

Convolution Generated Motion of Interfaces

Steven Ruuth (SFU)

IPAM

May, 2001

- Introduction to interface motion
- Diffusion-generated motion
- Convolution generated motion
- Connections to other methods
 - Reaction-diffusion (phase field)
 - Cellular automata
- Conclusions

Introduction

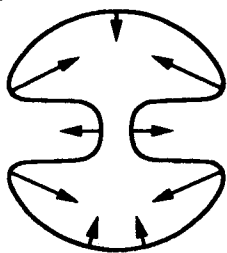
There are many natural phenomena in which sharp interfaces form, persist and propagate...

- Interface has energy/tension
⇒ will decrease (or increase) in area in order to lower energy
- Idealized models:
surface moves with a curvature dependent normal velocity

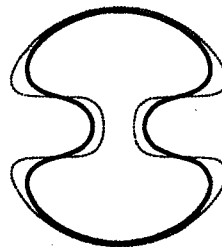
Fundamental Example: Curvature Motion

In a variety of applications, one wants to track the motion of a front that moves with a curvature-dependent speed.

For example, consider a curve, each point of which moves with a speed equal to its curvature. We want to compute the evolution of such a front.



Initial Motion



Later Region (Solid)

Figure 1: A Region Evolving According to Curvature Motion

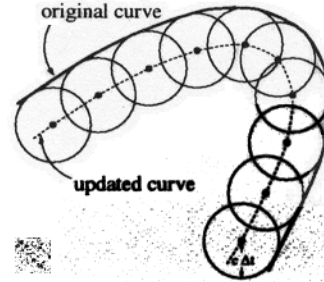
In higher dimensions, we consider a normal velocity equal to the mean curvature: "Motion by mean curvature"

This type of motion also causes the most curved parts of a surface to move most quickly.

Introduction

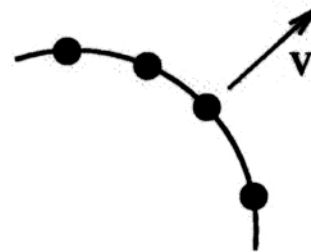
Several methods have been developed to study interface motion:

- Huygens' principle: $\vec{v} = c\hat{n}$.

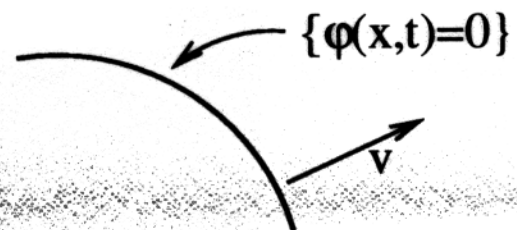


- Reaction-diffusion equations: $u_t = \Delta u - R(u)$.
(Phase field, Ginzburg-Landau)

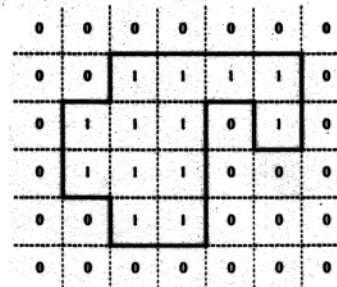
- Front tracking



- Level set method



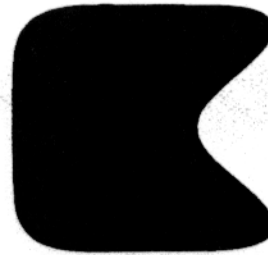
- Cellular automata



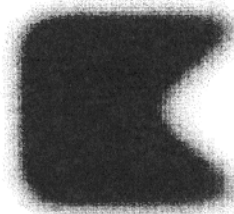
- Diffusion-generated motion by mean curvature
(Merriman, Bence, Osher 1992)

Intuition for Diffusion-Generated Motion

Allow a set of points
to “Diffuse”



Sharp corners rapidly
smooth out



IDEA: Diffusion moves the boundary of a set in a curvature-dependent way...

Intuition for Diffusion-Generated Motion

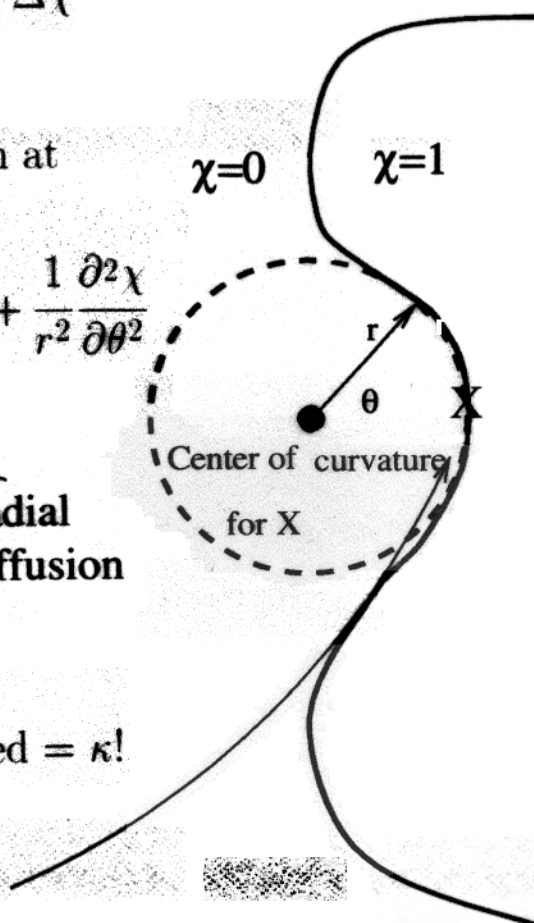
Set χ equal to the characteristic function for the initial region

Apply "diffusion" to χ $\frac{\partial \chi}{\partial t} = \Delta \chi$

Use local polar coords with origin at the center of curvature:

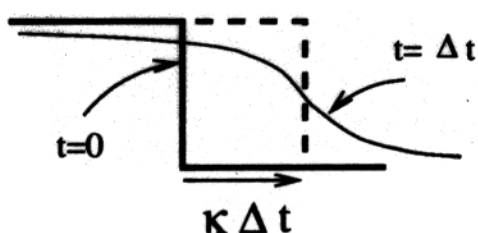
$$\frac{\partial \chi}{\partial t} = \frac{1}{r} \frac{\partial \chi}{\partial r} + \frac{\partial^2 \chi}{\partial r^2} + \frac{1}{r^2} \frac{\partial^2 \chi}{\partial \theta^2}$$

radial advection speed $1/r = \kappa$ radial diffusion



⇒ Advection-diffusion equation!
 ⇒ Level set $1/2$ moves with speed $= \kappa!$

Radial view of time evolution



Initially, the level set $\chi=1/2$ moves exactly with speed $\kappa=1/r$

Intuition for Diffusion-Generated Motion

Set χ equal to the characteristic function for the initial region.

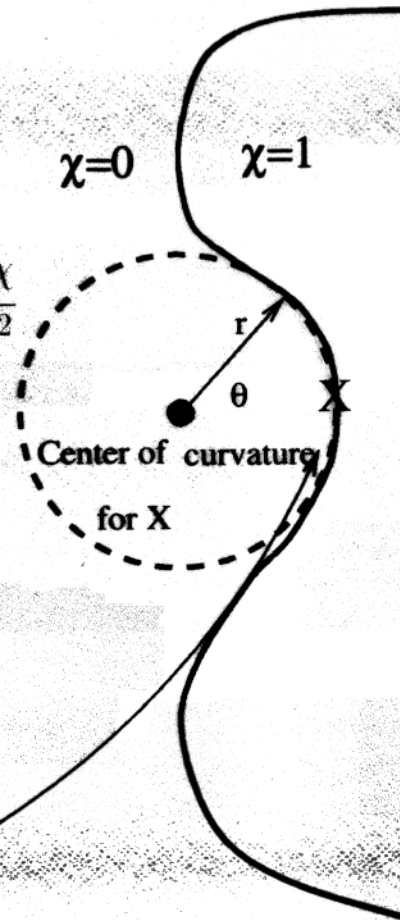
Apply "diffusion" to χ : $\frac{\partial \chi}{\partial t} = \Delta \chi$

Use local polar coords with origin at the center of curvature:

$$\frac{\partial \chi}{\partial t} = \frac{1}{r} \frac{\partial \chi}{\partial r} + \frac{\partial^2 \chi}{\partial r^2} + \frac{1}{r^2} \frac{\partial^2 \chi}{\partial \theta^2}$$

radial advection
speed $1/r = \kappa$

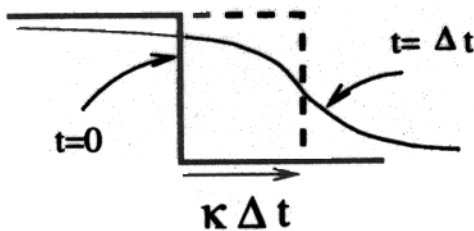
radial
diffusion



⇒ Advection-diffusion equation!

⇒ Level set $1/2$ moves with speed $= \kappa$!

Radial view of time evolution



Initially, the level set $\chi=1/2$ moves exactly with speed $\kappa=1/r$

Diffusion-Generated Motion by Mean Curvature

The previous intuition can be used to construct an algorithm for moving an interface by mean curvature (Merriman, Bence, Osher 1992):

1. χ is set to the characteristic function for the region.



$$\chi(\vec{x}, 0) = \begin{cases} 1 & \text{if } \vec{x} \text{ belongs to the initial region} \\ 0 & \text{otherwise.} \end{cases}$$

2. Diffusion is applied for a time,



$$\text{Find } \chi(\vec{x}, \Delta t) \text{ using } \chi_t = \Delta \chi.$$

3. The function, χ , is "Sharpened."



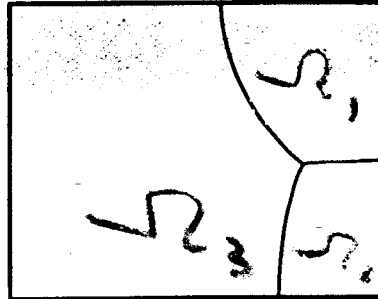
$$\chi(\vec{x}, \Delta t) = \begin{cases} 1 & \text{if } \chi(\vec{x}, \Delta t) > \frac{1}{2} \\ 0 & \text{otherwise.} \end{cases}$$

This gives an approximation to the interface after a time, Δt . To produce an approximation to the interface over a time, $n\Delta t$, we carry out n repetitions of steps 2 and 3.

Extension to Multiple Junctions

An extension of the algorithm to multiple regions has also been made.

Start from an initial state over a domain, D :



Initial curve cuts space into N regions.

