

Published in final edited form as:

Int J Comput Vis. 2005 November ; 65(1-2): 5–27. doi:10.1007/s11263-005-3849-9.

On the Evolution of Vector Distance Functions of Closed Curves

MARC NIETHAMMER, PATRICIO A. VELA, and ALLEN TANNENBAUM

Georgia Institute of Technology, School of Electrical and Computer Engineering, 777 Atlantic Drive NW, Atlanta, GA 30332-0250

Abstract

Inspired by the work by Gomes et al., we describe and analyze a vector distance function approach for the implicit evolution of closed curves of codimension larger than one. The approach is set up in complete generality, and then applied to the evolution of dynamic geometric active contours in \mathbb{R}^4 (codimension three case). In order to carry this out one needs an explicit expression for the zero level set for which we propose a discrete connectivity method. This leads us to make connections with the new theory of cubical homology. We provide some explicit simulation results in order to illustrate the methodology.

Keywords

vector distance function; level set methods; dynamic active contours

1. Introduction

Recently there has been an increased interest in evolving objects of codimension larger than one, with applications ranging from geometric optics (Burchard et al., 2001; Min, 2004; Osher et al., 2002), to image processing (Lorigo et al., 1999), and to geometric optimization (Pottman and Hofer, 2003). Theoretical aspects of these evolutions have been investigated in Ambrosio and Soner (1996), Evans and Spruck (1991, 1992a,b, 1995), Gomes et al. (2001) and Slepcev (2003).

This paper describes an approach based on Gomes and Faugeras (2001a, 2000b) and Gomes et al. (2001), for the implicit evolution of (possibly multiple) closed curves with codimension greater than one. The usual level set approach (Osher and Fedkiw, 2001; Sethian, 1999), valid for the evolution of hypersurfaces, is difficult to apply in higher codimensions. Indeed, in the codimension one case, there is a clearly defined interior and a clearly defined exterior of the given object of interest (if the surface is closed). In this case, a signed distance function can be used for a level set implementation of the evolution equation.

On the other hand, for objects of larger codimension, this is no longer the case. A possible remedy to this problem is to evolve an unsigned distance function. However, this is numerically challenging since numerical dissipation causes the zero level set to drift away from zero requiring the detection of points of minimal distance to extract an approximation to the zero level set. Closely related to the evolution of an unsigned distance function, Ambrosio and Soner (1996) analyze mean curvature motion in arbitrary codimension. Here, the evolving surface of codimension k in \mathbb{R}^d is surrounded by a family of hypersurfaces,

where the normal velocity is given by the sum of the geometrically relevant (the $d - k$ smallest) principal curvatures. Existence and uniqueness of a weak solution is established.

Lorigo et al. (1999) use the ideas of Ambrosio and Soner for vessel segmentation based on magnetic resonance angiography images, a codimension two problem. Classical numerical schemes for level set evolutions can be used since a tube (an ε -level set) is evolved, thus generating an “artificial” inside and outside. A disadvantage of this method is the occurrence of “fattening,” which does not allow for straightforward topological changes (Osher and Fedkiw, 2001) since curves can develop interiors. (See Burchard et al., 2001 for an example of fattening and (Bellettini et al., 1998) for an analysis of the phenomenon.)

Another possibility for the evolution of manifolds of arbitrary codimension is to represent a manifold of codimension k by the intersection of k scalar functions with non-vanishing gradient on the surface of the manifold (e.g., k signed distance functions with the intersection of their respective zero level sets representing the desired manifold). This approach was also proposed by Ambrosio and Soner, but not further pursued since the resulting system of evolution equations is not straightforward to analyze (the theory of viscosity solutions is not available for such systems of equations). Nevertheless, this approach has been very successfully applied, in particular to problems in geometric optics (Burchard et al., 2001; Min, 2004; Osher et al., 2002), where the handling of topological changes is of no importance (as they do not occur in this setting Osher et al., 2002). However, if one is interested in topological changes these approaches may require global initializations, and it is not clear how to automatically initialize the scalar functions to guarantee proper topological behavior. This is of special importance when numerical efficiency is crucial (e.g., when it would be beneficial to employ a narrow-band approach) (Burchard et al., 2001).

An alternative approach, closely related to the evolution of an unsigned distance function, is the evolution of a vector distance function, which is the topic of the present work. In this scheme not only the distance to an object is known at any space point, but also the normal direction. For a vector distance function, there will always be a clearly defined zero level set (where vectors flow away from each other), there are no initialization problems and narrow-band approaches are possible. A vector distance function can easily be calculated from a distance function and vice versa. Unfortunately, the theory underlying the numerical methods for vector distance functions is still in its infancy, since one is dealing with discontinuous vector fields. Also, as for the level set intersection methods discussed above, there is no analog to the theory of viscosity solutions available for these systems of partial differential equations (Gomes and Faugeras, 2001a). In this paper, we restrict ourselves to the representation of closed curves. This will simplify the numerical implementation and facilitate a reinitialization procedure compensating for fattening artifacts occurring throughout the evolution process (due to numerical inaccuracies).

We now summarize the contents of this paper. In Section 2 we describe the main level set approaches, focusing on the vector distance function based approach. Section 3 discusses the vector distance function based level set method. Next in Section 4, we apply the methodology developed in Section 3 to the evolution of a normal geometric dynamic active contour. Simulation results are presented in Section 5, and finally we make some conclusions in Section 6.

2. Level Set Approaches

In this section, we outline some of the main methods for the evolution of manifolds of codimension greater than one via various level set approaches. We will then describe in

some detail the vector distance function method which we will subsequently apply to the problem of dynamic active contours which are represented as curves evolving in \mathbb{R}^4 .

Level set approaches to date are extremely versatile and based on solid mathematical foundations for codimension one problems (Osher and Fedkiw, 2001). Ambrosio and Soner extended the theory of level set evolutions to mean curvature flows in arbitrary codimensions in their seminal paper (Ambrosio and Soner, 1996). Specifically, they prove existence and uniqueness of weak solutions for the curvature evolution of surfaces of arbitrary codimension represented by a surrounding family of hypersurfaces. They also hint at the possibility of representing smooth surfaces by the intersection of the level sets of multiple scalar functions, but do not follow this path due to theoretical complications (it is not clear how to theoretically analyze the resulting system of equations).

The first approach (as employed by Lorigo et al. (1999) for the evolution of a smooth manifold \mathcal{M} of codimension $k > 1$ in \mathbb{R}^d , makes use of the nonnegative scalar auxiliary function $v: \mathbb{R}^d \mapsto \mathbb{R}^+$,

$$\mathcal{M} = \{x \in \mathbb{R}^d : v(x) = 0\},$$

which vanishes on \mathcal{M} and fulfills $\nabla v(x) \neq 0$ for $x \in \mathbb{R}^d \setminus \mathcal{M}$. For $x \notin \mathcal{M}$, but ε -close to \mathcal{M} , we can define the matrix

$$J(x) := \frac{1}{\|\nabla v(x)\|} P_{\nabla v(x)} \nabla^2 v(x) P_{\nabla v(x)},$$

where

$$P_p = I - \frac{p p^T}{\|p\|^2}, \quad p \neq 0,$$

is the orthogonal projection operator along the direction of the vector p and ∇^2 denotes the Hessian. According to Ambrosio and Soner (1996), the evolution described by the differential equation

$$u_t = F(\nabla u, \nabla^2 u),$$

with

$$F(p, A) = \sum_{i=1}^{d-k} \lambda_i (P_p A P_p)$$

such that

$$\lambda_1 (P_p A P_p) \leq \lambda_2 (P_p A P_p) \leq \dots \leq \lambda_{d-1} (P_p A P_p)$$

are the eigenvalues of $P_p A P_p$ orthogonal to p , represents the mean curvature evolution of \mathcal{M} where

$$\mathcal{M}_t = \{x \in \mathbb{R}^d : u(x, t) = 0\}.$$

Only the $d - k$ smallest eigenvalues of $J(x)$ orthogonal to $\nabla v(x)$ are related to the geometry of \mathcal{M} . The uniqueness and existence results obtained in Ambrosio and Soner (1996) can be extended to a general normal velocity

$$V = \mathcal{H} + \Pi_{\mathcal{M}}^N g(x, t),$$

where \mathcal{H} is the mean curvature vector, $g(x, t) \in \mathbb{R}^d$ is a given vector field and $\Pi_{\mathcal{M}}^N$ is the projection operator onto the normal space of \mathcal{M} . The evolution equation then becomes

$$u_t = F(\nabla u, \nabla^2 u) - \nabla u \cdot g.$$

This approach is mathematically well founded, however, fattening of the evolving manifold can occur (Bellettini et al., 1998; Burchard et al., 2001) and so the handling of topological merging is problematic. Furthermore, numerical extraction of the zero level set is not straightforward. This can be circumvented (at the cost of less theoretical insight) by evolving the intersection of isocontours of k scalar functions in a way consistent with the desired movement of the codimension k object, which is the second method proposed by Ambrosio and Soner (1996) (see Burchard et al., 2001; Osher and Fedkiw, 2003; Bertalmio et al., 1999) for codimension two, (Osher et al., 2002) for codimension three, and (Gomes and Faugeras, 2002b) for arbitrary codimensions). Given the k scalar functions $\alpha^i: \mathbb{R}^d \mapsto \mathbb{R}$, $0 < i \leq k$, the evolving object is implicitly described by the simultaneous evolution of

$$\alpha_t^i + x_t \cdot \nabla \alpha^i = 0, \quad 0 < i \leq k,$$

where x_t is the velocity vector. The object is represented (for example) by the intersection of the respective zero level sets of the k scalar functions α^i .

Two major questions arise from this formulation:

1. How are the scalar functions initialized? This has to be performed globally (over the whole computational domain) if well-behaved handling of topological changes is required. Narrow-banding approaches (Adalsteinsson and Sethian, 1995) cannot be used *per se* for this problem. Further, the representation of an object by intersection of multiple hypersurfaces is not unique.
2. How should the speed functions in the complete computational domain be determined for the scalar functions α^i if the desired velocities are only known at the intersection of the hypersurfaces? Based on these velocities, extension velocities have to be constructed (on the zero level sets, and in the interior of the domain).

To resolve these questions, a novel approach based on vector distance functions is introduced in (Gomes and Faugeras 2001a, and Gomes et al. 2001). Given a manifold \mathcal{M} ,

$$\delta(x) := \text{dist}(x, \mathcal{M})$$

is defined as the distance from point $x \in \mathbb{R}^d$ to the manifold \mathcal{M} . The vector distance function $\mathbf{u}(x)$ is then given as the derivative of the squared distance function (see Ambrosio and Soner, 1996; Pottman and Hofer, 2003) for details on the squared distance function)

$$\eta(x) := \frac{1}{2} \delta^2(x).$$

Thus

$$\mathbf{u}(x) = \nabla \eta(x) = \delta(x) \nabla \delta(x). \quad (1)$$

The vector distance function is an implicit representation of the manifold \mathcal{M} with

$$\mathcal{M} = \mathbf{u}^{-1}(0),$$

the intersection of the d hypersurfaces

$$u_i = 0, 1 \leq i \leq d.$$

The description is redundant, but unique.

