution. The droplets are obtained by taking the union of the disks (some of which may be overlapping). From such initial configurations (i.e., randomly generated droplet configurations), it would be reasonable to expect a certain collision rate during the evolution under area preserving curvature motion. Figure 4.2 illustrates the area preserving motion by curvature evolution on initial data with 10% and 40% area fraction. To avoid boundary effects, the computations are done on a slightly larger domain than  $[0, 1]^2$ . Additionally, as an extra precaution to prevent premature mergings of nearby droplets, at each time step we divide the droplets into subsets containing only droplets that are far enough apart, and update the signed distance function of each subset separately. This allows individual grains to evolve independently all the way up to the time of overlap with another evolving droplet that may be as close as a single grid point. Finally, since the average size of the droplets increases during the evolution (slowing down the dynamics), we perform our computations using an adaptive time step regulated by the average size of the droplets. This adaptivity is made possible by the unconditional stability of our algorithms. In Figure 4.3, we demonstrate that taking larger time steps once the average grain size is large enough does not significantly change the configurations of droplets.

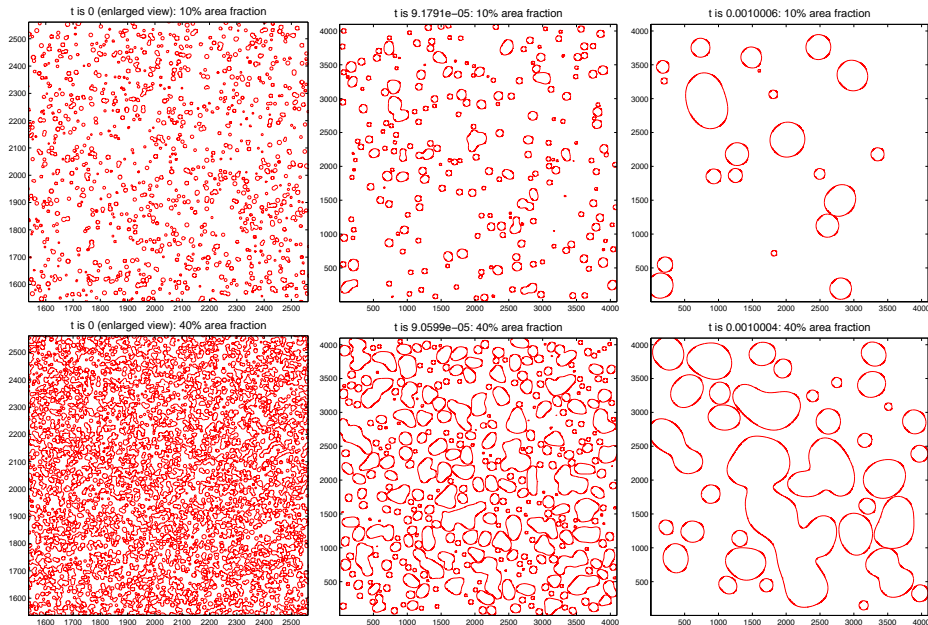


FIGURE 4.2. Evolution of two configurations of droplets under area preserving curvature motion. The top row displays the evolution of a configuration of droplets with 10% area fraction. The bottom row shows the evolution for a configuration with 40% area fraction. Because of the very large number of droplets in the early configurations, we only show a subset of these configurations in the two plots of the first column. These subsets have been enlarged for a better view.

Figure 4.4 compares the rate of decrease of the total length  $L$  (which is also