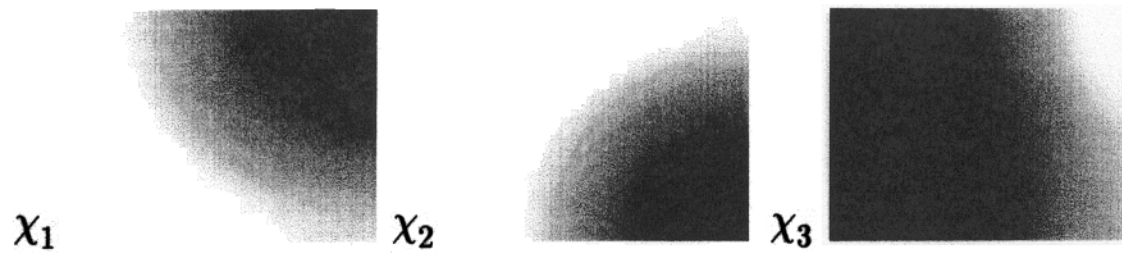
One step of the algorithm for multiple regions proceeds as follows:

1. χ_i is set to the characteristic function for the i th region:



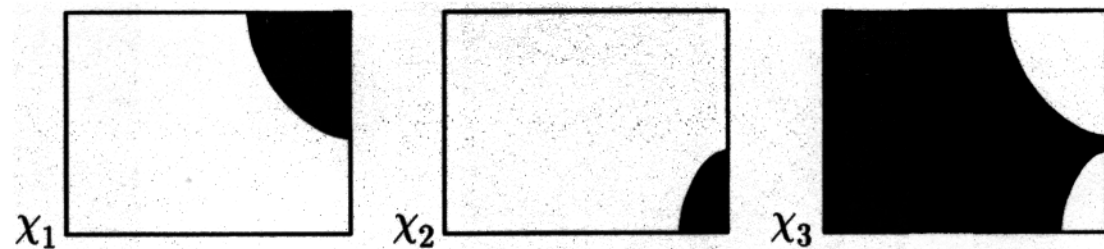
Characteristic functions

2. Diffusion is applied for a time, Δt , independently.



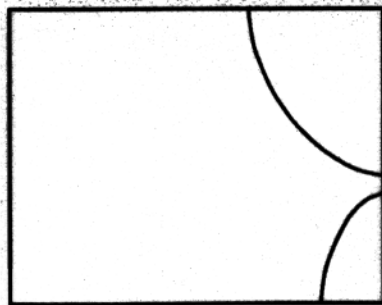
Find $\chi_i(\vec{x}, \Delta t)$ using $\frac{\partial \chi_i}{\partial t} = \Delta \chi_i$.

3. The regions are “Sharpened”. The “biggest χ wins.”



I.E. Set the largest χ_i equal to 1 and the others equal to 0 on each point of the domain.

The final location of the interface may then be found,



Final curves

Motion by Mean Curvature Algorithm

As outlined, diffusion-generated motion by mean curvature is only a semi-discrete method. How should we discretize in space?

We could simply use a finite difference method, but this leads to several problems, some of which we consider below:

- The time step, Δt , must be large enough so that the motion of the interface over each step can be resolved by the spatial discretization. For the case of a finite difference discretization, the level set $1/2$ must move at least one grid point, otherwise the front will remain stationary. This produces the restriction that

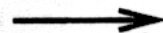
$$(\text{speed of motion of the interface}) \times \Delta t \gg \text{grid spacing.}$$

$$\kappa \Delta t \gg h.$$

Thus a prohibitively fine grid spacing will be needed if κ is small.

```
1 1 1 | 0 0 0
1 1 1 | 0 0 0
1 1 1 | 0 0 0
1 1 1 | 0 0 0
```

Initial Interface



```
x x x
x x x
x x x
x x x
```

Sharpen

```
1 1 1 | 0 0 0
1 1 1 | 0 0 0
1 1 1 | 0 0 0
1 1 1 | 0 0 0
```

Final Interface

1/2

Motion by Mean Curvature Algorithm

Suppose that the above restriction can be satisfied everywhere.

- Diffusion-generated motion by mean curvature produces an $\mathcal{O}((\Delta t)^2)$ error in the position of the front at each step.
- The sharpening step produces an error in the position of the front which is about the same size as the mesh spacing. Thus the sharpening step produces an $\mathcal{O}(h)$ error in the position of the front.

To preserve the overall accuracy of the method, $h = \mathcal{O}((\Delta t)^2)$.

$\Rightarrow \mathcal{O}\left(\frac{1}{(\Delta t)^4}\right)$ grid points using a uniform mesh in 2D

$\Rightarrow \mathcal{O}\left(\frac{1}{(\Delta t)^4}\right)$ operations per step

If a banded method could be used then a significant speed-up would occur: $\mathcal{O}\left(\frac{1}{(\Delta t)^3}\right)$ operations per step would be needed.

This idealized approach is still far too slow for many applications.

A finite difference discretization of the diffusion-generated algorithm can be expensive, even for simple 2-dimensional applications. We now describe a new, spectral method which is much faster than the usual finite difference approach.

To begin, we require a method to solve the heat equation,

$$\chi_t = \Delta \chi$$

repeatedly over intervals of length Δt . This is easily accomplished using a Fourier series...

Q. χ is initially discontinuous. Isn't a Fourier representation a poor choice for such functions?

A. The Fourier representation of $\chi(\vec{x}, 0)$ will contain a high frequency error which arises from truncating the Fourier series. We are only require χ after a time Δt . After a time Δt , high frequency error modes have been damped out. Since the problem is linear, the various modes do not interact and thus there is no need to approximate the high frequency components of χ . Thus a Fourier series is an excellent choice, because far fewer basis functions are required than might be expected.

A Fast Discretization

We now must carry out the thresholding step. Using the usual orthogonality conditions, it is easy to show that the Fourier coefficients of the characteristic function after thresholding are

$$c_{jk} = \iint_{R_t} \exp(-2\pi i j x) \exp(-2\pi i k y) dA$$

where

$$R_t = \left\{ \vec{x} : \chi(\vec{x}, t) > \frac{1}{2} \right\}$$

is the approximation to the phase we are following.

Q. How can these integrals be evaluated?

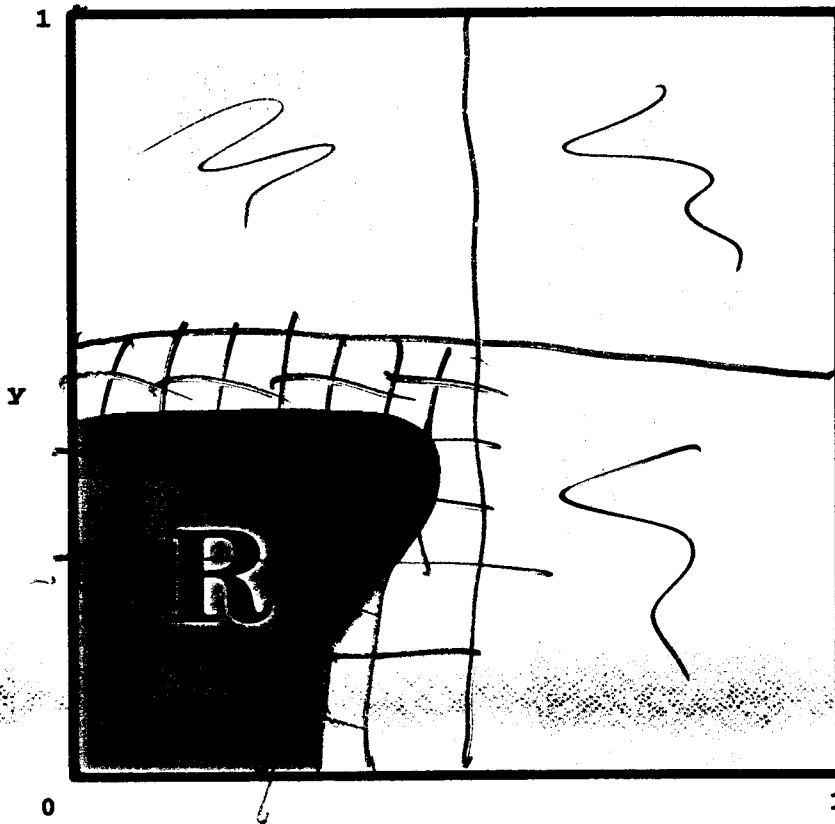
A. It is possible to integrate exactly over rectangular regions. We may integrate over a non-rectangular region, R_t , by recursively subdividing the domain into square subregions.

A Fast Discretization

We now evaluate

$$c_{jk} = \iint_R \exp(-2\pi i j x) \exp(-2\pi i k y) dA$$

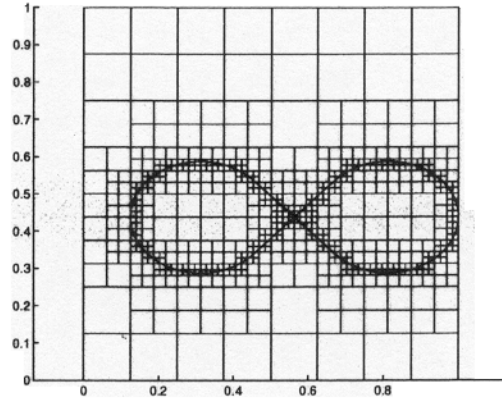
over the following:



At the finest grid subdivision, triangles are used to approximate the small subregions.

A Fast Discretization

1. By carrying out refinements gradually, nonsmooth features and corners are captured.



2. Evaluating the function, χ , and the Fourier coefficients CANNOT be done using the FFT since the mesh is unequally spaced. We use the unequally spaced fast Fourier transform method of Beylkin (1995) to carry out these evaluations efficiently.

This leads to $\mathcal{O}(N \log^2(N))$ operations in 2 dimensional, 2-phase problems with N basis functions, which corresponds to

$$\mathcal{O}\left(\frac{1}{\Delta t} \log^2(\Delta t)\right)$$

operations.

An idealized finite difference discretization leads to

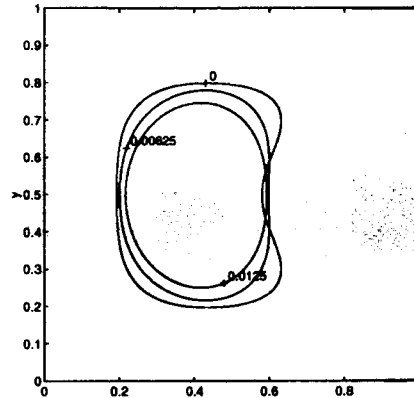
$$\mathcal{O}\left(\frac{1}{\Delta t^3}\right)$$

operations.

Comparison to the FD Approach

Test Problem:

Find the area lost over a time $t = 0.0125$ for a shrinking kidney-shaped interface.



Note: Exact result is given by $\dot{A} = -2\pi$

Results:

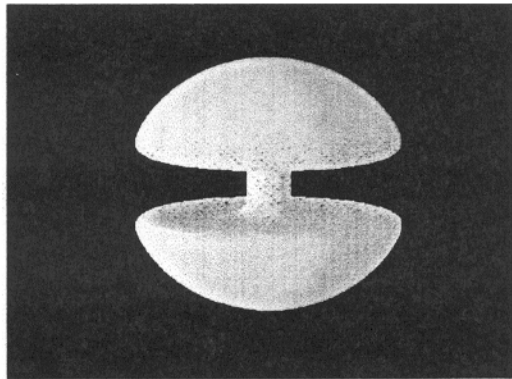
Δt	Width of Finest Cells	Error	Execution Time
0.003125	2 ⁻⁹	4%	0.4 s
0.00078125	2 ⁻¹¹	1%	8 s

Table 1. New, Spectral Method

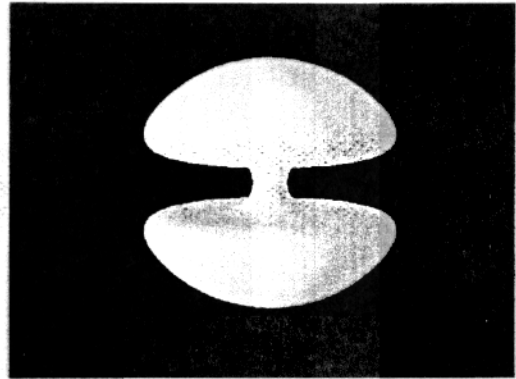
Δt	h	Error	Execution Time
0.0001	2 ⁻⁷	4%	85 s
0.00002	2 ⁻⁹	3%	10341 s