The evolution equation for the manifold $\mathcal{M}(p, t)$, parameterized by p , becomes (using the notation of Gomes and Faugeras (2001a))

$$\mathcal{M}_t(p, t) = \Pi_{\mathcal{M}(p, t)}^N (\mathcal{D}(\mathcal{M}(p, t), t)) = \mathbf{V}(\mathcal{M}(p, t), t),$$

where $\mathcal{D}(x, t) = \dot{x}_t$ is a vector field defined on $\mathbb{R}^d \times \mathbb{R}^+$, determining the evolution speed at x . Note that we do not need the tangential component of the evolution equation since the evolution is performed in \mathbb{R}^d .

To evolve the manifold \mathcal{M} , a speed function must be constructed on the subspace of \mathbb{R}^d that contains the evolution of the manifold. This speed function should

1. maintain the vector distance function throughout the evolution, and
2. move the manifold \mathcal{M} appropriately.

It can be shown (see Gomes et al. (2001)) that the characteristic equation for the vector distance function $\mathbf{u}(x)$ is

$$(\mathbf{D}\mathbf{u})^T \mathbf{u} = \mathbf{u}, \quad (2)$$

where $\mathbf{D}\mathbf{u}$ denotes the Jacobian of \mathbf{u} . Taking the time derivative of Equation (2) and using the fact that $(\mathbf{D}\mathbf{u})^T = \mathbf{D}\mathbf{u}$ (Gomes et al., 2001)

$$\mathbf{D}\mathbf{b}\mathbf{u} = (\mathbf{I} - \mathbf{D}\mathbf{u}) \mathbf{b}, \quad (3)$$

where \mathbf{b} is the appropriate velocity for the vector distance function evolution with initial condition

$$\mathbf{b}(\mathcal{M}, t) = -\mathbf{V}(\mathcal{M}, t).$$

The overall evolution is then given by

$$u_t + (Du)^T b = 0,$$

which completes the vector distance function based approach.

The vector distance function to a given manifold \mathcal{M} is unique, and its initialization is straightforward. Significant reductions in computational cost are possible through narrow-banding techniques. A fattening phenomenon (similar to that found in Ambrosio and Soner (1996)) can be observed for vector distance function based level set approaches.

In what follows we restrict ourselves to the representation of closed one-dimensional curves in \mathbb{R}^d , i.e., we will be considering a codimension $d - 1$ problem. This restriction is beneficial since it allows for the construction of an explicit reinitialization method, which in turn alleviates the fattening problem, and will provide a good representation for the evolutions based on the equations of dynamic active contours. Dynamic active contours will be presented in Section 4.

3. Level Set Approach by Means of a Vector Distance Function Evolution

An implicit description of an object by a vector distance function is extremely versatile; there is no restriction regarding an object's shape. Specifically, objects with varying dimensions can be represented (Gomes and Faugeras, 2001a). Objects can also change dimension throughout the vector distance function evolution, in contrast to level set approaches using a signed distance function, where objects necessarily need to be closed (unless we are working within a bounded domain). The representational flexibility of vector distance functions is clearly a desirable property. However, it is not clear at this point how to devise numerical methods for this general case. To facilitate the numerical implementation, we restrict ourselves to the representation of possibly multiple closed curves.

We perform the following steps for our vector distance function based curve evolution. To evolve a curve according to the vector distance function approach,

- 0 Initialize the vector distance function.
1. Detect (a band around) the zero contour (i.e., $u^{-1}(0) = \mathcal{M}$).
2. Redistance the vector distance function outside of the zero band or (if necessary) explicitly on the whole computational domain.
3. Compute the vector field b on the band determined in (1).
4. Extend the b vector field to the whole domain, keeping the values of step (3) fixed.
5. Do an evolution step.
6. Go to step (1).

Steps (0)–(5) are described in the following subsections.

3.1. Initializing the Vector Distance Function

Assume the existence of a piecewise linear approximation of the closed curve to be represented. Let this approximation be

$$\mathcal{A} = \bigcup_{l_i \in \mathcal{L}} l_i,$$

where \mathcal{L} is the set of line segments constituting the piecewise linear approximation. Working on a discrete grid, we explicitly initialize¹ the vector distance function \mathbf{u} on a set, \mathcal{B} , γ -close to the piecewise linear approximation \mathcal{A} :

$$\mathcal{B} := \{x \in \Omega : \text{dist}(x, \mathcal{A}) < \gamma\}.$$

In the remainder of the computational domain Ω , i.e., $\Omega \setminus \mathcal{B}$, we can reinitialize by first computing the distance function δ to \mathcal{A} and then converting the distance function to the vector distance function. One way to compute this distance function δ on $\Omega \setminus \mathcal{B}$ is to use the reinitialization approach proposed by Sussman et al. (1994). There is no inside and outside when representing objects of codimension larger than one. This precludes the application of the scheme by Sussman et al. to initialize over the whole domain Ω (see Section 3.3 for more details).

3.2. Simple Zero Band Detection

As will be shown later, redistancing the vector distance function over the whole computational domain Ω is not straightforward. Since redistancing is not straightforward, neither is the extension of quantities (like the velocities). However, frequently it is sufficient to perform these operations away from the zero level set (Dupont and Liu, 2004), where standard methods known from the evolution of closed curves or surfaces of codimension one can be applied. This is the approach followed here. We simply do not redistance on the set $\widehat{\mathcal{B}}$ which is γ -close to the zero level set (in analogy to Section 3.1). The current vector distance function \mathbf{u} can be used to find the set $\widehat{\mathcal{B}}$,

$$\widehat{\mathcal{B}} := \{x \in \Omega : \|\mathbf{u}(x)\| < \gamma\}.$$

We will see in Section 3.7 that a more precise approximation for the zero level set is needed for the explicit reinitialization of the vector distance function field on $\widehat{\mathcal{B}}$. The value for γ needs to be chosen conservatively to make sure that the actual zero level set is contained in $\widehat{\mathcal{B}}$.

3.3. Redistancing the Vector Distance Field Away from the Zero Band

Inspired by Dupont and Liu (2003), the vector distance function is reinitialized only outside the zero band obtained as described in Section 3.2. Gomes and Faugeras (2001a) propose to minimize the functional

$$\frac{1}{2} \int_{\Omega} \|D\mathbf{u}^T \mathbf{u} - \mathbf{u}\|^2 dx. \quad (4)$$

By means of calculus of variations they derive the corresponding gradient descent flow. This is a flow that directly works with the vectors \mathbf{u} . Unfortunately, the numerical implementation is not straightforward (except in the case of simple geometries, e.g. lines).

¹For an efficient implementation this set has to be approximated, since we do not know the distance of a point x to \mathcal{A} beforehand.

Alternatively, redistancing is performed via the intermediate computation of a scalar distance function. Based on this reinitialized scalar distance field, we can then compute the vector distance field. This two-step approach has the advantage that standard algorithms can be used and iteration free solutions (e.g., fast marching) are possible². The current distance function $\delta(x)$ relates to the vector distance function by

$$\delta(x) = \|u(x)\|. \quad (5)$$

The distance function can be reinitialized on $\Omega \setminus \mathcal{B}$ for example by a fast marching approach (Sethian, 1999) or by a partial differential equation approach along the lines of Sussman et al. (1994). However, since we are not dealing with a signed distance function here, the approach by Sussman et al. (1994) will simplify to solving the equation

$$\delta_t = 1 - \|\nabla \delta\|. \quad (6)$$

Given the reinitialized distance function δ , the reinitialized vector distance function can be determined by solving³ for u in Eq. (1).

3.4. Computing the Velocity Field on the Zero Band

It is sufficient to compute a velocity vector for every point in $\widehat{\mathcal{B}}$. From Gomes and Faugeras(2000a) the identity $Du(x)t = 0$ holds for every element t of the tangent space to the curve. The symmetry of (Du) in the evolution equation

$$u_t + (Du)^T b = 0$$

means that the velocity vector b need not be projected onto the normal space of the curve. For example, for a mean curvature evolution, the velocity vector b is given by the mean curvature vector which can be computed (Gomes and Faugeras, 2001a) (for a one-dimensional curve) as

$$\mathcal{H}(x) = -(\nabla u(x)),$$

the component-wise Laplacian of u . Section 4 contains the corresponding expressions for the normal dynamic active contour.

3.5. Extending the Velocity Field

We now show that the velocity field b satisfies Equation (18), if it is normal to \mathcal{M} and extended in the normal direction to \mathcal{M} . Assume b is extended in the normal direction to \mathcal{M} , i.e., given any point $x \in \Omega$, b will be constant in the u direction. Then,

$$Dbu = 0$$

and Eq. (3) reduces to

$$b = Dub. \quad (7)$$

²The overall scheme is then an iteration free scheme to produce a vector distance function.

³Choose the gradient direction dictated by the numerical scheme.

We know that

$$Du(x) = D(\delta(x) D\delta(x)) = D\delta(x) D\delta(x) + \delta(x) D(D\delta(x)). \quad (8)$$

But

$$\delta(x) D(D\delta(x)) \mathbf{b} = 0, \quad (9)$$

since the gradient of $\delta(x)$ is constant along the normal direction, and \mathbf{b} and the normal direction are linearly dependent by assumption. Combining (7–9) yields

$$\mathbf{b} = (\nabla \delta(x)^T \mathbf{b}) \nabla \delta(x),$$

which holds because \mathbf{b} is assumed normal to \mathcal{M} .

By Section 3.4, only the normal direction of \mathbf{b} matters for the evolution of \mathbf{u} . Thus, instead of solving Eq. (3) directly, it is sufficient to extend the components of \mathbf{b} along the normal direction to \mathcal{M} . Since only the normal component of \mathbf{b} to \mathcal{M} matters for the evolution, \mathbf{b} need not be projected onto the normal space of \mathcal{M} before extending it in the normal direction of \mathcal{M} .

3.6. Evolving the Vector Distance Function

There are numerical advantages to evolving vector distance functions over evolving distance functions directly. Whereas the vector distance function clearly defines the zero level set at every point in time (vectors are emanating from the zero level set), the zero level set of the distance function is not so easy to find. In case of the distance function it is unreasonable to expect to find a level set that is exactly zero. Instead, locating the zero level set would require searching for distance minima or locations of diverging gradients of the distance function. The latter essentially goes back to the idea of the vector distance function. Also, numerical algorithms usually cause dissipation. For the distance function this means, in the extreme case, that the “zero level set” drifts away from zero over time. This drift is not desirable.

We assume a velocity field \mathbf{b} in Ω as discussed in Section 3.5. We evolve the vector distance function as⁴

$$\mathbf{u}_t + (Du)^T \mathbf{b} = 0 \quad (10)$$

Increased numerical accuracy can for example be achieved through back and forth error compensation and correction (Dupont and Liu, 2003). If \mathcal{L}_h is the numerical solution operator of Eq. (10) the backward error compensation method is given by the following three steps:

1. Solve forward: $\tilde{\mathbf{u}}^{n+1} = \mathcal{L}_h \mathbf{u}^n$.
2. Solve backward: $\mathbf{u}_1^n = \mathcal{L}_h^{-1} \tilde{\mathbf{u}}^{n+1}$.
3. Solve with error compensation: $\mathbf{u}^{n+1} = \mathcal{L}_h \left(\mathbf{u}^n + \frac{1}{2} (\mathbf{u}^n - \mathbf{u}_1^n) \right)$.