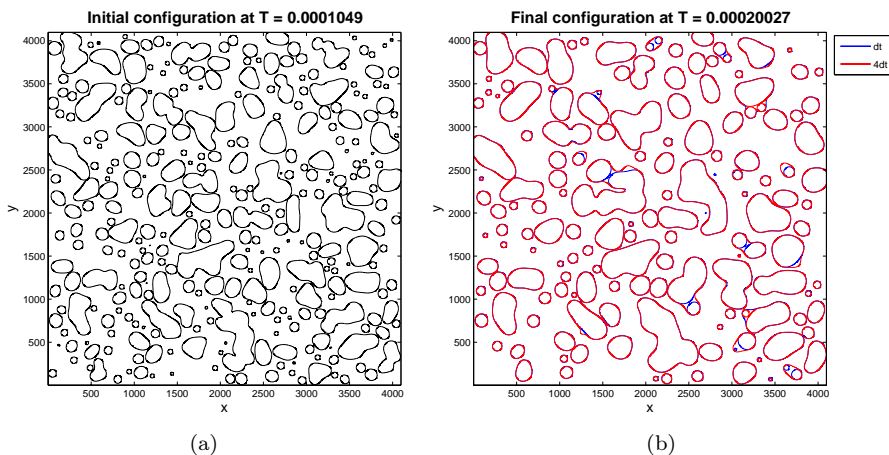


FIGURE 4.3. Comparison of the effect of two different time step sizes on the computed solution: fixed time step size of  $\delta t$ , and the adaptive time step size that reaches  $4\delta t$  at the initial condition shown on the left Figure 4.3(a). In this example,  $\delta t = 4.7710^{-7}$ . Figure 4.3(a) shows the initial configuration and Figure 4.3(b) displays the configurations obtained from computations using  $\delta t$  and  $4\delta t$ . Except for minor differences, both computations are able to resolve the dynamics, thus exonerating the use of larger time steps as the coarsening proceeds.

the energy dissipated by the evolution) with the theoretical bound  $t^{-\frac{1}{2}}$  for various area fractions. Note that this bound has been rigorously established [7] for convex curves that never collide. Moreover, it is known [8] that it is possible to construct configurations of droplets that coarsen arbitrarily fast under this geometric motion. Nevertheless, for randomly generated initial conditions, the rate of decrease of  $L$ , obtained from our computations, closely follows the theoretical bound despite frequent collisions between droplets.

Another quantity that we study is the number of connected components  $K$  in the configurations of droplets. Based on the rate of decay of  $L$ , and using a simple heuristic argument on a uniform configuration of disks, we can show that the number of connected components should essentially decay as  $\frac{1}{t}$ . In Figure 4.5 we compare the numerically observed rate of decrease of  $K$  for various area fractions with the bound  $\frac{1}{t}$ . For the randomly generated initial conditions chosen in our simulations, the plots show very good agreement between the computed rate and the bound  $\frac{1}{t}$ . Intuition also suggests that at any given time, a certain population of droplets will be just about to collide, generating configurations at which the energy decrease rate must be elevated (the proportion of droplets just about to collide to all droplets would of course depend on the area fraction). However, it appears that even if collisions between droplets cause a deviation in the coarsening rate of randomly generated initial configurations, our numerical experiments, despite their large size, are still not large enough to discern such a difference – perhaps the effect is very small. Nevertheless, as one would expect, we observe that *at any time* during the evolution (outside of the transient initial period and the final stage where only a few droplets remain), there is a constant proportion of eccentric droplets (i.e., droplets that are the result of a recent collision). This observation agrees with the expectation that collisions occur at a definite rate in proportion to the number of droplets. To exhibit this behavior,

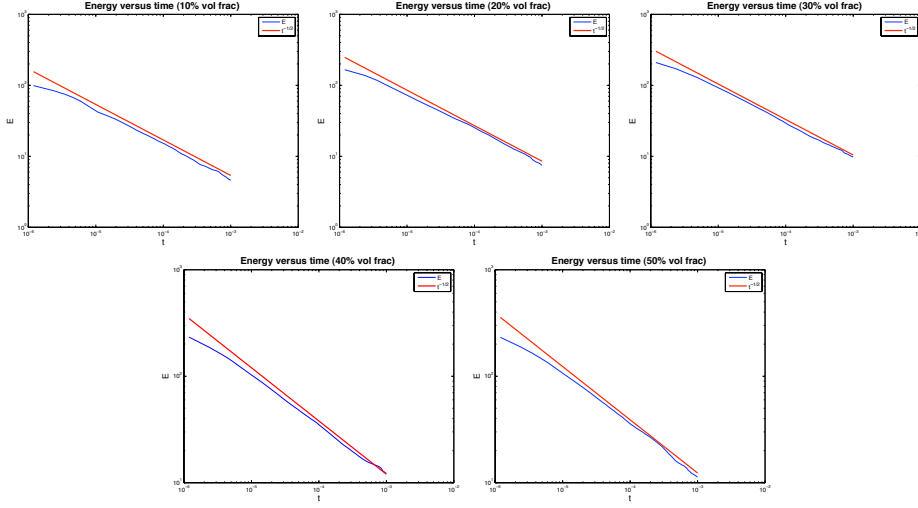


FIGURE 4.4. Loglog plot of the total length  $L$  of the boundary of all droplets (also the energy being decreased by the evolution) versus time for various area fractions. The thick line corresponds to the theoretical bound  $t^{-\frac{1}{2}}$ . From top to bottom, and left to right, the plots correspond to area fractions ranging from 10% to 50%. The plots acknowledge good agreement between the theoretical bound  $t^{-\frac{1}{2}}$  and the numerically observed rates (fine line).