Table 2. Finite Difference Discretization

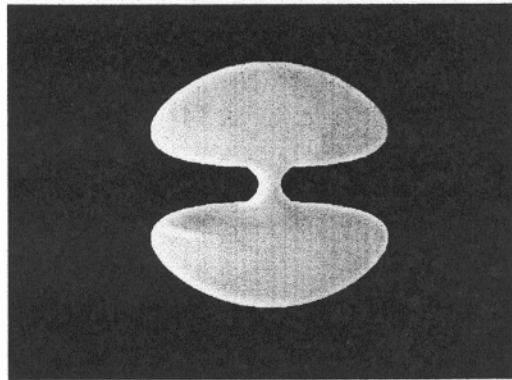
A Two Phase Case: Mean Curvature Motion



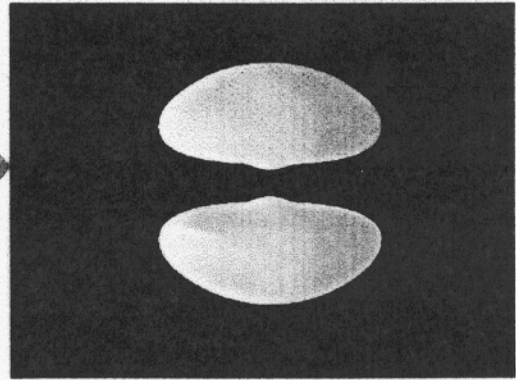
$t = 0.0000$



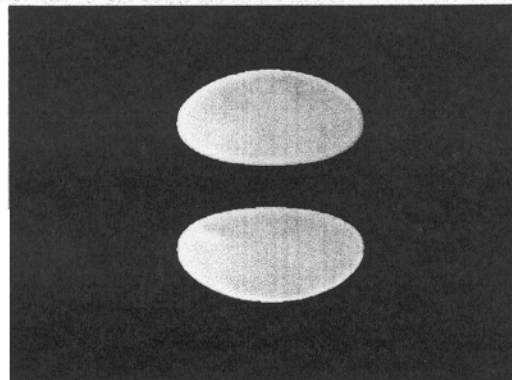
$t = 0.0008$



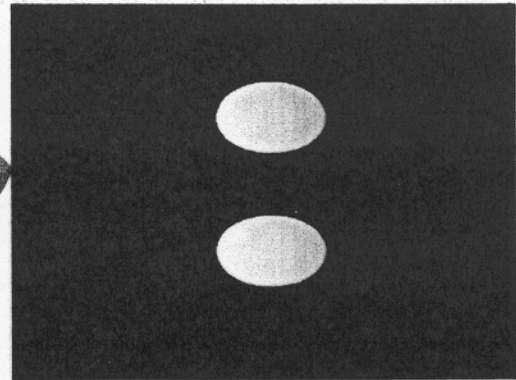
$t = 0.0020$



$t = 0.0032$



$t = 0.0064$



$t = 0.0128$

Multiple Junctions

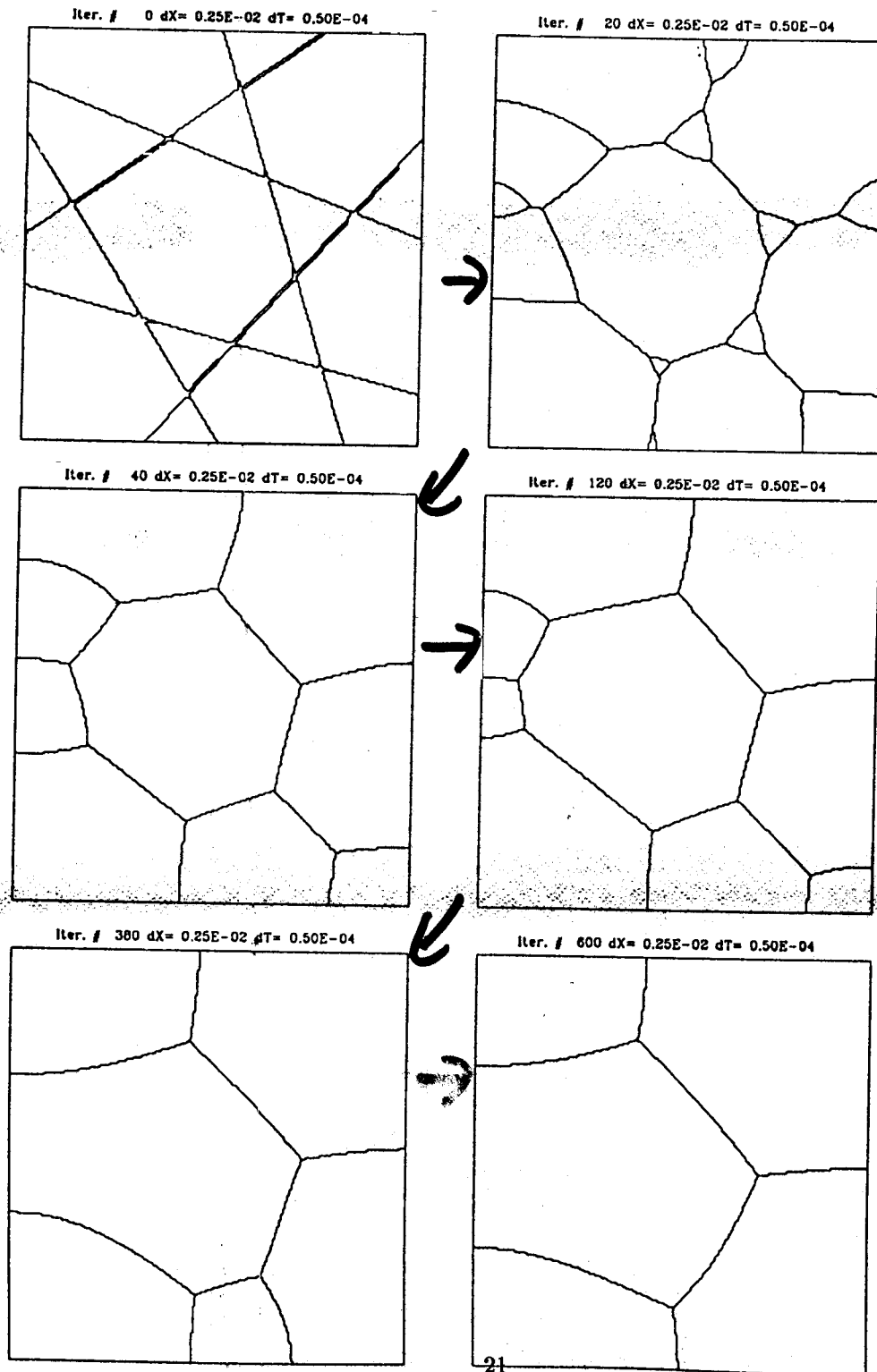
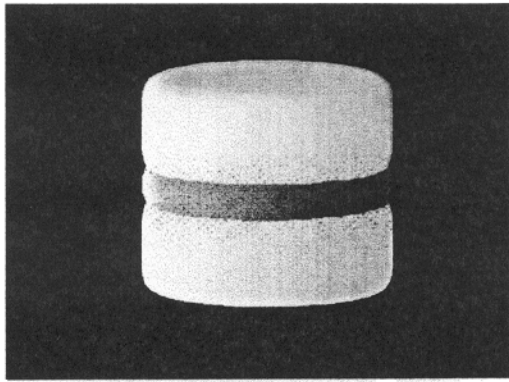
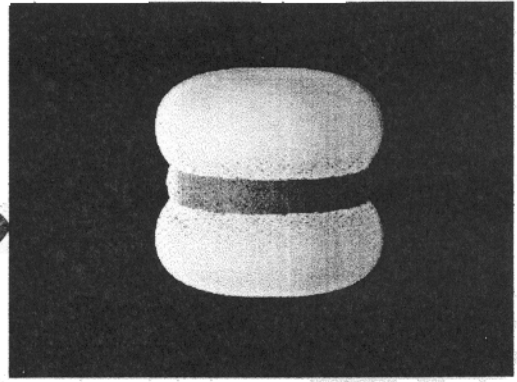