⁴For vector distance function evolutions: $\mathbf{b} = -\mathbf{v}$, where \mathbf{v} is the speed of the zero level set.

We show an example of backward error compensation in Section 5, but refrain from its use for the remainder to save on computational complexity (this scheme increases the numerical complexity roughly by a factor of three).

3.7. Redistancing the Vector Distance Function Field

Due to numerical errors, the evolving vector field \mathbf{u} will drift away from the class of vector distance functions over time. Close to the zero level set, the vectors will fail to be perpendicular to the curve being evolved and the approach of Subsection 3.3 will no longer suffice.

In this case, the vector field has to be reinitialized on the zero band (or alternatively, we could constantly reinitialize as we go). Instead of using the flow based on the energy functional (4), we could directly solve the characteristic equation

$$(D\mathbf{u})^T \mathbf{u} - \mathbf{u} = 0$$

numerically. Inspired by (6) it seems reasonable to evolve

$$\mathbf{u}_t + (D\mathbf{u})^T \mathbf{u} = \mathbf{u}$$

to steady state. Since $D\mathbf{u}$ is symmetric this is equivalent to

$$\mathbf{u}_t + (D\mathbf{u}) \mathbf{u} = \mathbf{u}. \quad (11)$$

Equation (11) has a particularly simple structure. It is a multi-dimensional transport equation with a right hand source term. Unfortunately, solving Eq. (11) numerically is not straightforward. Only in the most trivial cases, i.e., when representing one single line, will there be no solution shocks. Generically, the solution will have discontinuous shocks.

To get a feel for this equation and its associated problems, consider the one-dimensional case. Equation (11) becomes

$$u_t = u(1 - u_x), \quad (12)$$

a one-dimensional Burgers' equation with source term. The steady state for this equation is either $u_x = 1$ or $u = 0$. However, this is only true if we are implicitly representing one single point. Otherwise, we expect a discontinuous solution that should be interpreted in the weak sense. If the equation is discretized using a conservative scheme (e.g., volume of fluids) the singled-out solution is not necessarily the desired solution. This can easily be seen in the one-dimensional case by looking at the shock-speed given by the Rankine-Hugoniot condition. Due to the additional source term on the right hand side of the Burgers'-like equation, a derivation of the shock speed is instructive. To derive the shock speed, we consider a small time interval Δt for which the shock speed s is approximately constant (Leveque, 2002). During that time period the shock travels the distance $\Delta x = s\Delta t$. Assuming the discontinuous states across the shock are u_l and u_r , we derive the shock speed based on the conservation form of

$$u_t + f(u)_x = u,$$

which yields Eq. (12) when $f(u) = \frac{1}{2}u^2$. We obtain (see Fig. (1))

$$\int_{x_1}^{x_1+\Delta x} u(x, t_1+\Delta t) dx - \int_{x_1}^{x_1+\Delta x} u(x, t_1) dx = \int_{t_1}^{t_1+\Delta t} f(u(x_1, t)) dt - \int_{t_1}^{t_1+\Delta t} f(u(x_1+\Delta x, t)) dt + \int_{t_1}^{t_1+\Delta t} \int_{x_1}^{x_1+\Delta x} x_1 + \Delta x u(x, t) dx dt. \quad (13)$$

Since the value for u is constant in each of the triangles of Fig. 1, Equation (13) simplifies to

$$\Delta x (u_l - u_r) = \Delta t (f(u_l) - f(u_r)) + \frac{1}{2} \Delta x \Delta t (u_l + u_r) + \mathcal{O}(\Delta t^2).$$

With $\Delta x = s \Delta t$,

$$\begin{aligned} s(u_l - u_r) &= f(u_l) - f(u_r) + \frac{1}{2} s \Delta t (u_l + u_r) + \mathcal{O}(\Delta t) \\ &= f(u_l) - f(u_r) + \mathcal{O}(\Delta t), \end{aligned}$$

resulting in an expression for the shock speed s :

$$s = \frac{f(u_r) - f(u_l)}{u_r - u_l}. \quad (14)$$

The latter expression is independent of the right hand-side source term (it is negligible for this infinitesimal time interval). This is the classical Rankine-Hugoniot condition for scalar conservation laws. Substituting $f(u) = \frac{1}{2}u^2$ into Eq. (14) yields

$$s = \frac{1}{2} (u_l + u_r),$$

the shock speed for Burgers' equation. For the one-dimensional case, discontinuities will only arise in between the represented points (the zero level set). For the representation to be a true vector distance function the left-hand and the right-hand side limits of the vector distance function magnitude at the discontinuity have to be equal. Since the numerics will introduce errors, this exact balance will not be stable. Redistancing performed by solving Eq. (12) will favor high magnitudes of the vector distance function. The vector distance function will converge to a vector distance function of a single point (whichever one was the most dominating). Figure 2 shows the shock speeds for two vector based level set representations of a two-point scenario. In both cases the shock does not move in the desired direction.

To get a feeling for the fundamental difference between the signed distance function based and the vector distance function based approaches, it is instructive to briefly review the rationale behind the numerics for the level set evolution based on a signed distance function. Here, the slope of the signed distance function is conserved (Sethian, 1999). In one dimension, the evolution equation is given as

$$\Phi_t + F \sqrt{\Phi_x^2} = \Phi_t + F \|\Phi_x\| = 0, \quad (15)$$

where F is the speed function. Differentiation of Eq. (15) gives

$$(\Phi_t)_x + (F \|\Phi_x\|)_x = (\Phi_x)_t + (F \|\Phi_x\|)_x = 0.$$

Substituting $w = \Phi_x$ we get

$$w_t + (F\|w\|)_x = w_t + H(w)_x = 0,$$

which is in conservation form for w , the slope. Here, $H(\cdot)$ is the flux function. However, a numerical solution to Φ instead of w is sought. Thus, the Hamilton-Jacobi equation

$$\Phi_t + H(\Phi_x) = 0 \quad (16)$$

needs to be solved using a numerical approximation of $H(\cdot)$. Equation (16) evolves a continuous function Φ as opposed to a discontinuous vector field \mathbf{u} .

3.7.0.1. Explicit Redistancing of the Vector Distance Function—Since it is not immediately obvious how to devise a partial differential equation based reinitialization strategy, reinitialization is done by construction. Doing so requires an explicit expression for the zero level set, which is also useful for visualization purposes. The remainder of this section proposes a particle based method and a method based on discrete connectivity. Both methods are computationally expensive; more efficient methods would be desirable. Fortunately, it is not necessary to reinitialize the vector distance function after every iteration step.

3.7.1. Particle Based Explicit Redistancing of the Vector Distance Function—

Given a sampling set \mathcal{S} of a (discrete) domain containing the zero level set (e.g.,

$\mathcal{S} = \mathcal{S}(\widehat{\mathcal{B}})$), move each particle in the sampling set along the vector distance field:

$$\mathbf{x}_t = -\mathbf{u}(\mathbf{x}), \mathbf{x}(0) = \mathbf{x}_0, \forall \mathbf{x}_0 \in \mathcal{S}.$$

The particles will converge to the zero level set (possibly suffering from fattening). A sufficiently dense sampling will guarantee a good representation of the zero level set. Denoting the set of points to which the particles converge by \mathcal{S}_c , the vector distance function on $\widehat{\mathcal{B}}$ can be reinitialized by explicitly computing $\text{dist}(\mathbf{x}, \mathcal{S}_c), \forall \mathbf{x} \in \widehat{\mathcal{B}}$.

3.7.2. Discrete Connectivity Based Redistancing of the Vector Distance

Function—The discrete connectivity based approach hinges on a discrete approximation of the zero level set and a piecewise linear approximation thereof.

3.7.2.1. Thinning: To construct the discrete approximation of the zero level set, begin with a discrete representation of the zero band (as proposed in Section 3.2). This discrete approximation is then thinned. Ideally, we would obtain a $(3^d - 1)$ -connected (where d is the space dimension) approximation of the zero level set where this approximation is composed of a union of sets representing discrete simple closed curves (Thürmer, 2003)⁵ whose union does not violate the property of every d-xel having exactly two neighbors. If the represented curves are sufficiently distant from each other, this two-neighbor-property will not be violated. However, if the represented curves are sufficiently close it may be, as illustrated in Fig. 3.

⁵For our purpose, a *discrete simple closed curve* γ_d in \mathbb{Z}^d is a finite subset of \mathbb{Z}^d , such that γ_d is $(3^d - 1)$ -connected and each point of γ_d is adjacent to exactly two other points of γ_d .

Topology preserving thinning is frequently used to determine discrete skeletons of discretely represented objects. A key issue is the notion of a *simple point*: simple points are points of discrete sets that do not alter the topology of the represented object upon removal. Testing for simple points gets increasingly complex for higher dimensions. The two-dimensional case is straightforward. The three-dimensional case is well studied. Higher-dimensional cases have not received much attention (Gau and Kong, 2002; Kong, 1997). However, this is the case relevant to our application. For now, assume the existence of a test for the simple point property in higher dimensions (Section 3.7.3 will explain the method used).

To decide if a point is simple, it suffices to check if its removal changes the topology within its $(3^d - 1)$ neighborhood. To utilize the additional directional information encoded in the vector distance function for the thinning algorithm, define the local $(3^d - 1)$ neighborhood $\overline{\mathcal{N}}_{(3^d-1)}(\mathbf{p})$ of a point \mathbf{p} as

$$\overline{\mathcal{N}}_{(3^d-1)}(\mathbf{p}) = \left\{ \mathbf{x} \in \mathcal{N}_{(3^d-1)}(\mathbf{p}) : \frac{(\mathbf{x} - \mathbf{p})^T}{\|\mathbf{x} - \mathbf{p}\|} (\mathbf{u}(\mathbf{x}) - \mathbf{u}(\mathbf{p})) < \nu \right\},$$

and base the decision for a simple point on this neighborhood. As a consequence, the modified thinning algorithm will no longer be topology preserving. Instead it will be allowed to break undesired connectivities (see Fig. 4 for a two-dimensional example) based on the projection of the vector distance function vectors on connecting line segments between two points.

The thinning is performed by successive removal of simple points until there are no simple points left. The simple points with maximal vector distance norm are removed first.

3.7.2.2. Constructing a Piecewise Linear Approximation: Given the discrete approximation of the zero level set, we want to construct a piecewise linear approximation by means of the discrete connectivity information, i.e., each line in the piecewise linear approximation will correspond to a $(3^d - 1)$ connected pair of d-xels. The discrete connectivity induces a graph $(\mathcal{V}, \mathcal{E})$ over the discrete approximation \mathcal{Z} of the zero level set, where the vertex set \mathcal{V} is the set of points in \mathcal{Z} and the edge set \mathcal{E} is given by the discrete connectivity information. (See Fig. 5 for an illustration.) We are interested in simple cycles of this graph that are consistent with the definition for discrete, simple closed curves. Fig. 6 shows some possible simple cycles of the graph associated with a simple two-dimensional discrete approximation of a zero level set of two circular objects. None of these exemplary simple cycles are valid representatives for a discrete, simple closed curve.