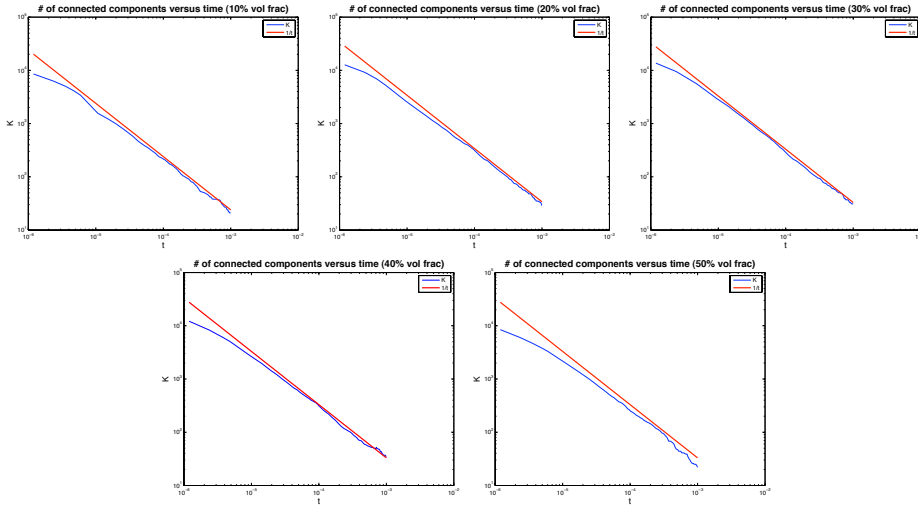


FIGURE 4.5. Loglog plot of the number of connected components  $K$  versus time for various area fractions. From top to bottom, and left to right, the plots correspond to area fractions ranging from 10% to 50%. The plots acknowledge good agreement between the bound  $\frac{1}{t}$  (thick line) and the numerically observed rates (fine line).

we measure the isoperimetric ratio

$$I(C) := \frac{P^2}{A} \quad (4.1)$$

of each grain in order to characterize their shape. In (4.1),  $C$  is a closed curve,  $P$  is its perimeter and  $A$  its area. Since the isoperimetric ratio is minimized by a circle,

we have that for any closed curve

$$I \geq 4\pi \sim 12.57,$$

where  $I = 4\pi$  when  $C$  is a circle. For an ellipse with minor axis  $b$  and major axis  $a = 3b$ , the isoperimetric ratio is approximately  $I \sim 18.95$ . For a more elongated ellipse with minor axis  $b$  and major axis  $a = 4b$  the isoperimetric ratio is approximately  $I \sim 23.42$ . From these references, we look at the proportion of droplets with isoperimetric ratio  $I > 20$ . Figure 4.6 shows the distribution of isoperimetric ratios at different times throughout the evolution for a configuration of droplets with 50% area fraction. Table 4.1 displays the proportion of eccentric droplets for the various area fractions studied in our computations. Figure 4.7 shows the proportion of eccentric droplets and the total number of droplets for configurations with area fractions 30%, 40% and 50%. In each case, the proportion of eccentric droplets decreases very quickly during the transient initial phase (as is the total number of droplets) and then stabilizes itself around a constant value. The proportion of eccentric droplets remain constant until the total number of droplets becomes too small.