FIGURE 11. Motion of several regions under curvature

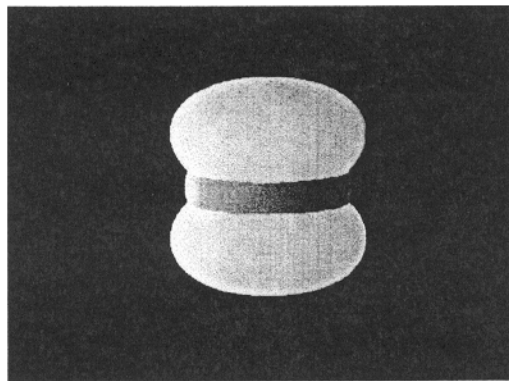
Multiple Junctions



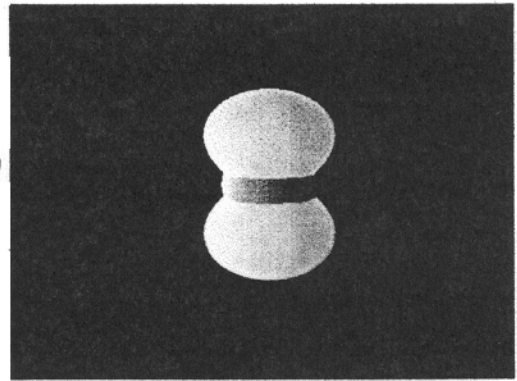
$t = 0.0000$



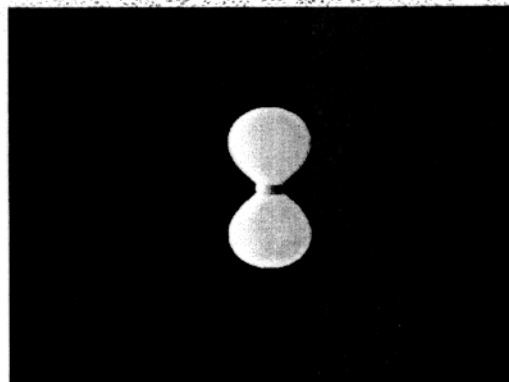
$t = 0.0025$



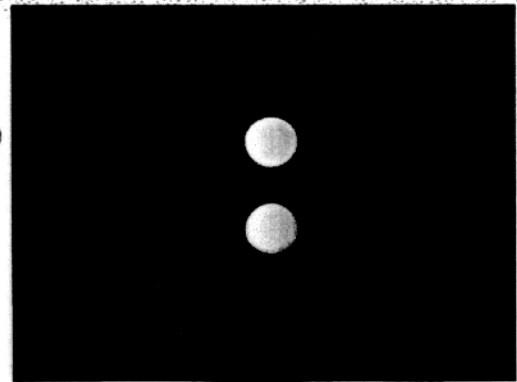
$t = 0.0100$



$t = 0.0175$

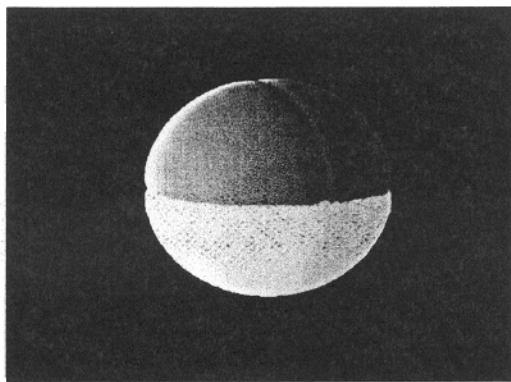


$t = 0.0220$

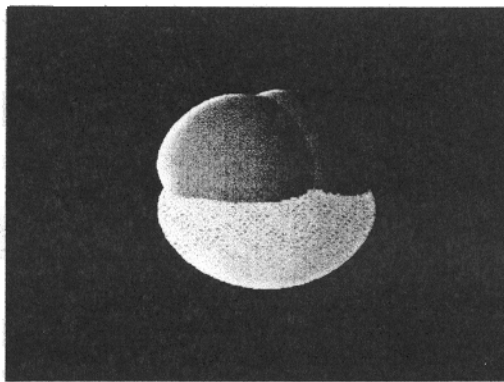


$t = 0.0260$

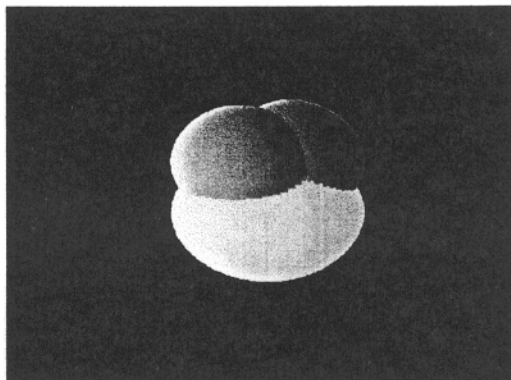
Multiple Junctions



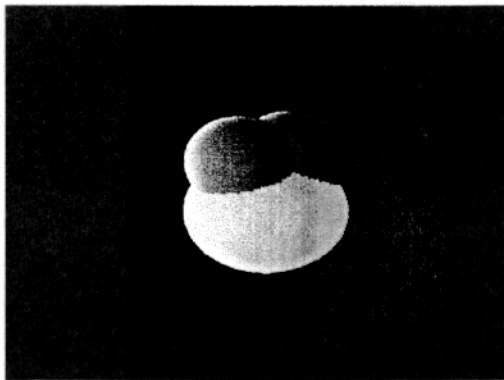
$t = 0.0000$



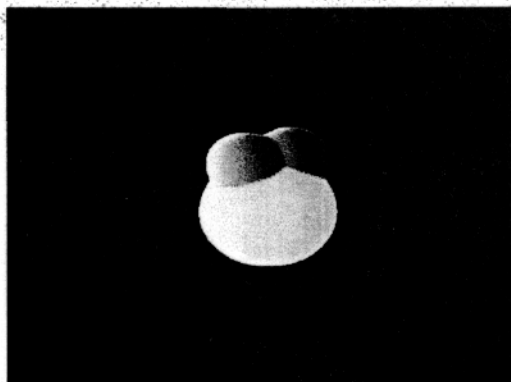
$t = 0.0036$



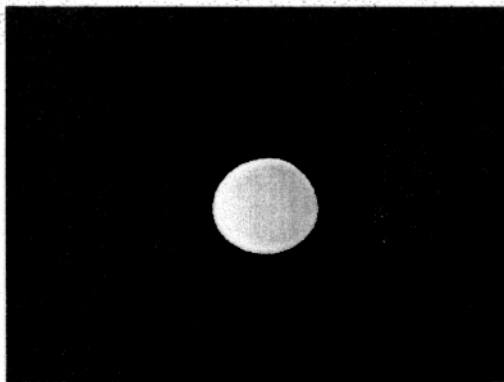
$t = 0.0072$



$t = 0.0108$



$t = 0.0144$



$t = 0.0180$

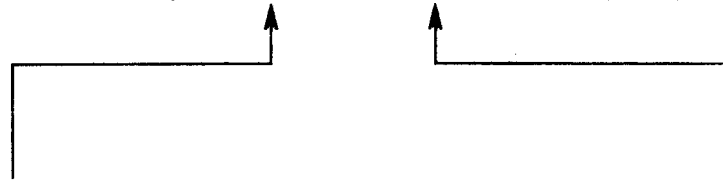
Evolve by $\chi_t = \Delta\chi$
for a time Δt

=

Convolve with Gaussian
heat Eqn. kernel
 $K(x) \sim \exp(-x^2/\Delta t)$
 $\chi(\Delta t) = \chi * K$

We may re-write the diffusion-generated motion algorithm as:

$$\{x : \chi * K(x) > 1/2\}$$



Different Kernel Functions

- asymmetric kernels can be used to produce anisotropic motion laws
- kernels should be simple, easy to compute with and produce general motion laws

General Thresholds: $\lambda \in (0,1)$

- $\lambda=0$ corresponds to constant motion
- $\lambda=1/2$ corresponds to curvature motion for rotationally symmetric kernels
- $\lambda = 1/2 + c\sqrt{\Delta t}$ corresponds to curvature plus a constant
- We can also let λ vary according to the local normal to obtain a variety of anisotropic motion laws

Convolution Based Methods

We have described a class of schemes which are computationally efficient and have a great deal of unexplored flexibility. Some interesting questions include:

- Given a particular kernel, what motion law arises?
(Ishii, Pires and Souganidis 1997)
- How should the kernel and thresholding be selected to derive a desired motion law?
(Ruuth and Merriman 1998)

Anisotropic Curvature-Dependent Motion (2D)

In 2D, general anisotropic motions of the form

$$v_n = a(\theta) + b(\theta)\kappa$$

can be derived by selecting a non-symmetric kernel and varying the threshold according to the normal direction.

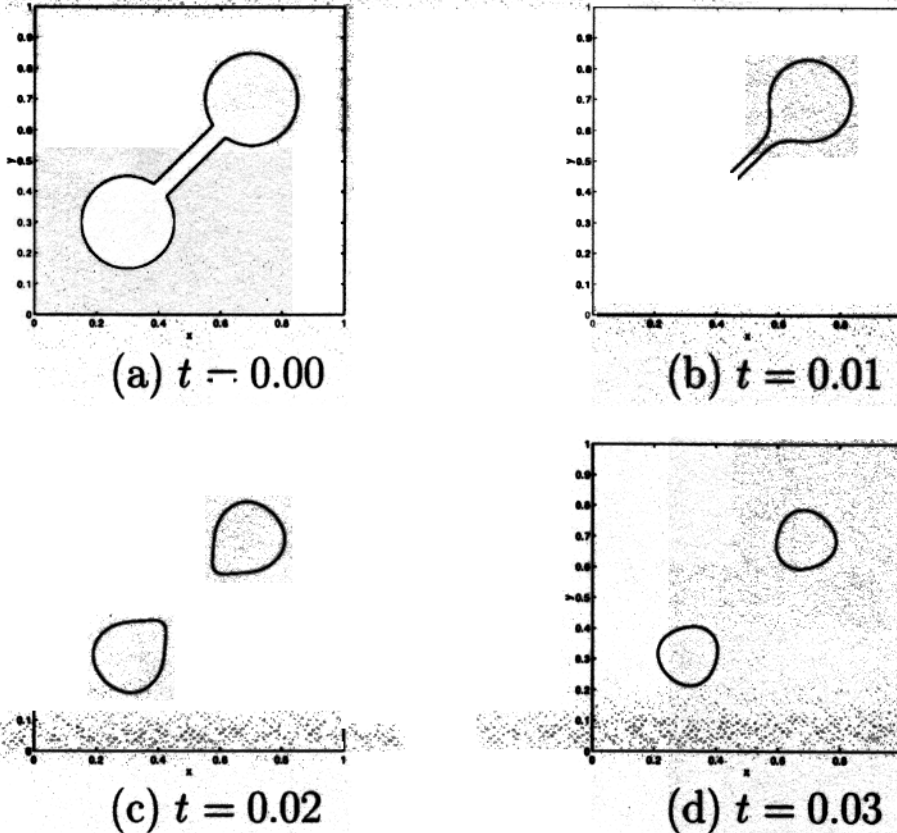
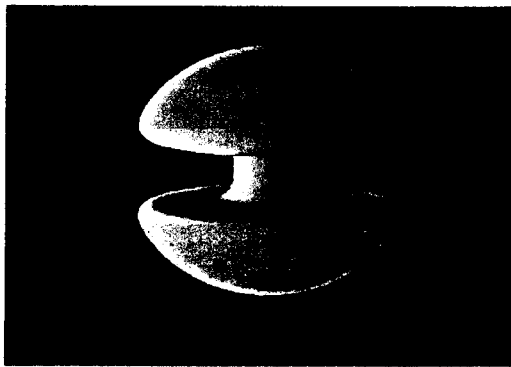
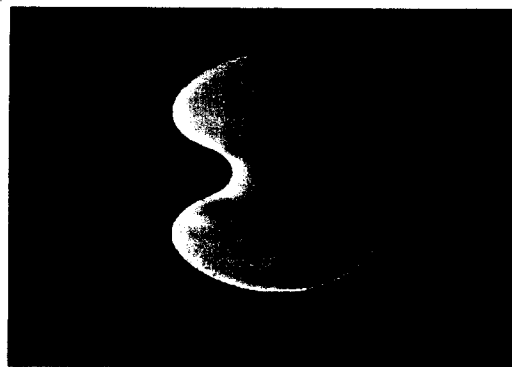


Figure 2: A test problem at various times, t . The normal velocity is given by $v_n = a(\theta) + b(\theta)\kappa$ where $a(\theta) = -|\sin(\theta)| - |\cos(\theta)|$ and $b(\theta) = \frac{1}{10} + \frac{1}{20}\sin(2\theta)$.

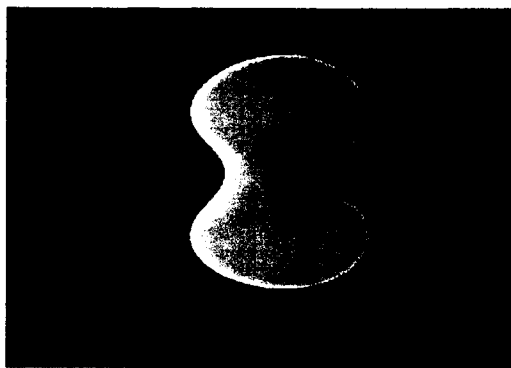
Curiously, it is impossible to achieve $v_n = b(n)\kappa$ in three dimensions using a single kernel (Ishii, Pires and Souganidis 1997). This limitation can be removed by using 2 kernels...



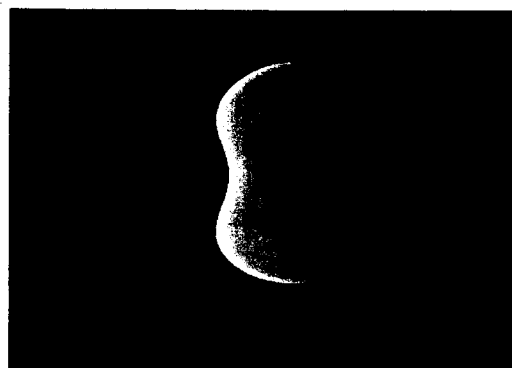
$t = 0.00000$



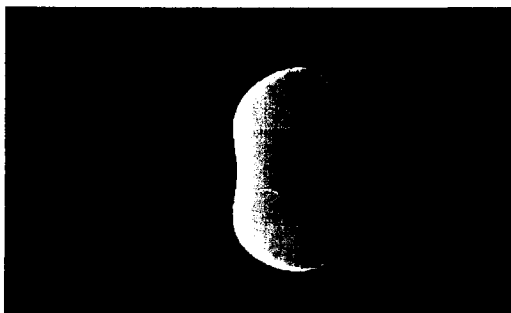
$t = 0.00200$



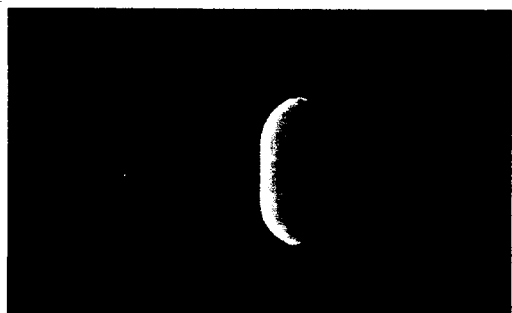
$t = 0.00375$



$t = 0.00575$



$t = 0.00750$



$t = 0.00950$

A normal velocity $v_n = (1 + \sqrt{n_1^2 + n_2^2} + \sin(\pi n_1))\kappa$

Volume Preserving Mean Curvature Motion

A volume-preserving motion is derived by thresholding according to the level that encloses the same volume as the original set.