Initial Candidates: In order to single out sensible candidates for the discrete, simple closed curves in which we are interested, we introduce a measure for edge deviation from the edge contained in the sought-after piecewise linear approximation of the zero level set. Specifically, given the two points \mathbf{x}_0 and \mathbf{x}_1 (the vertices of the edge e), define the two line segments

$$\begin{aligned} \mathbf{x}(p) &= \mathbf{x}_0 + (\mathbf{x}_1 - \mathbf{x}_0)p, \quad \text{and} \\ \mathbf{x}'(p) &= \mathbf{p}_0 + (\mathbf{p}_1 - \mathbf{p}_0)p, \end{aligned}$$

where $p \in [0, 1]$, and

$$\mathbf{p}_0 = \mathbf{x}_0 - \mathbf{u}(\mathbf{x}_0) \quad \text{and} \quad \mathbf{p}_1 = \mathbf{x}_1 - \mathbf{u}(\mathbf{x}_1).$$

The distance is then defined as (see Fig. (7) for an illustration):

$$\begin{aligned} d(x, x') &= d(e) := \int_0^1 \| (x - x')(p) \|^2 dp \\ &= \|u(x_0)\|^2 + u(x_0)^T (u(x_1) - u(x_0)) + \frac{1}{3} \|u(x_1) - u(x_0)\|^2. \end{aligned}$$

Define the set of illegal starting edges \mathcal{E}_i as

$$\mathcal{E}_i := \{e \in \mathcal{E} : e \text{ is out-edge of } v \in \mathcal{V}, \exists \text{ simple cycle } \mathcal{S} : v \in \mathcal{S} \text{ and } |\mathcal{S}|=3\},$$

i.e., the set of edges that are part of simple cycles of length three. These are edges at positions where the discrete curve “ceases to be simple” and a point has more than two neighbors.

The following heuristic algorithm is used to find discrete, simple closed curve candidates:

- 0 Set the set of uncovered edges to $\mathcal{E}_u = \mathcal{E} \setminus \mathcal{E}_i$.
1. While $\mathcal{E}_u \neq \emptyset$: (repeat steps (2)–(4)):
2. Find $e \in \mathcal{E}_u : d(e) \leq d(e') \forall e' \in \mathcal{E}_u, e \neq e'$.
3. Find all simple cycles containing e that represent a discrete, simple closed curve and put them in \mathcal{S}_e (the set of these cycles).
4. If $\mathcal{S}_e = \emptyset$ remove e from \mathcal{E}_u . Otherwise, the smoothest⁶ of the cycles in \mathcal{S}_e is the desired, discrete, simple closed-curve candidate whose edges are removed from \mathcal{E}_u .

Measuring Smoothness: The smoothness of a cycle depends on the likelihood of a cycle to follow a path across a vertex that has more than two neighbors. Specifically, if \mathcal{P} is the set of all paths through a vertex $p \in \mathcal{V}$, define the path-likelihood as

$$l(p_i) = e^{-\frac{c_i(p)^2}{2\sigma^2}}, p_i \in \mathcal{P},$$

where c_i is an approximation to the curvature at p for the path p_i and the probability P of a cycle taking the path p_i at vertex p as

$$P(p_i) = \frac{l(p_i)}{\sum_i l(p_i)}.$$

The smoothness s of a cycle C is then defined to be

$$s(C) := 1 - \prod_{p_i \in C} P(p_i).$$

The smaller $s(C)$, the smoother the cycle C is. Fig. (8) illustrates this smoothness computation. Starting with an edge emanating from the light gray block at the bottom left of Fig. 8(a), the three cycles given in Figs. 8(b–d) are possible. Due to symmetry, it suffices if we compute curvature approximations at the point p_a denoted by the asterisk. The four

⁶See the subsequent subparagraph on how to measure the smoothness of such a cycle.

hatched blocks are used for the discrete curvature computations. We assume that the diagonal distance between points is one. In case (b), curvature $\alpha(p_a) = 0$. In the cases (c) and (d),

$$c(p_a) := \frac{y_0 - 2y_1 + y_2}{(\Delta y)^2} = \frac{0 - 2\frac{1}{\sqrt{2}} + 0}{\left(\frac{1}{\sqrt{2}}\right)} = -2\sqrt{2}.$$

Using $\sigma = 2$, the likelihoods at the crossing points are

$$l(p_a) = \begin{cases} 1 & \text{for case (b),} \\ \frac{1}{e} & \text{for cases (c) and (d).} \end{cases}$$

The corresponding probabilities thus become

$$P(p_a) = \begin{cases} \frac{e}{e+2} & \text{for case (b),} \\ \frac{1}{e+2} & \text{for cases (c) and (d),} \end{cases}$$

which yields to the smoothness values

$$s(C) = \begin{cases} 1 - \left(\frac{e}{e+2}\right)^2 & \text{for case (b),} \\ 1 - \left(\frac{1}{e+2}\right)^2 & \text{for cases (c) and (d).} \end{cases}$$

Since s is smallest in case (b), the corresponding cycle is the smoothest.

Refining the Piecewise Linear Approximation: Once a cycle has been determined, its piecewise linear approximation is still relatively crude. To refine the approximation (keeping the connectivity at the same time), successively split a line segment into two line segments. Given the two end-points p_0 and p_1 of a line segment, define

$$p_m := \frac{1}{2}(p_0 + p_1) - u(p_0 + p_1).$$

The line segment (p_0, p_1) is then replaced by the two line segments (p_0, p_m) and (p_m, p_1) .

This process can be repeated multiple times if necessary. Due to numerical inaccuracies in the vector distance function oversampling may result in a rugged-looking curve. To increase curve smoothness approximate this over-sampled piecewise linear approximation to the zero level set by a least squares quadratic spline (Dierckx, 1993) or a least squares piecewise linear approximation. It is then straightforward, but relatively computationally costly, to redistance the vector distance function based on the obtained piecewise linear approximation.

Explicit Redistancing: Given a line segment L defined by its two endpoints p_0 and p_1 , define the vector distance u of a point p to the line segment L as:

$$u(p, L) = p - p_0 - \left((p - p_0)^T t\right) t \quad \text{if} \quad (p - p_0)^T t \in [0, \|p_1 - p_0\|],$$

where

$$t = \frac{\mathbf{p}_1 - \mathbf{p}_0}{\|\mathbf{p}_1 - \mathbf{p}_0\|}.$$

Otherwise,

$$\mathbf{u}(\mathbf{p}, L) = \begin{cases} \mathbf{p} - \mathbf{p}_0 & \text{if } \|\mathbf{p}_0 - \mathbf{p}\| \leq \|\mathbf{p}_1 - \mathbf{p}\|, \\ \mathbf{p} - \mathbf{p}_1 & \text{otherwise.} \end{cases}$$

The reinitialized vector distance function at a point \mathbf{x} is then approximately

$$\mathbf{u}(\mathbf{x}) = \underset{\mathbf{u}(\mathbf{x}, L), L \in \mathcal{L}}{\operatorname{argmin}} \|\mathbf{u}(\mathbf{x}, L)\|.$$

3.7.3. Detecting Simple Points—Detecting simple points gets increasingly complex with higher space dimensions. Since the potential space dimensions may be higher than three, it is desirable to use a method for simple point detection applicable in arbitrary space dimensions. Previous work has focused on specific space dimensions: see for example (Bertrand, 1996; Gau and Kong, 2002; Klette, 2003; Kong, 1997).

Our approach is based on cubical homology and is not restricted to a specific space dimension. Cubical homology is ideally suited to digital images, due to its ability to handle voxels or pixels directly. Whereas homology is by now a standard tool of algebraic topology (Massey, 1991), cubical homology is more recent (Kaczynski et al., 2004; Kalies et al., 1999). Homology seeks to count holes in a topological space. For three-dimensional image data for example, three non-trivial homology groups H_0 , H_1 and H_2 exist. The number of connected components, tunnels and voids present in the image are given by the Betti numbers β_0 , β_1 and β_2 respectively; where β_i is the rank of the homology group H_i . Homology is a combinatorial theory, i.e., it can be computed by decomposing the space into a finite number of units. In the traditional simplicial homology, these units are simplices (a simplex is a hypertetrahedron, it is the simplest possible polytope for a given space dimension, e.g., a line in one dimension, a triangle in two dimensions, etc.). In the cubical homology these units are d-xels and also their respective vertices, edges and higher-dimensional faces, they are called elementary cubes.

Formally, an *elementary cube* Q is given by the finite product (Kaczynski et al., 2001)

$$Q = I_1 \times I_2 \times \cdots \times I_d \subset \mathbb{R}^d, \quad (17)$$

where I_i is either a singleton (*degenerated*) interval $I = [l, l] = [l]$ or an interval of unit length $I = [l, l + 1]$ for some $l \in \mathbb{Z}$. The number of non-degenerate components in Q is called the *dimension* of Q ($\dim Q$). If a set $X \subset \mathbb{R}^d$ can be written as a finite number of elementary cubes, it is called a *cubical* set. The set of all elementary cubes is denoted by \mathcal{K} , and the set of all elementary cubes Q in \mathbb{R}^d with $\dim Q = k$ by \mathcal{K}_k , for $k \in \mathbb{N}$. In three dimensions, the set of all elementary cubes \mathcal{K} consists of all vertices, edges, faces and voxels.

Definition 1. Let $X \subset \mathbb{R}^d$ be a cubical set. Let

$$\begin{aligned}\mathcal{H}(X) &:= \{Q \in \mathcal{H} \mid Q \subset X\} \quad \text{and} \\ \mathcal{H}_k(X) &:= \{Q \in \mathcal{H}(X) \mid \dim Q = k\}.\end{aligned}$$

To pass from the combinatorial structure of the elementary cubes, e.g., the collection of voxels, to the algebraic structure of homology groups, one constructs the free Abelian group (i.e., a commutative group that has a basis) of k -chains, $C_k(X)$ by declaring each element of $\mathcal{H}_k(X)$ to be a distinct generator (or basis element). Let C_k denote $C_k(\mathbb{R}^d)$.

Given $k \in \mathbb{Z}$, the cubical boundary operator

$$\partial_k: C_k \rightarrow C_{k-1} \quad (18)$$

is the group homomorphism⁷ defined on every elementary cube $Q \in \mathcal{H}_k$ as the alternating sum of its $(k-1)$ -dimensional faces. Due to linearity this boundary operator extends to all k -chains. A k -chain $z \in C_k(X)$ is called a *cycle* in X if $\partial_k z = 0$. A k -chain $z \in C_k(X)$ is called a *boundary* in X if there exists a $c \in C_{k+1}(X)$ such that $\partial_{k+1} c = z$. The set of all cycles and the set of all boundaries in X form subgroups in $C_k(X)$ and are given by⁸

$$Z_k(X) := \ker \partial_k^X = C_k(X) \cap \ker \partial_k \quad (19)$$

$$B_k(X) := \text{image } \partial_{k+1}^X = \partial_{k+1}(C_{k+1}(X)) \quad (20)$$

respectively.