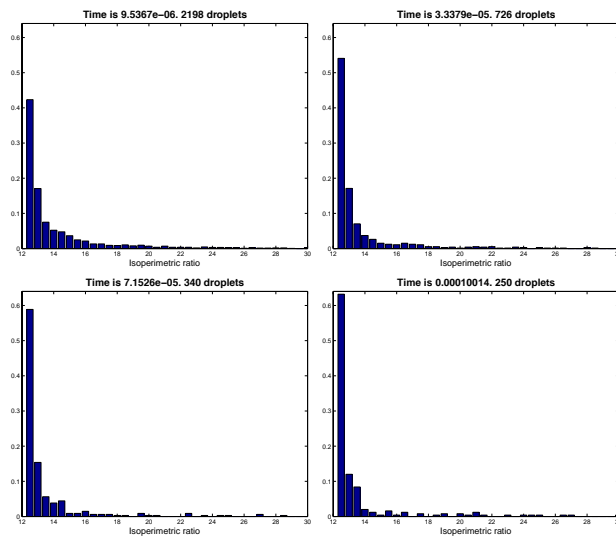


FIGURE 4.6. *Time evolution of the distribution of isoperimetric ratios computed from the evolution of an initial configuration of droplets with 50% area fraction under area preserving curvature motion. During the evolution, the distribution of isoperimetric ratios remains quite wide. This width underlines the existence, at all times, of a certain proportion of droplets that resulted from collisions.*



Area fraction	Proportion of eccentric droplets	# of droplets at onset of constant proportion	# of droplets at the end of constant proportion
10%	0.12%	5054	193
20%	0.46%	2623	78
30%	1.18%	2932	91
40%	2.49%	2766	34
50%	3.81%	3885	49

TABLE 4.1

Proportion of eccentric droplets (the isoperimetric ratio of which satisfies  $I > 20$ ) for various area fractions. The proportion of eccentric droplets increases with the area fraction, as one would expect.

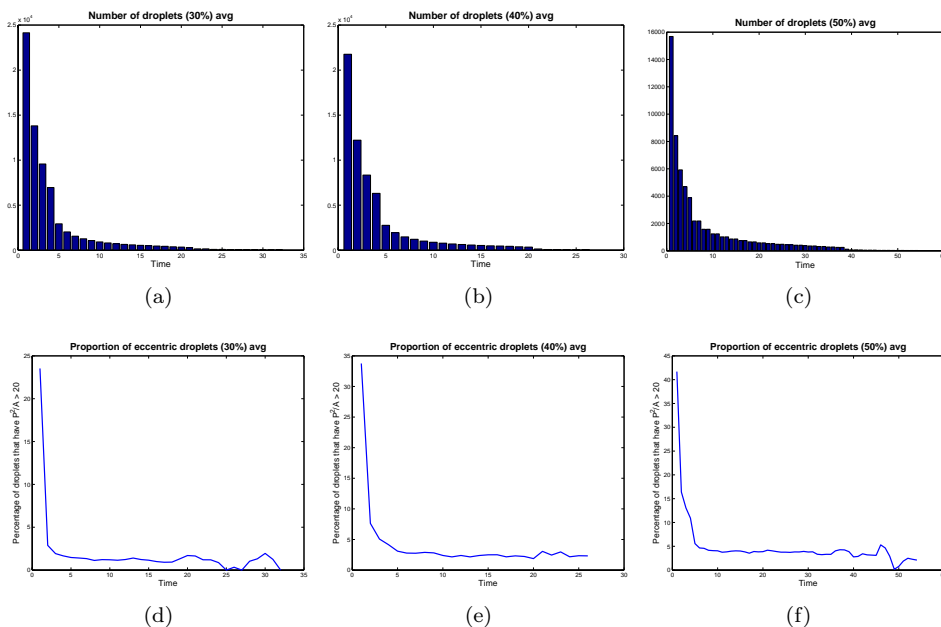


FIGURE 4.7. Proportions of eccentric droplets (i.e. with isoperimetric ratio  $I > 20$ ) in the case of 30%, 40% and 50% area fraction. Figures 4.7(d), 4.7(e) and 4.7(f) clearly show that after a transient initial time, the proportion of eccentric droplets seems to stabilize around a constant value. Figures 4.7(a), 4.7(b) and 4.7(c) corroborate the fact the proportion of eccentric droplets remain constant from the time when the total number of droplets is on average 3000 until the time when it is around 50.

**5. Conclusion.** We described efficient and highly accurate algorithms for simulating certain area preserving curvature flows in the plane, with normal speeds of the form  $v_N = \kappa - \bar{\kappa} + S$ . Our schemes are based on a diffusion generated motion approach using signed distance functions, thus making them more accurate than the standard MBO (threshold dynamics) schemes on uniform grids. We proposed first-order and second-order in time versions of our algorithms and carried out numerical studies to verify their convergence and accuracy. These schemes are also relevant for image processing applications where the additional term  $S$  in the velocity  $\kappa - \bar{\kappa} + S$  comes from data fitting terms in variational models.