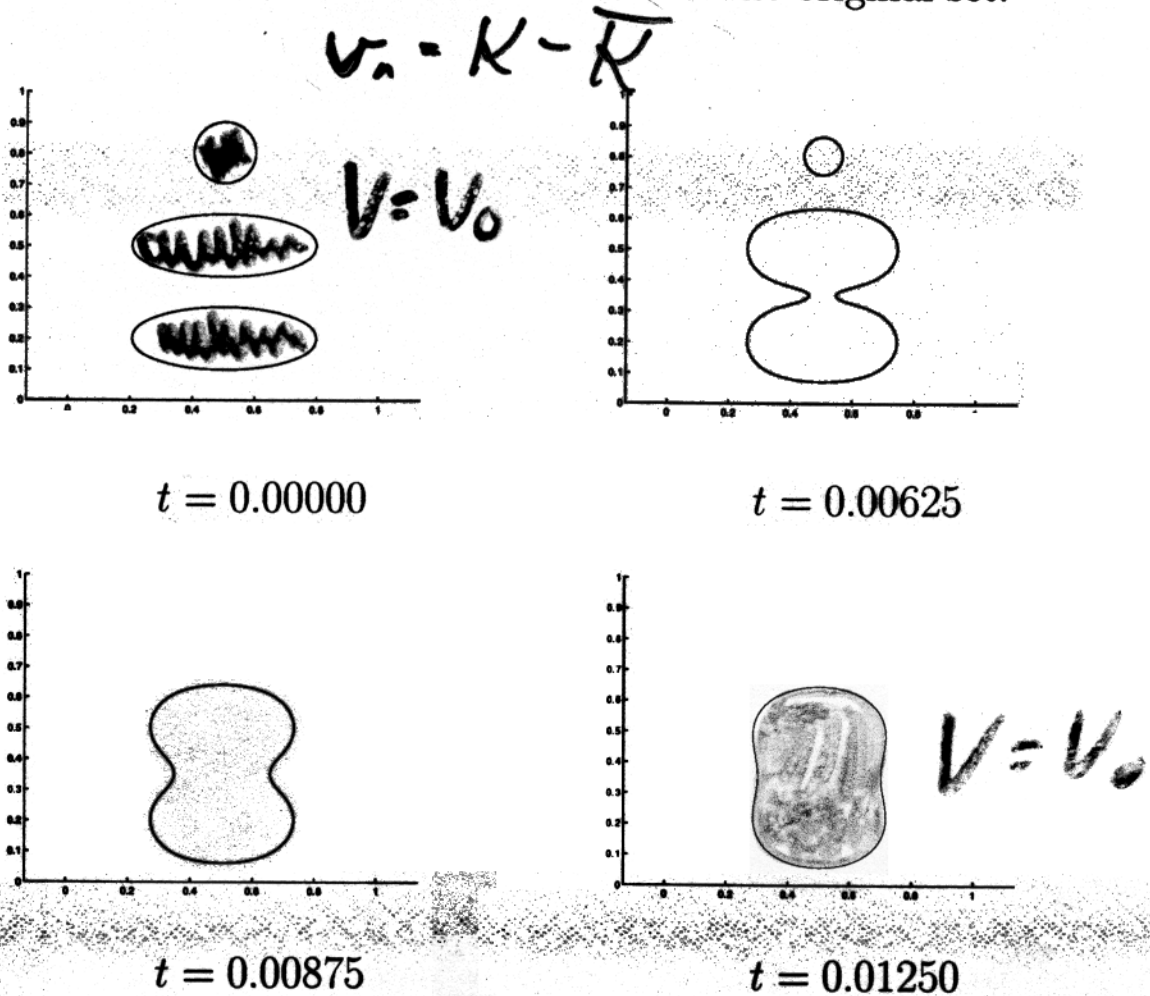
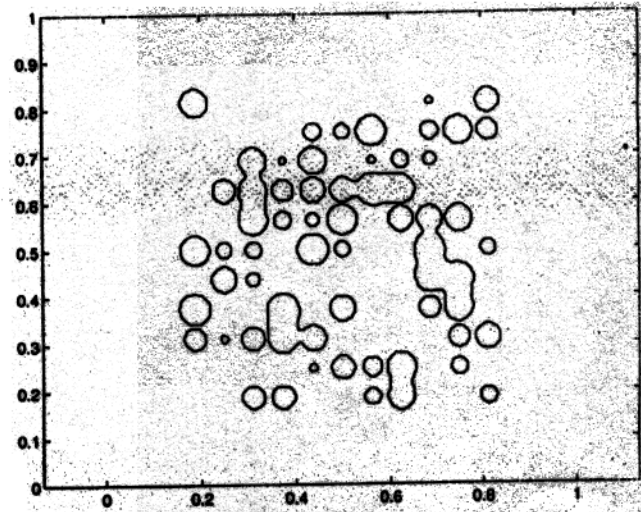
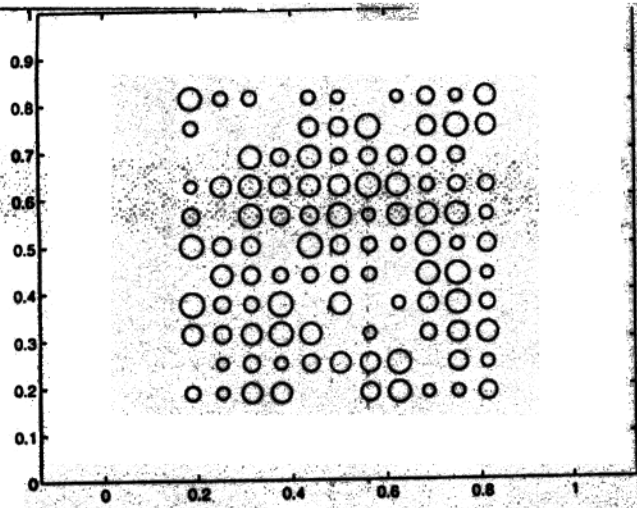


Figure 3: Nonlocal Model Which Preserves Area

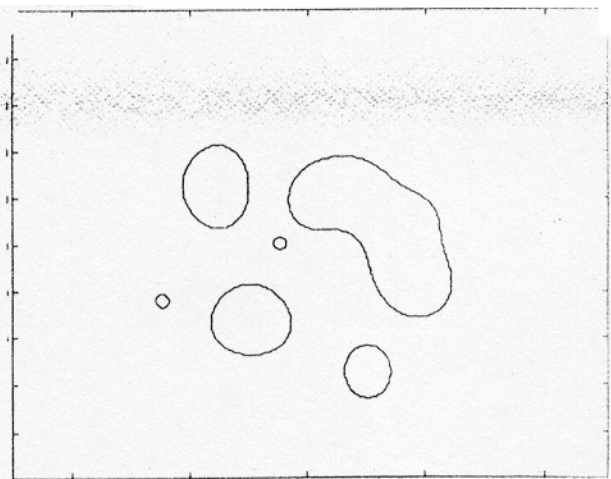
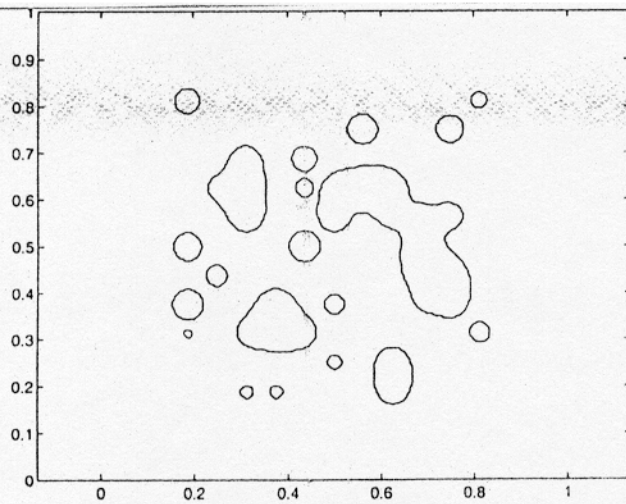
Numerically, this simple method agrees with the nonlinear evolution model $v_n = \kappa - \bar{\kappa}$.

Volume Preserving Mean Curvature Motion:

$$v_n = \kappa - \bar{\kappa}$$



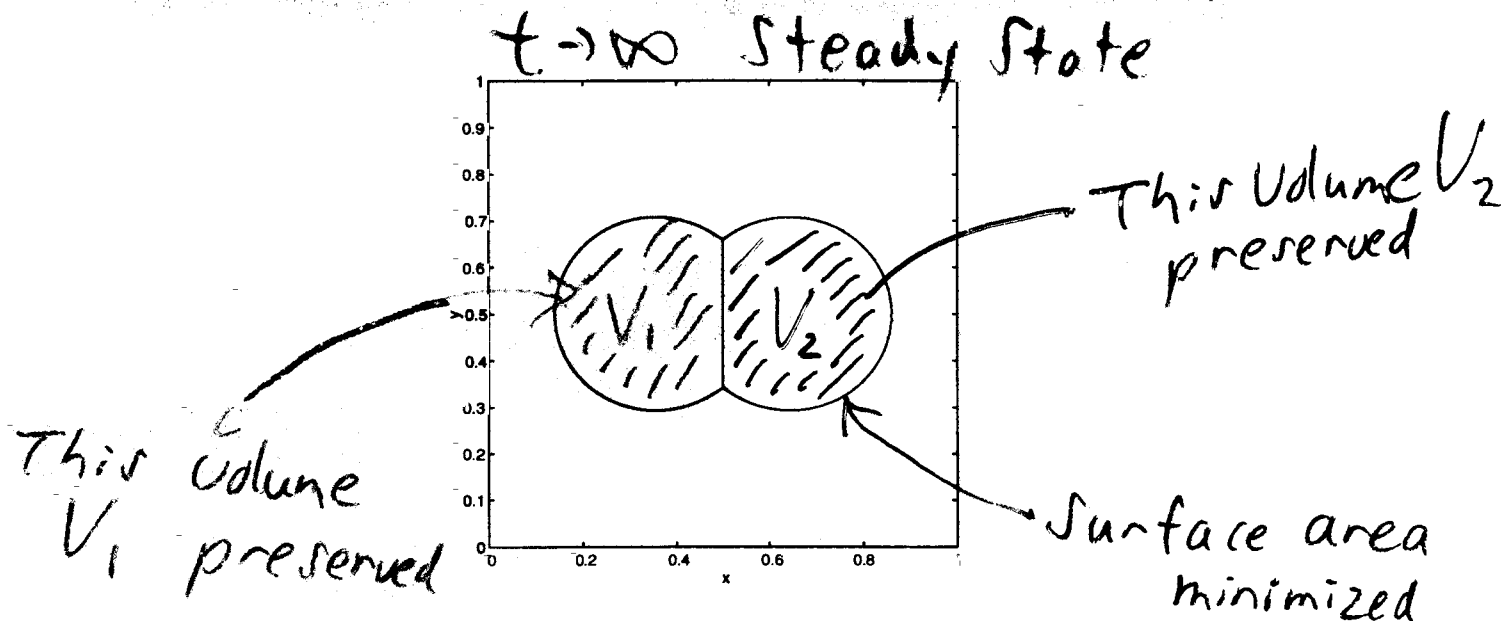
Initially: 100 "Bubbles"



Multi-Volume Preserving Mean Curvature Motion

When applied to multiple regions, this algorithm gives a means of finding the shapes having the least surface area enclosing prescribed volumes.

A test on the simplest case gives the standard double-bubble:



A proof that this is the correct minimizer is given by Foisy, Alfaro, Brock, Hodges, Zimba 1993. The 3D case is treated by Hass, Hutchings, Schlafly 1995.

Our approach allows for many bubbles. The surface tension between different bubbles can also be varied to obtain a minimization of the weighted surface area enclosing prescribed volumes.