Definition 2. The k -th cubical homology group of X is the quotient group

$$H_k(X) := Z_k(X) / B_k(X).$$

The Betti numbers β_k are then given as

$$\beta_k(X) := \text{rank}(H_k(X)). \quad (21)$$

The elements of $H_k(X)$ are called the k -generators of X . The homology groups are computed⁹ as (for example) described in Kalies et al. (1999). If N is the $(3^d - 1)$ -neighborhood of the d -xel \mathbf{x} , \mathbf{x} is a simple point if Niethammer et al. (2004a)

$$\beta_0(N) = 1, \beta_i(N) = 0, \quad 0 < i \leq 4, i \in \mathbb{N}.$$

See Niethammer et al. (2004a) for the simple point condition for arbitrary dimensions d

For our thinning purpose, it is also useful to allow for topological changes. One useful change for finding a discrete approximation to the zero level set is to allow for the “piercing” of discrete planes, i.e., if the removal of an d -xel changes the topology of the $(3^d$

⁷A map $f: G \rightarrow G'$ between the two Abelian groups G and G' is called a *homomorphism* (Kaczynski et al., 2004) if $f(g_1 + g_2) = f(g_1) + f(g_2)$ for all $g_1, g_2 \in G$.

⁸ $\ker f$ denotes the kernel of f , i.e., $\ker f = f^{-1}(0)$.

⁹Software to compute the Betti-numbers of a cubical complex is freely available (<http://www.math.gatech.edu/chom>).

– 1) neighborhood from $\beta_0 = 1, \beta_i = 0, i > 0$ to $\beta_0 = 1, \beta_1 = 1, \beta_i = 0, i > 1$, we allow to this removal occur.

4. Four-Dimensional Evolution for the Normal Geometric Dynamic Active Contour

One of the key examples for the proposed vector distance methodology is the evolution of a normal dynamic active contour (see Section 5 for results). We review some of the background material for this following (Niethammer et al., 2004b), to which we refer the reader for all relevant details.

Specifically, consider the evolution of closed curves of the form $\mathcal{C}: S^1 \times [0, \tau) \mapsto \mathbb{R}^2, \tau \in \mathbb{R}^+$ in the plane, where $\mathcal{C} = \mathcal{C}(p, t)$ and $\mathcal{C}(0, t) = \mathcal{C}(1, t)$ (Grayson, 1987), with $t \in [0, \tau)$ being the time, and $p \in [0, 1]$ the curve's parametrization on the unit circle S^1 .

The governing equations for the dynamic active contour model are derived by computing the Euler-Lagrange equation associated with the action integral

$$\mathcal{L} = \int_{t=t_0}^{t_1} \int_{p=0}^1 \left(\frac{1}{2} \mu \|\mathcal{C}_t\|^2 - g \right) \|\mathcal{C}_p\| dp \quad dt,$$

resulting in

$$\mu \mathcal{C}_{tt} = -\mu (\mathcal{T} \cdot \mathcal{C}_{ts}) \mathcal{C}_t - \mu (\mathcal{C}_t \cdot \mathcal{C}_{ts}) \mathcal{T} - \left(\frac{1}{2} \mu \|\mathcal{C}_t\|^2 - g \right) \kappa \mathcal{N} - (\nabla g \cdot \mathcal{N}) \mathcal{N}, \quad (22)$$

where \mathcal{N} is the unit inward normal, $\mathcal{T} = \frac{\partial \mathcal{C}}{\partial s}$ is the unit tangent vector to the curve, $\kappa = \mathcal{C}_{ss} \cdot \mathcal{N}$ denotes curvature, s is the arclength parameter (do Carmo, 1976), g is a potential function and μ is related to the mass of the curve. In comparison to the classical active contour, time in (22) is not an artificial time parameter introduced to solve an Euler-Lagrange equation, but instead appears naturally in Eq. (22), is physically motivated and augments the state space by a velocity vector for every point on the evolving curve.

It can be shown that the solution of Eq. (22) can only have a nonvanishing tangential velocity if its initial condition has a nonvanishing tangential velocity (Niethammer et al., 2004b). Assume that the initial condition is free of tangential velocities. Eq. (22) simplifies to

$$\begin{aligned} \mathcal{C}_t &= \beta \mathcal{N}, \\ \beta_t &= \left(\frac{1}{2} \beta^2 + \frac{1}{\mu} g \right) \kappa - \frac{1}{\mu} \nabla g \cdot \mathcal{N}, \end{aligned} \quad (23)$$

the evolution equation for normal dynamic geometric curve evolution (with β being the speed in the normal direction).

Although the evolution equations for the normal dynamic geometric curve evolution seem to only require a curve evolving in three dimensional space (for the position in the plane and the normal velocity for every point on the curve—the projection of the space curve onto the image plane then recovers the geometric shape of the evolving planar curve) this is not sufficient in general when implementing the evolution for multiple curves simultaneously. In certain cases (i.e., when the projections of curves overlap in the image plane) there is no clear inside and outside of the projected curve. Then it is not apparent how to assign a unique normal vector to every point of the projected curves in the image plane without the

normal vector exhibiting a discontinuity (when traced along an individual curve). Also, even if such a normal vector would be given, it is not desirable to perform the topological merging and splitting based on this representation, i.e., based on the position and the normal velocity coefficient. Indeed, if we use the unit inward normals of two curves, two curves will not necessarily merge even though they coincide with position and velocity, because the normal velocity coefficient β can (and in this case will) differ in sign (see Fig. (9) for an illustration). It is thus more desirable to perform the level set implementation in a four-dimensional space (this is a codimension three problem) so that merging and splitting is performed based on the real velocity vector. We call this the full level set approach.

Previous work (Niethammer et al., 2004b) was not concerned with the propagation of the curves in this higher-dimensional space. Instead a partial level set approach was used, where curves in the image plane are represented by a level set function and the normal speed is simply propagated along with the curves. This guarantees dynamic curve propagation according to Eq. (23) as long as the level sets, used for the representation of the curves' positions in the image plane, do not merge or split. Consequently, curves sliding past each other cannot be represented in this setting. If this is desired, a full level set approach needs to be employed. Then a completely implicit representation of the curve based on Eq. (23) is evolved, allowing for full topological flexibility. Of specific interest is the requirement that two curves will only be merged at a point if their positions *and* velocities at this point are identical, i.e., the methodology allows for curves to slide past each other if this is what their dynamical description requires them to do. When two curves slide past each other, the projection onto the image plane will show two intersecting closed curves. The merging behavior will be very different from the one observed for the partial level set approach. While two objects that get merged with the partial level set approach will lose their joint boundary, this will not be the case for the full level set approach where the two contours will get fused at their joint boundary.

We will treat the combined spatial and velocity evolution of closed curves within the vector distance function setting (as a full level set approach), thus providing a sample problem for the vector distance function approach to level-set evolutions in higher co-dimension. Eq. (23) was derived based on curves evolving in the plane. However, the implicit evolution of closed curves by a vector distance function approach requires the computation of planar quantities (e.g., the mean curvature vector in the image plane) based on the vector distance function values \mathbf{u} , which implicitly describe a curve in \mathbb{R}^4 . In short, Equation (23) needs to be written in terms of \mathbf{u} .

With

$$\mathbf{v} = \begin{pmatrix} v_0 \\ v_1 \end{pmatrix} = \beta \mathcal{N},$$

an evolution equation analogous to Equation (12) is

$$\begin{aligned} \mathcal{C}_t &= \mathbf{v}, \\ \mathbf{v}_t &= \beta_t \mathcal{N} - \beta \beta_s \mathcal{T} = \beta_t \mathcal{N} - \frac{1}{2} (\beta^2)_s \mathcal{T}, \end{aligned}$$

which is (upon substitution of Eq. (23))

$$\begin{aligned}\mathcal{C}_t &= \mathbf{v}, \\ \mathbf{v}_t &= \left(\frac{1}{2}\|\mathbf{v}\|^2 + \frac{g}{\mu}\right) \kappa \mathcal{N} - \frac{1}{\mu} (\nabla g \cdot \mathcal{N}) \mathcal{N} - \frac{1}{2} (\|\mathbf{v}\|^2)_s \mathcal{T}. \quad (24)\end{aligned}$$

To implement Eq. (24) in a level set framework the tangential, \mathcal{T} , and the normal, \mathcal{N} , vectors need to be written in terms of \mathbf{u} . Also, a replacement for the mean curvature vector in the plane $\mathcal{H}_p = \kappa \mathcal{N}$ has to be found. The curve position \mathcal{C} and the curve velocity \mathbf{v} are given implicitly by

$$\begin{aligned}\mathcal{C}(\mathbf{x}) &= \mathcal{P}(\mathbf{x}), \\ \mathbf{v}(\mathbf{x}) &= \mathcal{P}_v(\mathbf{x}), \quad (25)\end{aligned}$$

where

$$\begin{aligned}\mathcal{P}\left(\begin{pmatrix} x_1 & x_2 & x_3 & x_4^T \end{pmatrix}\right) &= \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \\ \mathcal{P}_v\left(\begin{pmatrix} x_1 & x_2 & x_3 & x_4^T \end{pmatrix}\right) &= \begin{pmatrix} x_3 \\ x_4 \end{pmatrix},\end{aligned}$$

and $\mathbf{x} \in \mathcal{M} = \mathbf{u}^{-1}(0)$.

Since the object dimension is one, exactly one eigenvalue of $D\mathbf{u}$ will be zero, while the others will be one (Gomes and Faugeras, 2001a). The eigenvector corresponding to the zero eigenvalue will be aligned with the tangential direction of the curve. For numerical robustness we propose to compute the full-dimensional tangential vector as

$$\mathcal{T}_f = \frac{\mathbf{t}}{\|\mathbf{t}\|},$$

where

$$\begin{aligned}\mathbf{t} &= \underset{\forall \mathbf{q}_i \in \mathcal{Q}}{\operatorname{argmin}} \|\lambda_i\|(\mathbf{q}_i), \\ \mathcal{Q} &= \left\{ \mathbf{q} \in \mathbb{R}^4 : \frac{1}{2} (D\mathbf{u} + (D\mathbf{u})^T) \mathbf{q} = \lambda \mathbf{q}, \lambda \in \mathbb{R} \right\}.\end{aligned}$$

The tangential vector \mathcal{T} in the plane is then the normalized projection of \mathcal{T}_f onto the image plane:

$$\mathcal{T} = \frac{\mathcal{P}(\mathcal{T}_f)}{\|\mathcal{P}(\mathcal{T}_f)\|}.$$

We define

$$\mathbf{o} := \mathcal{T}^\perp = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \mathcal{T}, \quad \mathcal{T}^u = \begin{pmatrix} \mathcal{T} \\ 0 \\ 0 \end{pmatrix}, \quad \mathcal{T}^u \in \mathbb{R}^4,$$

and propose to compute the mean curvature vector in the plane as

$$\mathcal{H}_p = \mathcal{T}_s = \left(\frac{1}{\|\mathcal{T} \cdot \mathcal{P}(\mathcal{T}_f)\|} \mathcal{P}((D\mathcal{T}^u) \mathcal{T}_f) \cdot \mathbf{o} \right) \mathbf{o}$$

and the squared velocity variation along the curve as

$$(\beta^2)_s = \frac{1}{\|\mathcal{T} \cdot \mathcal{P}(\mathcal{T}_f)\|} \nabla (\|\mathbf{v} \cdot \mathbf{o}\|^2) \mathcal{T}_f,$$

where we replaced quantities that should be normal to the planar curve by its projections onto \mathbf{o} to increase numerical robustness. Defining the reprojected velocity to be

$$\mathbf{v}_R := (\mathbf{v} \cdot \mathbf{o}) \mathbf{o},$$

we obtain (using the reprojected velocity) the overall evolution equation

$$\begin{aligned} \mathcal{C}_t &= \mathbf{v}_R, \\ \mathbf{v}_t &= \left(\frac{1}{2} \|\mathbf{v}_R\|^2 + \frac{1}{\mu} g \right) \mathcal{H}_p - \frac{1}{\mu} (\nabla g \cdot \mathbf{o}) \mathbf{o} - \frac{1}{2\|\mathcal{T} \cdot \mathcal{P}(\mathcal{T}_f)\|} \left(\nabla (\|\mathbf{v}_R\|^2) \mathcal{T}_f \right) \mathcal{T}. \end{aligned} \quad (27)$$