In addition, we presented an application of these new schemes to large scale

simulations of coarsening. This application demonstrated the ability of our algorithms to handle large scale computations due to their high accuracy and computational efficiency. In particular, we were able to simulate the coarsening of large numbers of droplets under area preserving curvature motion. The large number of droplets allowed us to obtain reliable statistical measurements, which are reported.

**Acknowledgments.** Catherine Kublik and Selim Esedoğlu were supported by NSF grants DMS-0748333 and DMS-0713767. Selim Esedoğlu was also supported by an Alfred P. Sloan Foundation fellowship. Jeffrey Fessler was supported by NIH grant R01 HL 098686.

#### REFERENCES

- [1] L. Alvarez, F. Guichard, P.-L. Lions, and J.-M. Morel. Axioms and fundamental equations of image processing. *Arch. Rational Mech. Anal.*, 123:199–257, 1993.
- [2] G. Barles and C. Georgelin. A simple proof of convergence for an approximation scheme for computing motions by mean curvature. *SIAM Journal on Numerical Analysis*, 32(2):484–500, 1995.
- [3] L. Bronsard and B. Stoth. Volume preserving mean curvature flow as a limit of a nonlocal Ginzburg-Landau equation. *SIAM J. on Mathematical Analysis*, 28(4):769–807, 1997.
- [4] L. Bronsard and B. Wetton. A numerical method for tracking curve networks moving with curvature motion. *J. Comput. Phys.*, 120(1):66–87, 1993.
- [5] A. Chambolle and M. Novaga. Approximation of the anisotropic mean curvature flow. *Mathematical Models and Methods in Applied Sciences*, 17(6):833–844, 2007.
- [6] L.-T. Cheng and Y.-H. Tsai. Redistancing by flow time dependent Eikonal equation. *J. Comput. Phys.*, 227(2):4002–4017, 2008.
- [7] S. Dai. On the shortening rate of collections of plane convex curves by the area-preserving mean curvature flow. *SIAM J. Math. Anal.*, 42(1):323–333, 2010.
- [8] S. Dai, B. Niethammer, and R. Pego. Crossover in coarsening rates for the monopole approximation of the Mullins-Sekerka model with kinetic drag. *Proc. Roy. Soc. Edin.: A*, 140:553–571, 2010.
- [9] M. C. Delfour and J.-P. Zolesio. Shapes and geometries. analysis, differential calculus and optimization. *Advances in Design and Control*, SIAM, 2001.
- [10] I. C. Dolcetta, S. F. Vita, and R. March. Area preserving curve shortening flows: from phase separation to image processing. *Interfaces and Free Boundaries*, 4:325–343, 2002.
- [11] S. Esedoğlu, S. Ruuth, and R. Tsai. Diffusion generated motion using signed distance functions. *Journal of Computational Physics*, 229(4):1017–1042, 2010.
- [12] S. Esedoğlu, S. J. Ruuth, and R. Tsai. Threshold dynamics for high order geometric motions. *Interfaces and Free Boundaries*, 10(3):263–282, 2008.
- [13] L. Evans. Convergence of an algorithm for mean curvature motion. *Indiana Univ. Math. J.*, 42:553–557, 1993.
- [14] H. Federer. Curvature measures. *Transactions of the American Mathematical Society*, 93:418–491, 1959.
- [15] M. Gage. Curve shortening makes convex curves circular. *Invent. Math.*, 76:357–364, 1984.
- [16] M. Gage. On an area-preserving evolution equation for plane curves. *Contemp. Math.*, 51:51–62, 1986.
- [17] M. Gage and R. Hamilton. The heat equation shrinking convex plane curves. *J. Differential Geom.*, 23(1):69–96, 1986.
- [18] D. Golovaty. The volume preserving motion by mean curvature as an asymptotic limit of reaction-diffusion equations. *Q. of Appl. Math.*, LV:243–298, 1997.
- [19] G. Huisken. Flow by mean curvature of convex surfaces into spheres. *J. Differential Geom.*, 20(1):237–266, 1984.
- [20] I. M. Lifshitz and V. V. Slyozov. The kinetics of precipitation from supersaturated solid solutions. *J. Phys. Chem. Solids*, 19:35–50, 1961.
- [21] P. Mascarenhas. Diffusion generated motion by mean curvature. Technical report, UCLA, July 1992. CAM Report 92-33.
- [22] B. Merriman, J. K. Bence, and S. Osher. Diffusion generated motion by mean curvature. Technical report, UCLA, April 1992. CAM Report 92-18.
- [23] B. Merriman, J. K. Bence, and S. Osher. Diffusion generated motion by mean curvature. In