Approximation of Multiphase Models

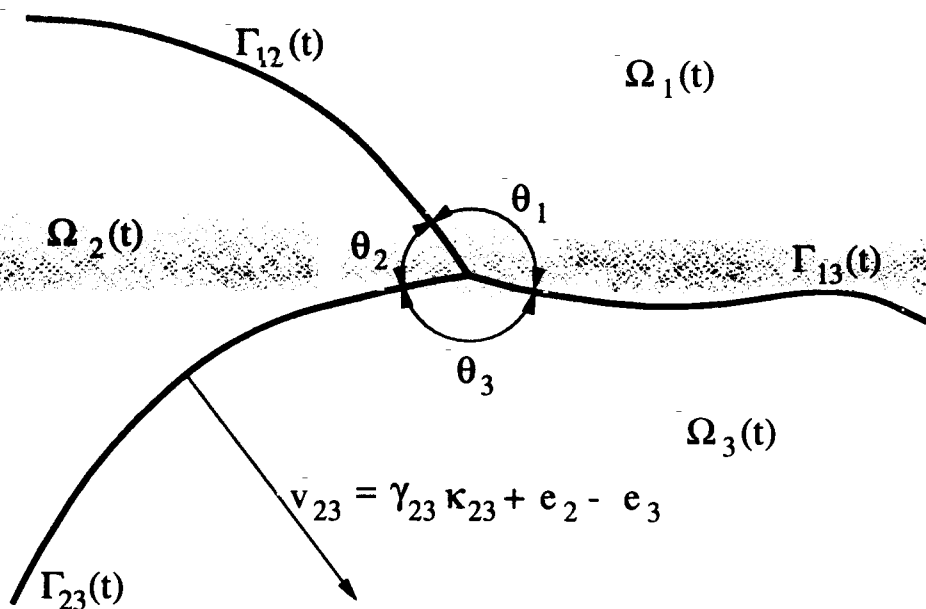
Diffusion-generated motion by mean curvature has direct extension to the motion of multiple junctions (Merriman, Bence, Osher 1992).

This algorithm gives symmetric junctions. Modifications for certain other junction conditions are given by Mascarenhas 1992 and Merriman, Bence and Osher 1994.

A more general, diffusion-generated approach for the case

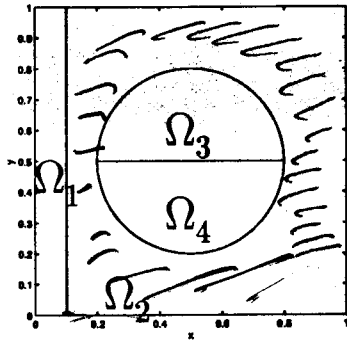
$$v_n = \gamma_{ij} \kappa + e_i - e_j$$

has also been developed (Ruuth 97). This method combines the results from a number of different problems and uses a nonsymmetric sharpening.

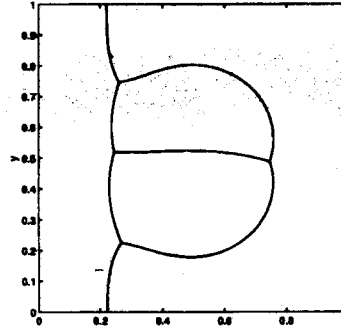


Approximation of Multiphase Models

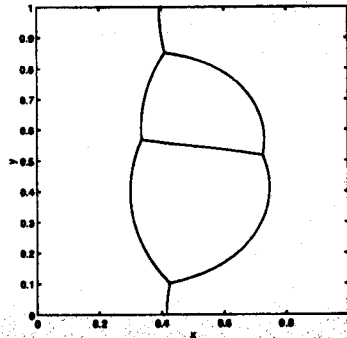
Complicated multiphase flows are easily accommodated using this approach:



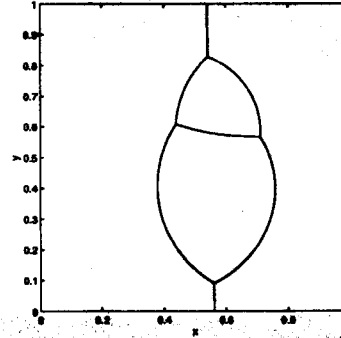
(a) $t = 0.00$



(b) $t = 0.03$

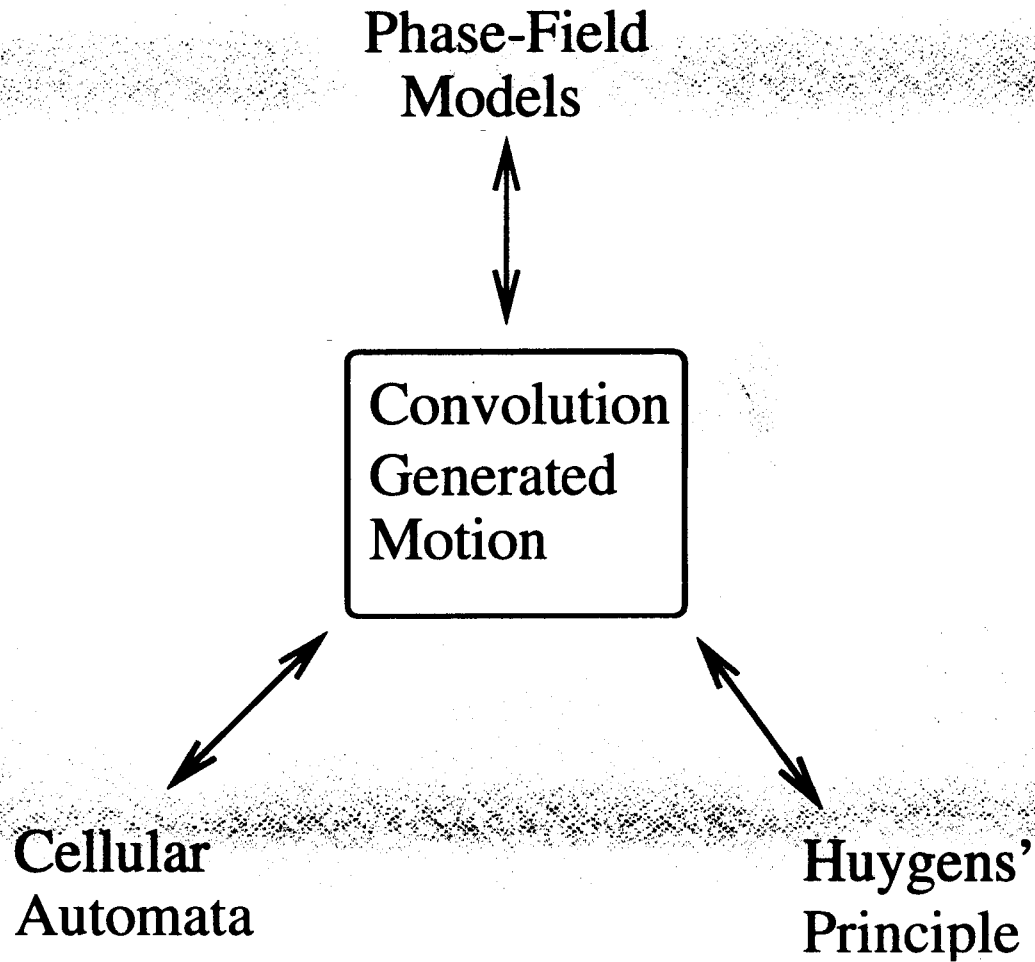


(c) $t = 0.06$

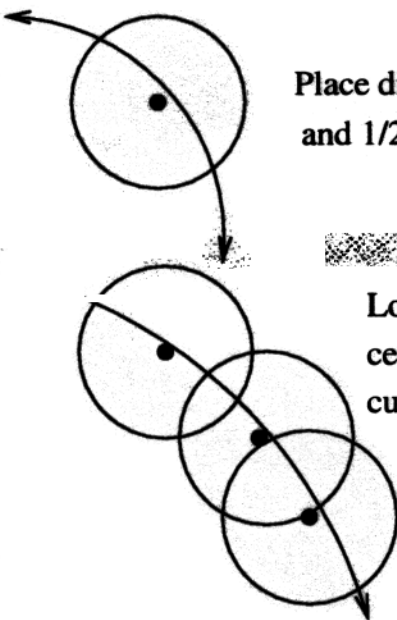



(d) $t = 0.09$

Figure 3: A test problem at various times, t . Here $(e_1, e_2, e_3, e_4) = (0, 4, 1, 0)$



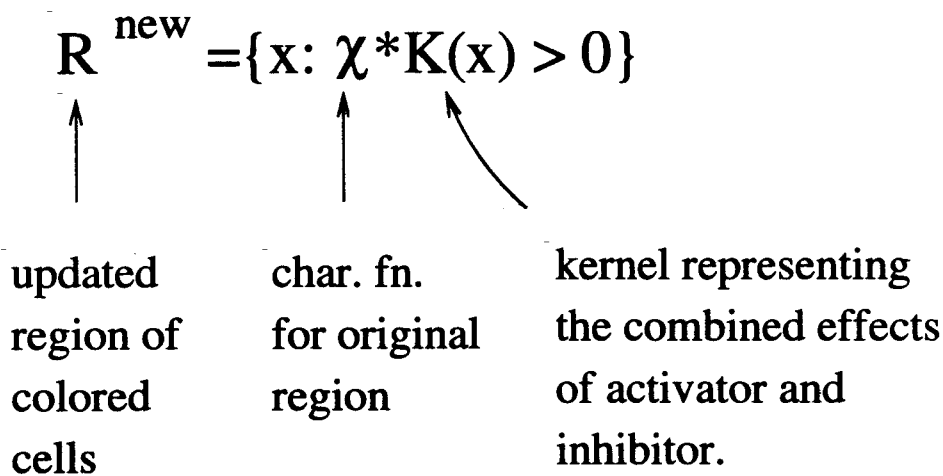
Connection to Huygens' Principle

CONSTANT MOTION	Analytic Form	Geometric Form
	<p>Take $K = \text{char. fn. for a small disc}$</p> $\{x: \chi * K > 0\}$ <p style="text-align: center;">↑</p> <p>Updated region consists of all points where there is no overlap of the disc with the original region.</p>	<p>Classical Huygens' Principle</p>
CURVATURE MOTION	<p>Convolution Generated Motion</p>	 <p>Place disc 1/2 in and 1/2 out by area.</p> <p>■ Locus of disc centers gives curvature motion</p>
	<p>Take $K = \text{char. fn. for a small disc}$</p>  <p>The updated region is given by:</p> $\{x: \chi * K > 1/2\}$ <p style="text-align: center;">↑</p> <p>Note that the overlap of the old region with the disc is 1/2 by area</p>	

A Developmental Biology Model

Young (1984) developed a simple automaton for modeling animal coat patterns which is based on short range activation and long range inhibition.