In principle \mathbf{v} should always be perpendicular to the evolving curve. In case of a numerical implementation this will not be the case due to numerical inaccuracies. It might thus be useful to enforce this property dynamically by changing Eq. (27) to

$$\begin{aligned} \mathcal{C}_t &= \mathbf{v}_R, \\ \mathbf{v}_t &= \left(\frac{1}{2} \|\mathbf{v}_R\|^2 + \frac{1}{\mu} g \right) \mathcal{H}_p - \frac{1}{\mu} (\nabla g \cdot \mathbf{o}) \mathbf{o} - \frac{1}{2\|\mathcal{T} \cdot \mathcal{P}(\mathcal{T}_f)\|} \left(\nabla (\|\mathbf{v}_R\|^2) \mathcal{T}_f \right) \mathcal{T} - K (\mathbf{v} - \mathbf{v}_R). \end{aligned}$$

The newly introduced term will ensure that the tangential velocity components will vanish for $t \rightarrow \infty$ (based on the correction gain $K > 0$).

5. Examples

In this section, we present some numerical test examples to illustrate the behavior of the algorithms. We show (i) a test example demonstrating the superiority of the vector distance function approach over the distance function approach, (ii) a mean curvature evolution, (iii) a translational evolution, and (iv) evolutions of one and two circular objects in a potential well.

Figure (10) shows a one dimensional toy problem. It demonstrates the superiority of the vector distance function approach as opposed to the distance function approach. Error compensation has a very beneficial effect on the distance function results, but cannot prevent the distance function from drifting away from its correct values (this is more dramatic for the case without error compensation). On the other hand, error compensation does not significantly improve the result for the vector distance function evolution in this example. In both cases (with or without error compensation) the vector distance function scheme accurately follows the desired movement.

Note that the main difference between the two approaches, and the reason for the dramatically different results in this example, lies in numerical problems for the distance function case: here the curve has a kink at the location of interest. The kink leads to numerical dissipation, smearing out the kink and resulting in smaller gradients (which cause very high sensitivity of the minimum of the level set function to numerical errors). On the other hand the vector distance function does not have a kink at the point of interest. In fact, the function is continuous and differentiable there. This is favorable for obtaining a reliable numerical solution.

Figure 11 shows a two-dimensional result for a vector distance function based curvature flow. This is merely a result to demonstrate the feasibility of vector distance function evolutions, since this problem could be solved by a standard level set approach using a signed distance function. The results show a nice regularization and the expected behavior (the object becomes increasingly circular over time). Finally, Figs. 12 and 13 show the result of propagating a circular object by applying a constant velocity field. By using error compensation (this is a prime example of the usefulness of this approach in certain cases) the theoretical result is almost indistinguishable from the numerical result.

Figures 14–15 show a circular object oscillating in a potential well based on the evolution equations derived in Section 4¹⁰. The curve shown is the projection of the space curve on the image plane with velocity vectors given by the space curve. Clearly the circular object oscillates. It dissipates energy due to the numerics and the reinitialization scheme, and stops oscillating in finite time.

To test the behavior for topological changes, Figs. 16–17 show two circles oscillating simultaneously in the same potential well used previously. They slide over each other while their velocities are significantly different from each other, but merge once their velocities become numerically indistinguishable. Note that this is a demanding example since we chose to discretize the velocity with the same accuracy as the spatial dimension (see the pixel size in the images). As illustration, Fig. (18) shows the three-dimensional projections for a representative thinned discrete approximation to the zero level set.

6. Conclusions

We have argued that the vector distance function based level set approach is a useful tool for practical evolutions of objects of codimension greater than one. To address implementation issues, analysis was restricted to the class of objects representable by the class of closed curves. The restriction greatly simplified the explicit reconstruction of the zero level set required for reinitialization purposes.

Increasing space dimensions leads to an increasing number of neighbors for each hypercube (in the computational narrow-banded grid) and thus to an increase in computational cost per iteration. Significant reductions in computational complexity are possible through narrow-banding schemes, which is of great importance for evolutions in high-dimensional spaces.

This paper thus presents the combination of an implicit scheme (the vector distance function based level set propagation) with occasional explicit reinitializations of the vector distance function. The explicit reinitializations are based on discrete geometry and the theory of computational homology. While a completely implicit scheme would be desirable, also with regard to the representation and evolution of a less restrictive object class, the reconstruction

¹⁰The potential function is of the form $g = \frac{1}{1 + (\nabla(I * G))^2}$, where I is the intensity image (here a circular blob) and G is a Gaussian filter. The results are for $\mu = K = 1$.

of an explicit representation of the zero level set has the advantage of remedying the fattening problem. Topological mergers can then be addressed satisfactorily.

A better mathematical understanding of the vector distance function evolution scheme is highly desirable. Numerical methods tailored specifically to the evolution of these discontinuous vector fields are useful and should be investigated. To obtain a completely implicit scheme, implicit reinitialization strategies need to be devised. This is a challenging problem, numerically and in the context of how to properly handle fattening artifacts.

Overall, the results obtained for the proposed methodology are encouraging. Initialization poses no problem in this setting, narrow-banding is feasible, and topological changes are handled satisfactorily.

Acknowledgments

This work was supported in part by grants from AFOSR, ARO, NSF, NIH (through NAC Brigham and Women's Hospital), MRI-HEL, and MURI. We would also like to thank Sigurd Angenent for some very helpful comments.

References

- Adalsteinsson D, Sethian JA. A fast level set method for propagating interfaces. *Journal of Computational Physics*. 1995; 118:269–277.
- Ambrosio L, Soner M. Level set approach to mean curvature flow in arbitrary codimension. *Journal of Differential Geometry*. 1996; 43:693–737.
- Bellettini G, Novaga M, Paolini M. An example of three dimensional fattening for linked space curves evolving by curvature. *Communications in Partial Differential Equations*. 1998; 23:1475–1492.
- Bertalmio M, Sapiro G, Randall G. Region tracking on level-sets methods. *IEEE Transactions on Medical Imaging*. 1999; 18(5):448–451. [PubMed: 10416806]
- Bertrand G. A Boolean characterization of three-dimensional simple points. *Pattern Recognition Letters*. 1996; 17:115–124.
- Burchard P, Cheng L, Merriman B, Osher S. Motion of curves in three spatial dimensions using a level set approach. *Journal of Computational Physics*. 2001; 170:720–741.
- Dierckx, P. *Curve and Surface Fitting with Splines*, Monographs on Numerical Analysis. Oxford Science Publications; 1993.
- do Carmo, MP. *Differential Geometry of Curves and Surfaces*. Prentice Hall; 1976.
- Dupont TF, Liu Y. Back and forth error compensation and correction methods for removing errors induced by uneven gradients of the level set function. *Journal of Computational Physics*. 2003; 190:311–324.
- Dupont, TF.; Liu, Y. Back and forth error compensation and correction methods for semi-Lagrangian schemes with applications to interface computation using level set method. Technical Report CDSNS2004-399. Georgia Institute of Technology; 2004.
- Evans LC, Spruck J. Motion of level sets by mean curvature I. *Journal of Differential Geometry*. 1991; 33:635–681.
- Evans LC, Spruck J. Motion of level sets by mean curvature II. *Transactions of the American Mathematical Society*. 1992a; 330:321–332.
- Evans LC, Spruck J. Motion of level sets by mean curvature III. *Journal of Geometric Analysis*. 1992b; (2):121–150.
- Evans LC, Spruck J. Motion of level sets by mean curvature IV. *Journal of Geometric Analysis*. 1995; (5):77–114.
- Gau CJ, Kong TY. 4D Minimal non-simple sets. *Lecture Notes in Computer Science*, Vol. 2301 of *Lecture Notes in Computer Science*. 2002:81–91.
- Gomes, J.; Faugeras, O. INRIA; 2000a. Representing and evolving smooth manifolds of arbitrary dimension embedded in R^n as the intersection of n hypersurfaces: The Vector Distance Functions. Technical Report 4012.

- Gomes, J.; Faugeras, O. Shape representation as the intersection of $n - k$ hypersurfaces. Technical Report 4011. INRIA; 2000b.
- Gomes J, Faugeras O, Kerckhove M. Using the vector distance functions to evolve manifolds of arbitrary codimension. *Scale-Space and Morphology in Computer Vision*, Vol. 2106 of *Lecture Notes in Computer Science*. 2001:1–13.
- Grayson M. The heat equation shrinks embedded plane curves to round points. *Journal of Differential Geometry*. 1987; 26:285–314.
- Kaczynski, T.; Mischaikow, K.; Mrozek, M. Computing Homology. Technical report. Georgia Institute of Technology; 2001.
- Kaczynski, T.; Mischaikow, K.; Mrozek, M. *Computational Homology*, Vol. 157 of *Applied Mathematical Sciences*. Springer Verlag; 2004.
- Kalies, WD.; Mischaikow, K.; Watson, G. Conley Index Theory. Banach Center Publications; 1999. Cubical approximation and computation of homology.; p. 115-131.
- Klette G. Simple points in 2D and 3D binary images. *Lecture Notes in Computer Science*, Vol. 2756 of *Lecture Notes in Computer Science*. 2003:57–64.
- Kong, TY. *Discrete Geometry for Computer Imagery*, Vol. 1347 of *Lecture Notes in Computer Science*. Springer; 1997. Topology-preserving deletion of 1's from 2-, 3- and 4-dimensional binary images.; p. 3-18.
- Leveque, RJ. Finite Volume Methods for Hyperbolic Problems. Cambridge Texts in Applied Mathematics; 2002.
- Lorigo LM, Faugeras O, Grimson WEL, Keriven R, Kikinis R, Westin C-F. Co-dimension 2 geodesic active contours for MRA segmentation. *Proceedings of the International Conference on Information Processing in Medical Imaging*. 1999:126–139.
- Massey, WS. A Basic Course in Algebraic Topology, Graduate Texts in Mathematics. Springer-Verlag; 1991.
- Min C. Local level set method in high dimension and codimension. *Journal of computational physics*. 2004; 200:368–382.
- Niethammer, M.; Kalies, WD.; Mischaikow, K.; Tannenbaum, A. Detecting simple points in higher dimensions.. 2004a. In preparation
- Niethammer M, Tannenbaum A, Angenent S. Dynamic geodesic snakes for visual tracking. Submitted to the *IEEE Transactions on Automatic Control*. 2004b
- Osher S, Cheng L, Kang M, Shim H, Tsai Y. Geometric optics in a Phase Space Based Level Set and Eulerian Framework. *Journal of Computational Physics*. 2002; 179(2):622–648.
- Osher, S.; Fedkiw, R. Level Set Methods and Dynamic Implicit Surfaces. Springer Verlag; 2003.
- Osher S, Fedkiw RP. Level set methods: An overview and some recent results. *Journal of Computational Physics*. 2001; 169:463–502.
- Pottman, H.; Hofer, M. Springer; 2003. *Visualization and mathematics III*, Chapt. Geometry of the squared distance function to curves and surfaces; p. 223-244.
- Sethian, JA. Level Set Methods and Fast Marching Methods. 2nd edition. Cambridge University Press; 1999.
- Slepcev D. On level-set approach to motion of manifolds of arbitrary codimension. *Interfaces and Free Boundaries*. 2003; 5:417–458.
- Sussman M, Smereka P, Osher S. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*. 1994; 114:146–159.
- Thürmer G. Closed curves in n -dimensional discrete space. *Graphical Models*. 2003; 65:43–60.