- AMS, editor, *Proceedings of the Computational Crystal Growers Workshop*, pages 73–83, Providence, RI, 1992.
- [24] B. Merriman, J. K. Bence, and S. Osher. Motion of multiple junctions: a level set approach. *Journal of Computational Physics*, 112(2):334–363, 1994.
- [25] W. W. Mullins. Two-dimensional motion of idealized grain boundaries. *J. Appl. Phys.*, 27:900–904, 1956.
- [26] B. Niethammer. The mathematics of Ostwald ripening. In S. Hildebrandt and H. Karcher, editors, *Geometric Analysis and Nonlinear Differential Equations*, pages 649–663. Springer, 2002.
- [27] R. H. Nochetto, M. Paolini, and C. Verdi. A dynamic mesh algorithm for curvature dependent evolving interfaces. *J. Comput. Phys.*, 123(2):296–310, 1996.
- [28] S. Osher and J. A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations. *J. Comp. Phys.*, 79:12–49, 1988.
- [29] D. Peng, B. Merriman, S. Osher, H.-K. Zhao, and M. Kang. A pde-based fast local level set method. *J. Comput. Phys.*, 155(2):410–438, 1999.
- [30] J. Rubinstein and P. Sternberg. Nonlocal reaction-diffusion equations and nucleation. *IMA J. Appl. Math.*, 48:248–264, 1992.
- [31] G. Russo and P. Smereka. A remark on computing distance functions. *J. Comput. Phys.*, 163:51–67, 2000.
- [32] S. J. Ruuth. A diffusion generated approach to multiphase motion. *J. Comput. Phys.*, 145(1):166–192, 1998.
- [33] S. J. Ruuth. Efficient algorithms for diffusion-generated motion by mean curvature. *J. Comput. Phys.*, 144(2):603–625, 1998.
- [34] S. J. Ruuth and B. Merriman. Convolution generated motion and generalized Huygens’ principles for interface motion. *SIAM J. Appl. Math.*, 60(33):868–890, 2000.
- [35] S. J. Ruuth and B. Merriman. Convolution-thresholding methods for interface motion. *J. Comput. Phys.*, 169(2):678–707, 2001.
- [36] S. J. Ruuth, B. Merriman, and S. Osher. Convolution generated motion as a link between cellular automata and continuum pattern dynamics. *J. Comput. Phys.*, 151:836–861, 1999.
- [37] S. J. Ruuth, B. Merriman, J. Xin, and S. Osher. Diffusion-generated motion by mean curvature for filaments. *J. Nonlinear Science*, 11(6):473–493, 2001.
- [38] S. J. Ruuth and B. T. R. Wetton. A simple scheme for volume-preserving motion by mean curvature. *Journal of Scientific Computing*, 19(1-3):373–384, 2003.
- [39] J. Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.
- [40] P. Smereka. Semi-implicit level set methods for curvature and surface diffusion motion. *Journal of Scientific Computing*, 19(1-3):439–456, 2003.
- [41] Y.-H. Tsai, L. Cheng, S. Osher, and H.-K. Zhao. Fast sweeping methods for a class of hamilton-jacobi equations. *SIAM Journal on Numerical Analysis*, 41(2):673–694, 2003.
- [42] J. Tsitsiklis. Efficient algorithms for globally optimal trajectories. *IEEE Transactions on Automatic Control*, 40:1528–1538, 1995.
- [43] C. Wagner. Theorie der Alterung von Niederschlägen durch Umlösen. *Z. Elektrochemie*, 65:581–594, 1961.
- [44] H. Zhao. A fast sweeping method for Eikonal equations. *Mathematics of Computation*, 74(250):603–627, 2005.
- [45] H.-K. Zhao, B. Merrimann, S. Osher, and L. Wang. Capturing the behavior of bubbles and drops using the variational level set approach. *J. Comput. Phys.*, 143:495–518, 1998.