Young's model has a particularly simple convolutional form:

$$R^{\text{new}} = \{x: \chi * K(x) > 0\}$$


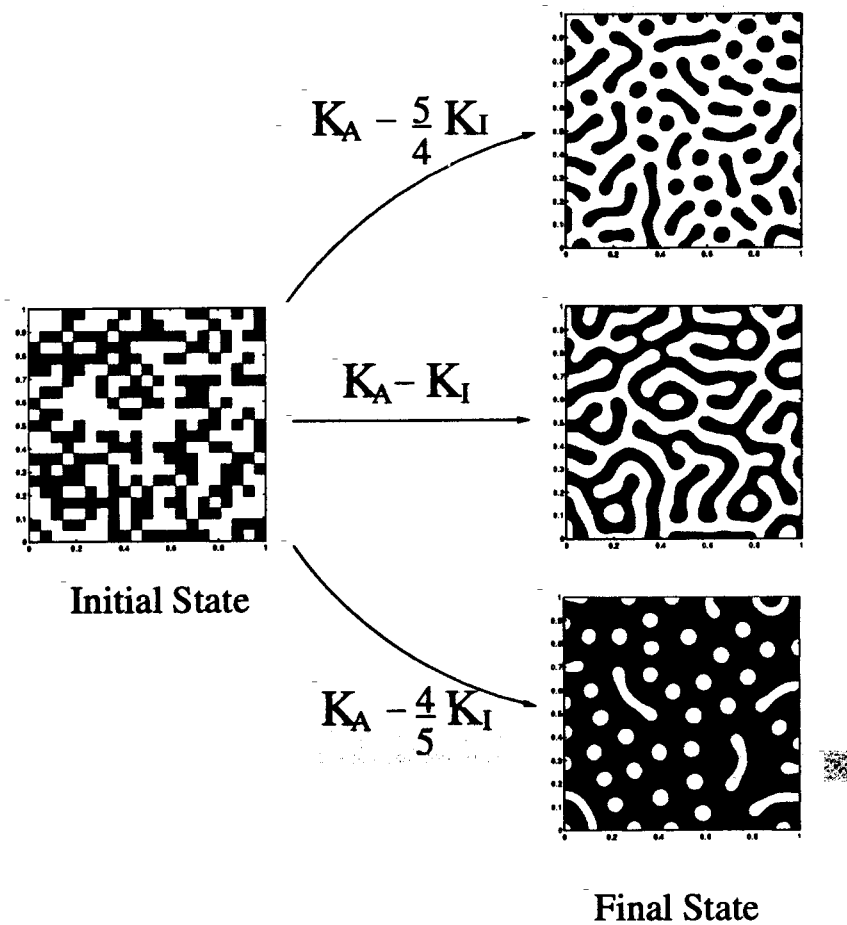
updated region of colored cells

char. fn. for original region

kernel representing the combined effects of activator and inhibitor.

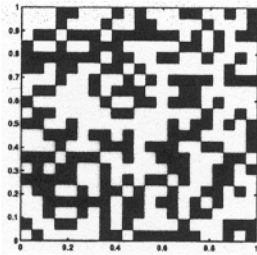
A Developmental Biology Model

Steady state patterns are efficiently obtained...

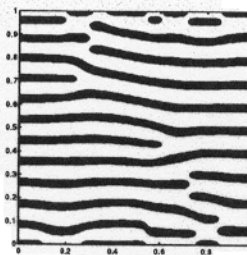
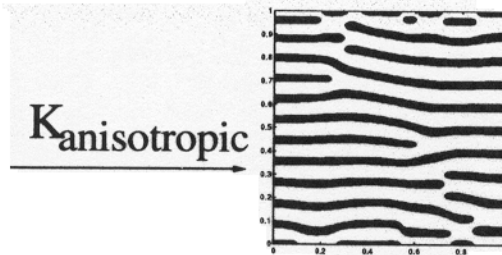


Anisotropic Developmental **Biology** Model

Anisotropy can be specified precisely.



Initial State



Final State

Excitable Media

Excitable media arise in diverse physical, chemical and biological systems including models for nerve cells, muscle cells, cardiac function, developmental biology, chemical reaction and star formation.

Automata models for excitable media often require large neighborhoods in order to:

- qualitatively model the effects of wavefront curvature
- make the speed of propagation depend on the recovery of medium
- reduce lattice-based anisotropy in the motion

A variety of authors have developed automata for excitable media, including
Gerhardt, Schuster and Tyson 1990-1991
Weimar, Tyson and Watson 1992
Henze and Tyson 1996

The main idea of these models is to evolve the excited region by threshold growth using a threshold level that depends on a second variable (called the recovery variable). The recovery variable can be treated using standard numerical PDE methods since it evolves according to a nonstiff PDE.

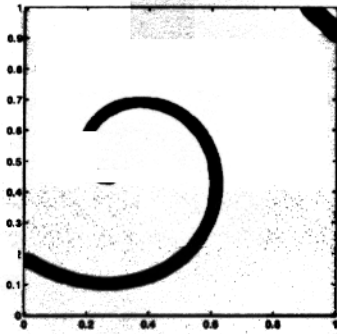
Unfortunately, non-physical grid-based effects can occur with these automata. Some specific problems include:

- Incorrect curvature effects \Rightarrow incorrect front speeds.
- Unwanted anisotropy.
- Low order accurate representation of the front location (note that each step produces an order- h error in the front location).
- Large neighborhoods lead to prohibitively expensive calculations.
- It is often impractical to determine the continuum limit of the model.

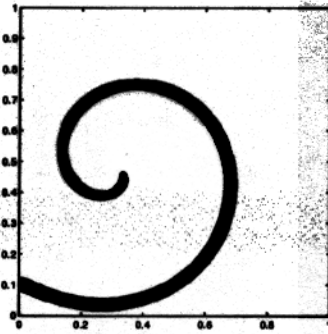
Our convolution based approach gives a solution to these problems:

- They easily accommodate large kernels with no additional expense.
- They give an accurate representation of the front. This eliminates unwanted anisotropy and produces a good approximation to higher order effects such as curvature.
- Since a convolution arises in the continuum limit of these automata, our approach automatically provides an approximation of the continuum limit of the model.

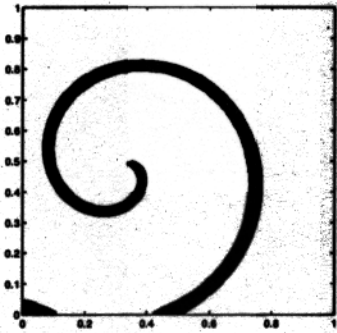
Excitable Media: Spiral Wave



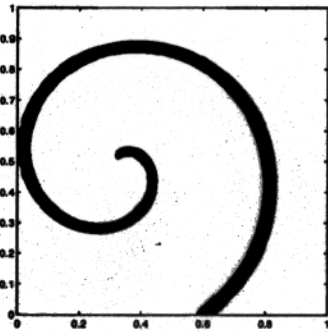
$t = 0.000$



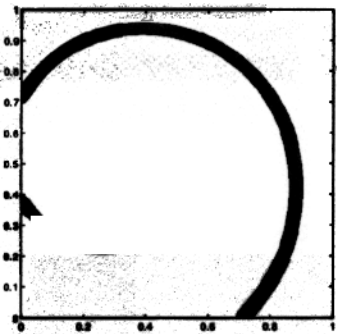
$t = 0.375$



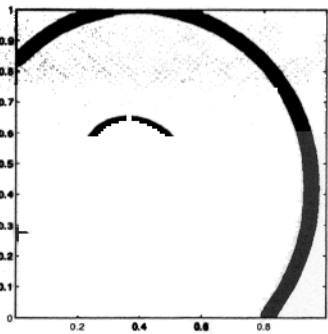
$t = 0.750$



$t = 1.125$



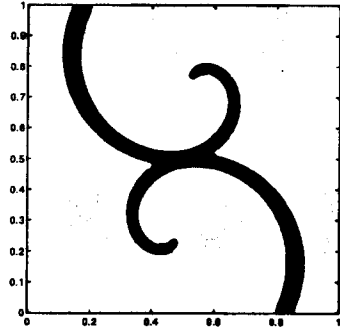
$t = 1.500$



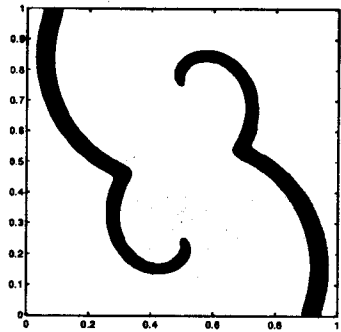
$t = 1.875$

A spiral wave.

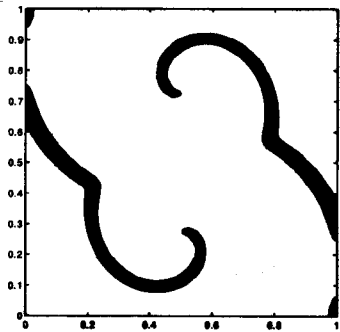
Excitable Media: Two Armed Spiral



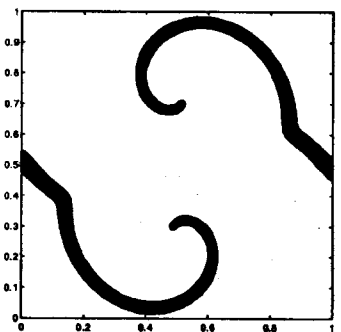
$t = 0.000$



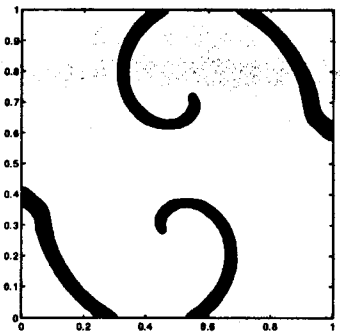
$t = 0.375$



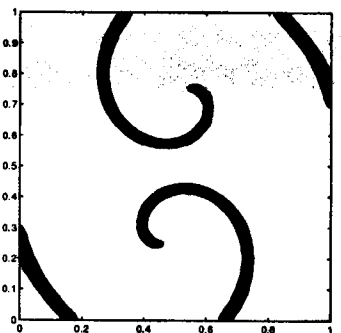
$t = 0.750$



$t = 1.125$



$t = 1.500$



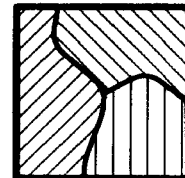
$t = 1.875$

A rotating two-armed spiral.

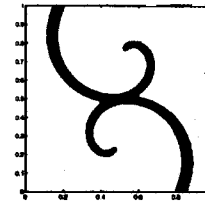
Convolution Based Methods: A Summary

- Convolution generated motion gives simple and efficient algorithms for computing interesting curvature-dependent motions.

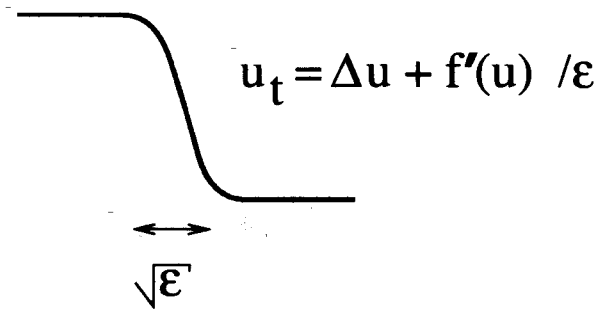
- The method has a very natural extension to the case of junctions.



- They provide a natural link between cellular automata and their continuum limit.



- Convolution based methods are reminiscent of phase field models, but do not introduce an artificial small length scale.



To get more details on the subject, you are welcome to visit the Diffusion-Generated Motion by Mean Curvature Home Page:

~~<http://www.math.ucla.edu/~rsmith/research.html>~~

or email me: ~~rsmith@math.ucla.edu~~

www.math.stu.ca/~rsmith

rsmith@stu.ca

Connection to Phase Field PDEs

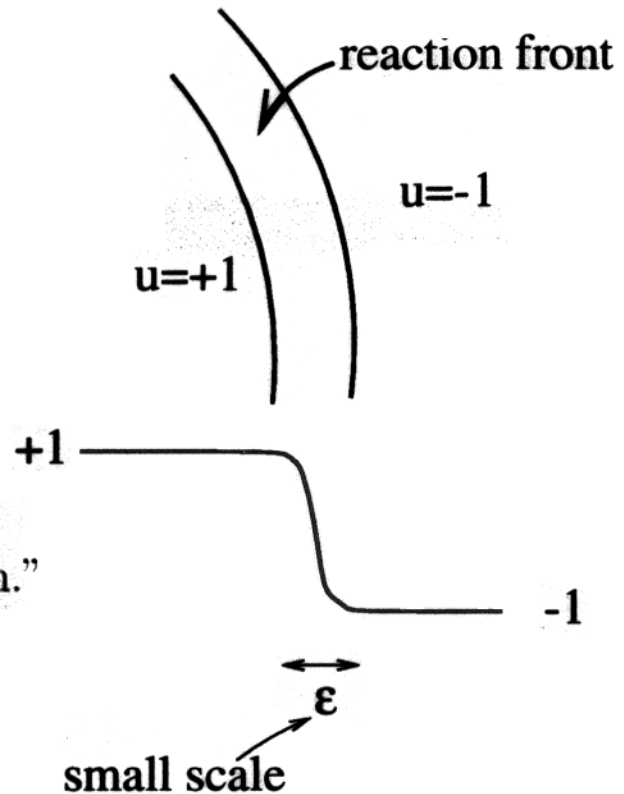
Consider the phase field PDE:

$$\frac{\partial u}{\partial t} = \Delta u - \frac{1}{\epsilon^2} u(|u|^2 - 1)$$

In the $\epsilon \rightarrow 0$ limit:

Front moves by mean curvature!