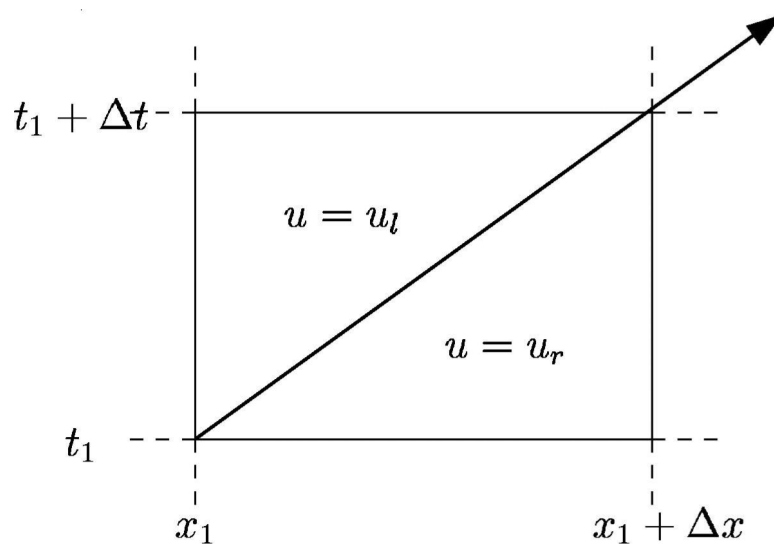
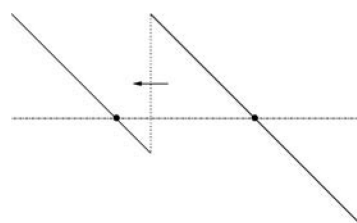
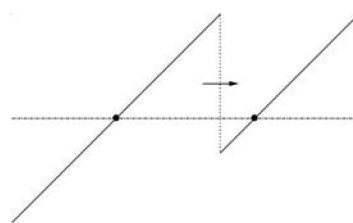


Figure 1.
Shock propagation over an infinitesimal time interval Δt .



a. Shock propagating to the left.



b. Shock propagating to the right.

Figure 2.
Shock propagation.

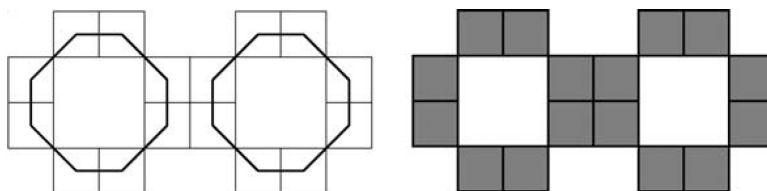


Figure 3.
Example of a thinning of the zero band that does not yield two clearly separable discrete closed simple curves.

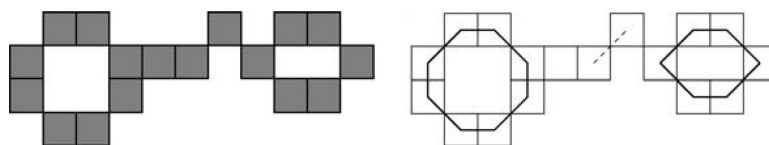


Figure 4.
Undesired topological connection remaining after topological thinning.

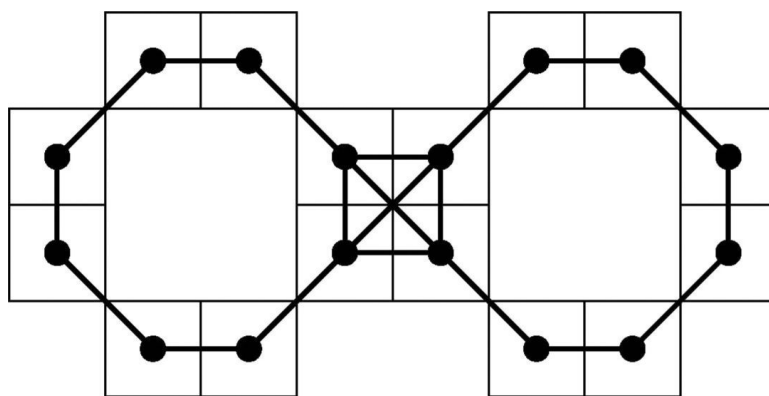


Figure 5.
All possible edges and vertices based on the discrete connectivity.

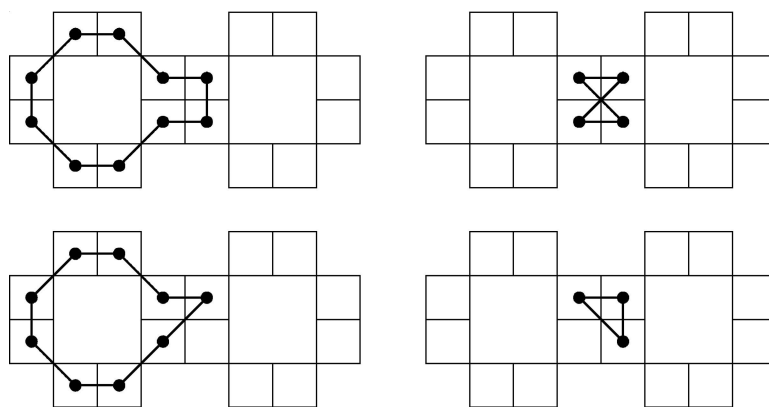


Figure 6.
Some possible simple cycles that are not valid representatives for a discrete closed curve.

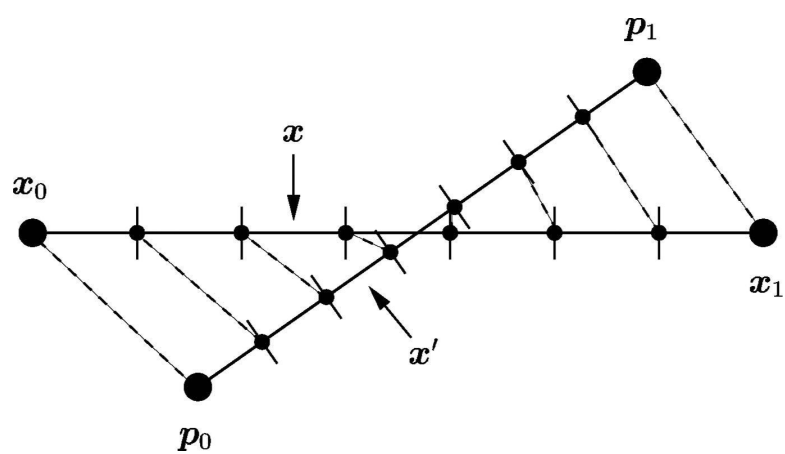
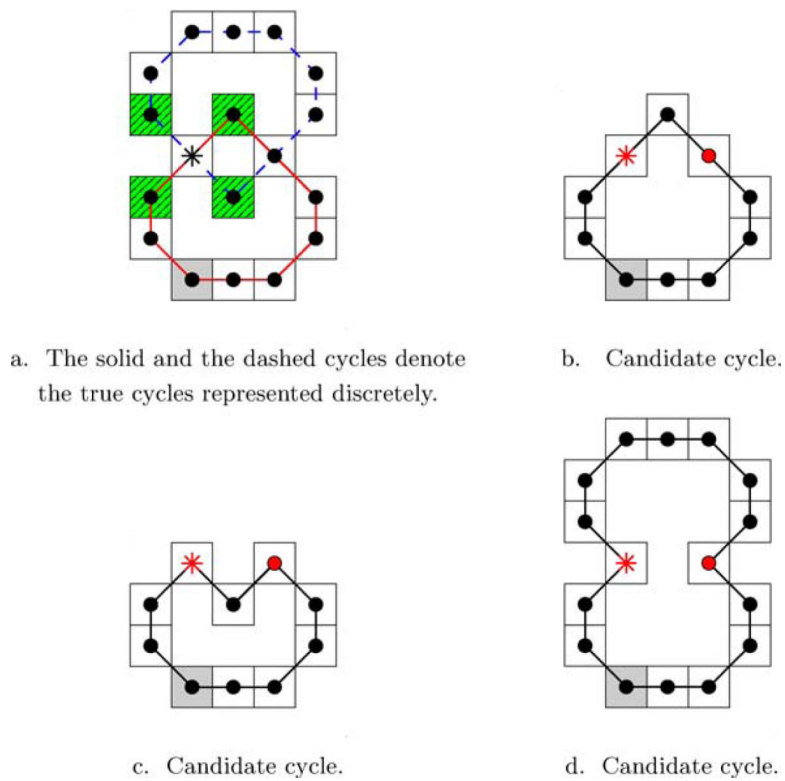
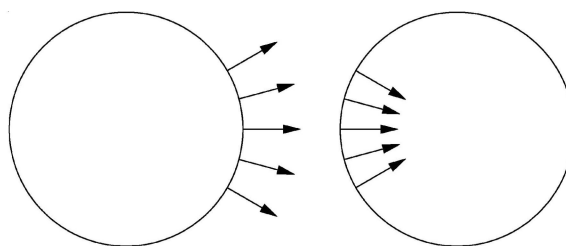


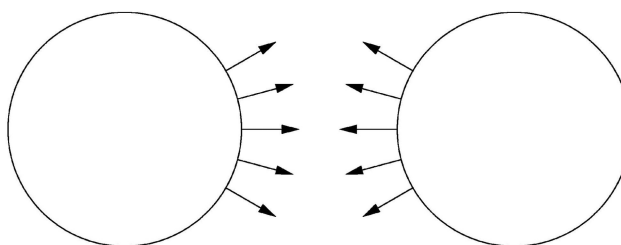
Figure 7.
Illustration of the line quality measure.

**Figure 8.**

Starting with an edge leaving the light gray block at the bottom left, three candidate cycles for the discrete configuration (a) are possible, with (b) being the smoothest one.