For numerics, grid must resolve front ($\Delta x \ll \epsilon$) or front gets “frozen.”



Diffusion Generated Motion:

ALTERNATES: Step of diffusion and Step of $u \rightarrow \pm 1$

Diffusion-generated motion directly obtains the $\epsilon \rightarrow 0$ limit without the small scale.

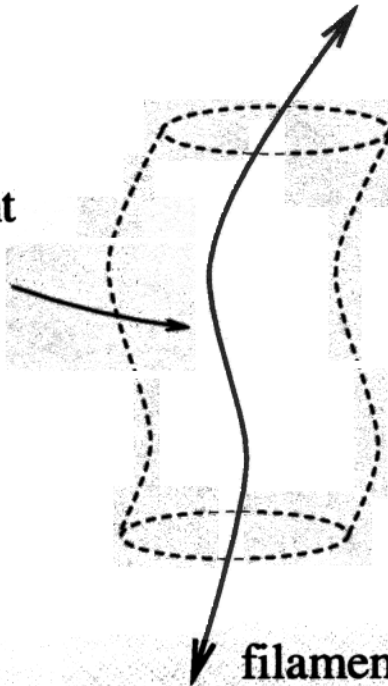
Q: Is this connection useful to obtain more general motions?

Connection to Phase Field PDEs

Consider the complex valued Ginzburg-Landau equation:

$$\frac{\partial u}{\partial t} = \Delta u - \frac{1}{\epsilon^2} u (|u|^2 - 1)$$

Δu is important
in an ϵ -nbhd
of the filament



reaction term
drives $u \rightarrow u/|u|$

filament is given by $u=0$
or by the center of winding of u

In the $\epsilon \rightarrow 0$ limit: Filament moves in the normal direction with a speed equal to curvature!

As with the diffusion-generated motion of surfaces, the grid must be resolved ($\Delta x \ll \epsilon$) or the filament gets "frozen."

Connection to Phase Field PDEs

Diffusion-generated motion by mean curvature for filaments alternates a step of diffusion and a step of $u \rightarrow u/|u|$.

1. "Initialize:" $\chi(x) = \exp(i\theta(x))$

Set χ so that its center of winding coincides with the filament.



Any angle function that winds around the filament.

2. "Normalize:" $\tilde{\chi} = \chi/|\chi|$

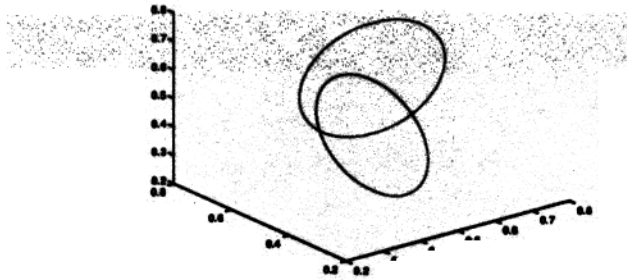
3. "Diffuse:" Starting from $\tilde{\chi}$, evolve χ for a time Δt according to $\chi_t = \nabla^2 \chi$.

The zero contour of χ gives an approximation to the interface after a time Δt .

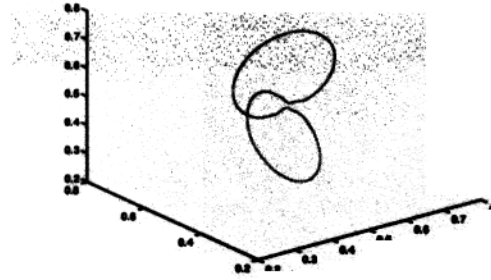
For the filament case:

- Topological mergers are captured with no special algorithmic procedures.
- Diffusion-generated motion directly obtains the $\epsilon \rightarrow 0$ limit of the phase field model without the small scale.
- An extension to surfaces of arbitrary codimension is also available.

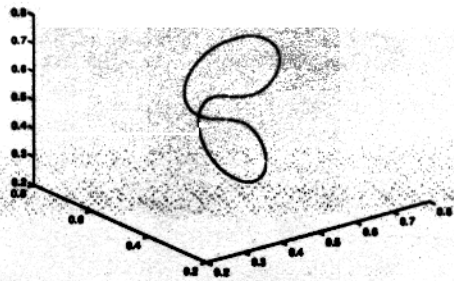
Filament Motion in Three Dimensions



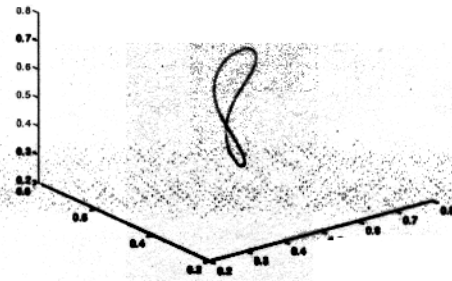
$t = 0.000$



$t = 0.008$



$t = 0.009$

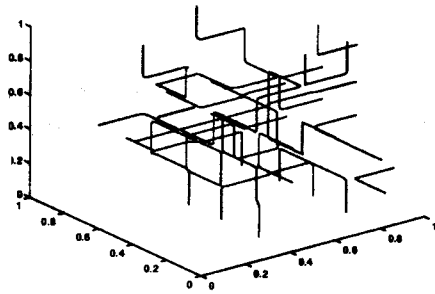


$t = 0.016$

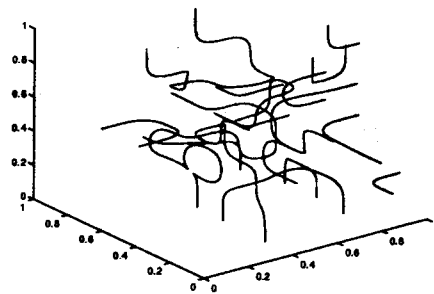
The curvature motion of two filaments: $\vec{v} = \kappa \hat{n}$

Filament Motion in Three Dimensions

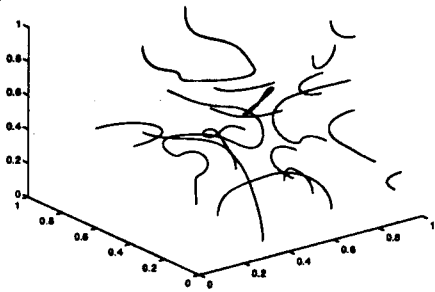
Large systems of filaments may also be treated.



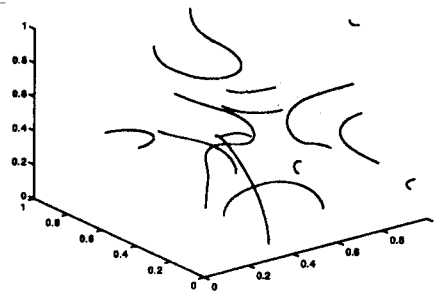
$t = 0.000$



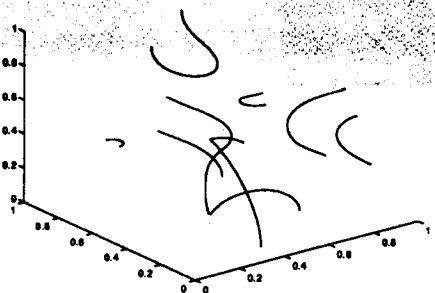
$t = 0.001$



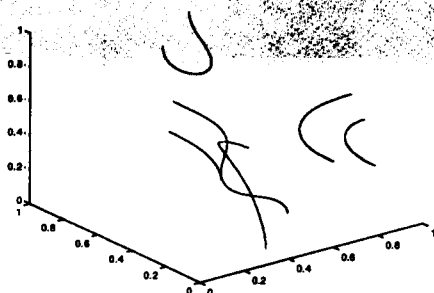
$t = 0.005$



$t = 0.010$



$t = 0.015$



$t = 0.020$

Connection to Cellular Automata

Cellular automata have been used to model the formation and dynamics of patterns in a variety of chemical, biological and ecological systems.

An important class of automata is obtained by imagining each neighbor's contribution to be a simple "vote" for or against a certain outcome; any number of affirmative votes above a certain threshold will yield that outcome.

Example:

0	1	1	1	1	1
0	1	1	1	1	1
0	0	1	1	1	1
0	0	0	1	1	1
0	0	0	0	0	0

Suppose there are 2 states:
1 and 0

A sum of cell's
own vote and 8
neighbors is
formed.

2	4	6	6	6	4
2	5	8	9	9	6
1	3	6	8	9	6
0	1	3	5	6	4
0	0	1	2	3	2

0	1	1	1	1	1
0	1	1	1	1	1
0	1	1	1	1	1
0	1	1	1	1	1
0	0	0	0	1	0

Where this sum is greater than or equal to the threshold value (e.g., 3) the cell is assigned state 1, and state 0 elsewhere.

Connection to Cellular Automata

By denoting the state of cell (j, k) at time step n by C_{jk}^n , we obtain a simple analytic representation for the automaton.

Set S_{jk} equal to the sum of all 1's in the neighborhood:



$$S_{jk} = \sum_{-1 \leq j', k' \leq 1} C_{j-j', k-k'}^n$$

and update by thresholding at the level λ :

$$C_{jk}^{n+1} = \begin{cases} 1 & \text{if } S_{jk} \geq \lambda \\ 0 & \text{otherwise} \end{cases}$$

More generally, each vote can be assigned some weight to produce threshold dynamics:

Set S_{jk} equal to the sum of all 1's in the general neighborhood, N ,

$$S_{jk} = \sum_{j', k' \in N} W_{j', k'} C_{j-j', k-k'}^n$$

where W is a matrix of weights.

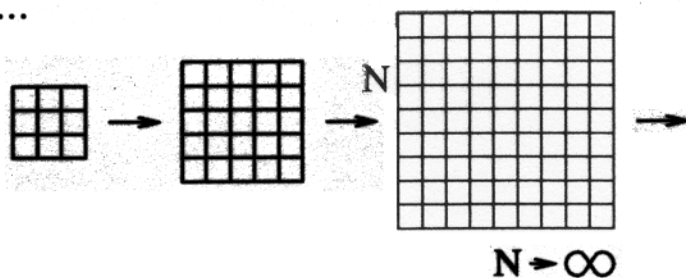
The update step is carried out by thresholding at the level λ :

$$C_{jk}^{n+1} = \begin{cases} 1 & \text{if } S_{jk} \geq \lambda \\ 0 & \text{otherwise} \end{cases}$$

Connection to Cellular Automata

In an attempt to reduce grid effects and to obtain a curvature contribution to the motion, several authors have considered refining the lattice and taking larger and larger neighborhoods.

In the limit as we refine the lattice and take larger and larger neighborhoods...



the summation step leads to a convolution:

$$\chi * K(x) = \int_{R^d} K(x - y)\chi(y)dy.$$

Thus in the continuum limit, threshold growth becomes

Convolution Generated Motion:

Find the updated region R^{new} according to

$$R^{new} = \{x : \chi * K(x) > \lambda\}$$

for some threshold value λ and some convolution kernel K