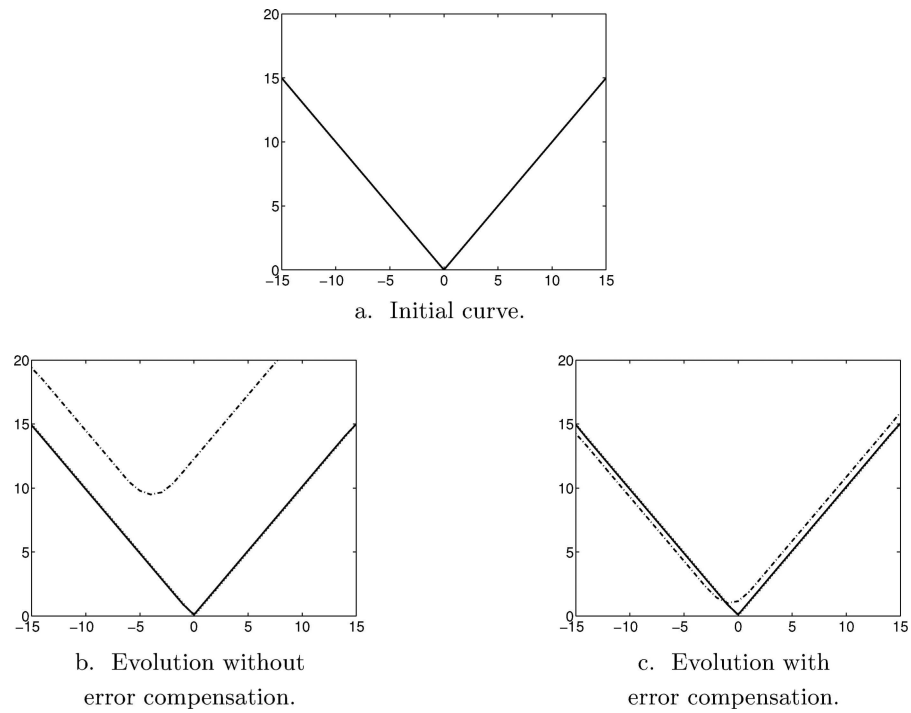


a. Curves should merge, but do not if only the scalar normal velocity β gets propagated.

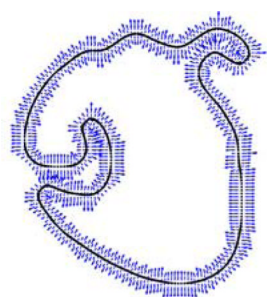


b. Curves should not merge, but merge if only the scalar normal velocity β gets propagated.

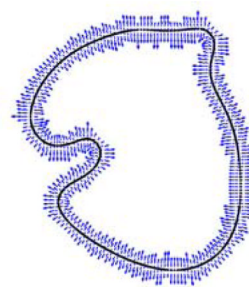
Figure 9.
Merging curves scenarios.

**Figure 10.**

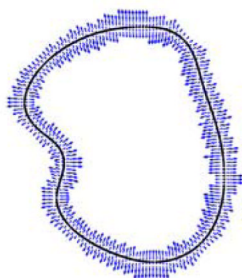
One-dimensional distance function and vector distance function evolutions. Dash-dotted curves are the results for the distance function evolution, dotted curves are the initial curves for comparison, and solid curves are the results for the vector distance function evolutions. The results represent evolutions for 400 timesteps with a timestep of 0.1. The evolutions were performed with a constant velocity field of 1 which flips sign every 100 iterations.



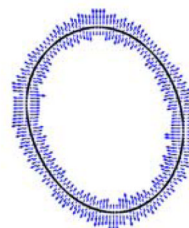
a. Vector distance function
based curvature flow.
Evolution step 50.



b. Vector distance function
based curvature flow.
Evolution step 500.



c. Vector distance function
based curvature flow.
Evolution step 1000.



d. Vector distance function
based curvature flow.
Evolution step 4000.

Figure 11.
Vector distance function based curvature flow. Arrows indicate the vector distance function u .

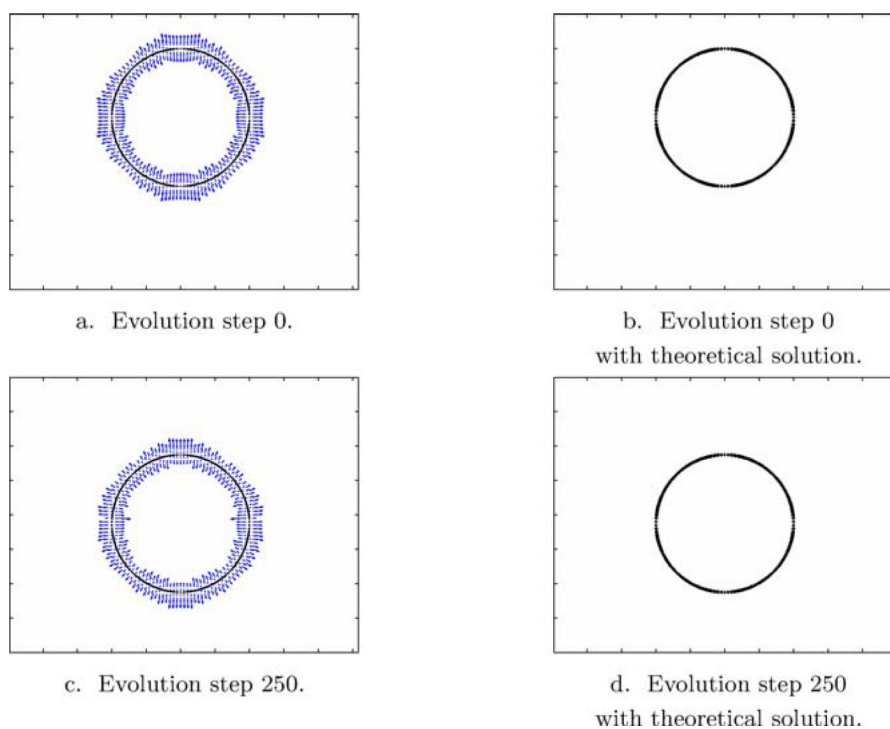


Figure 12. Circular object subject to a uniform flow field with theoretical solution. Evolution steps 0 and 250.

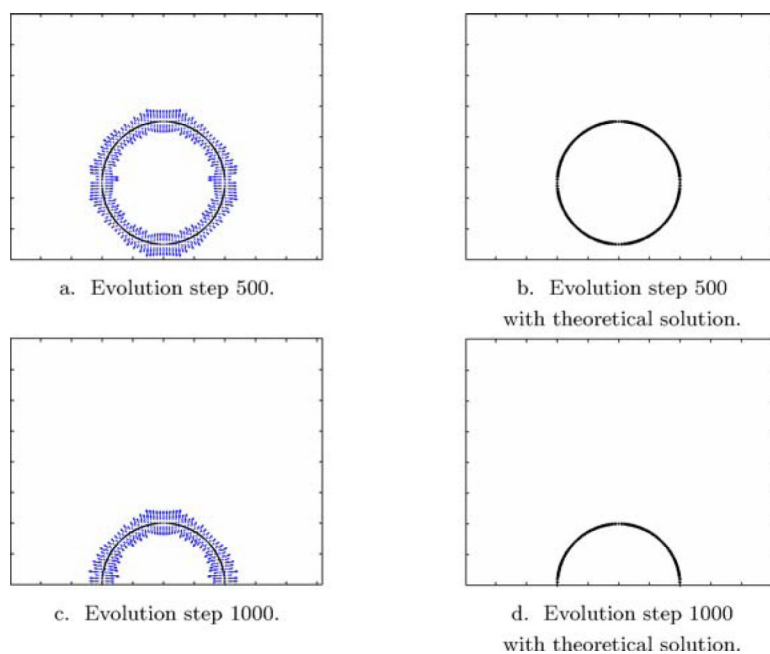


Figure 13. Circular object subject to a uniform flow field with theoretical solution. Evolution steps 500 and 1000.

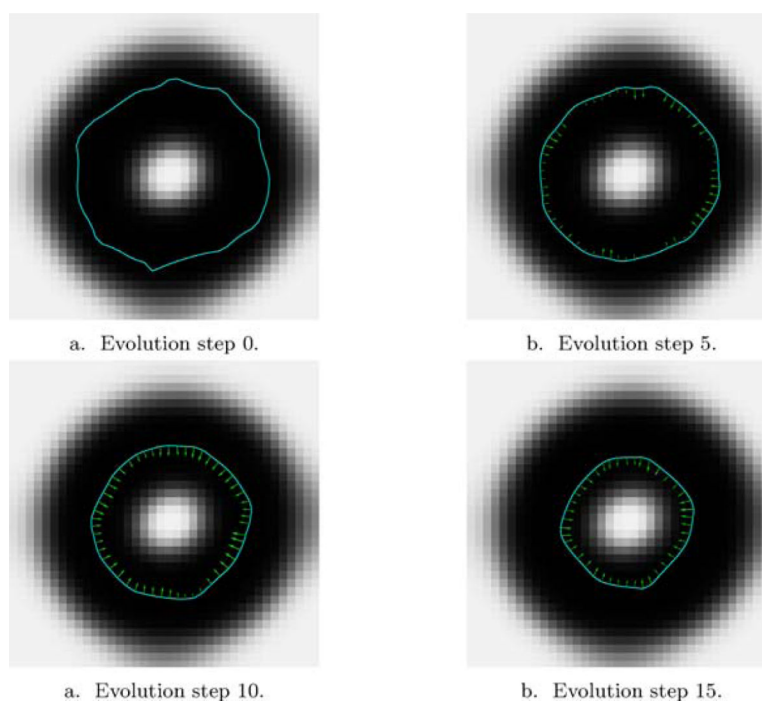


Figure 14.
Oscillating circular curve. Evolution steps 0-15.

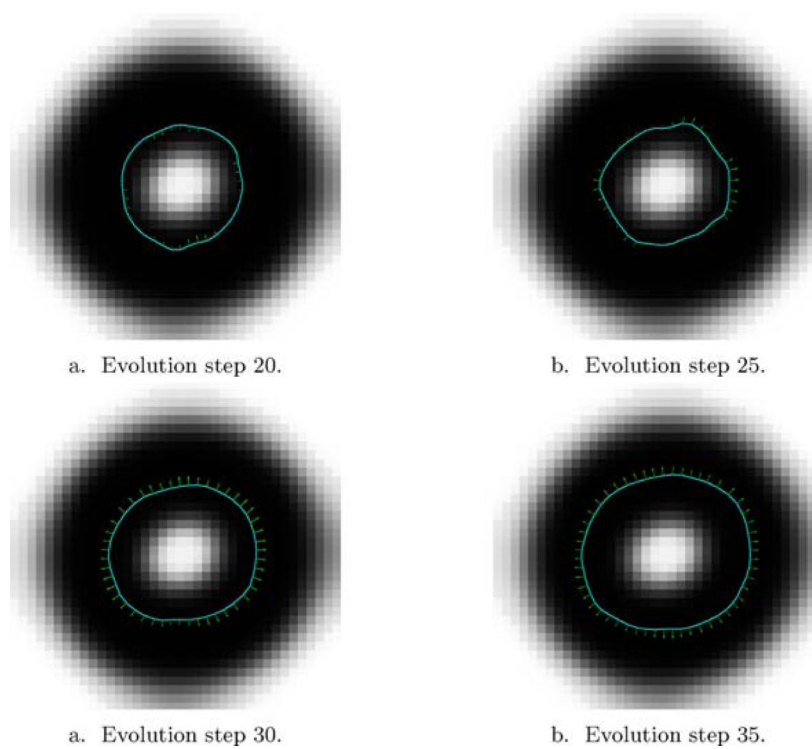


Figure 15.
Oscillating circular curve. Evolution steps 20–35.

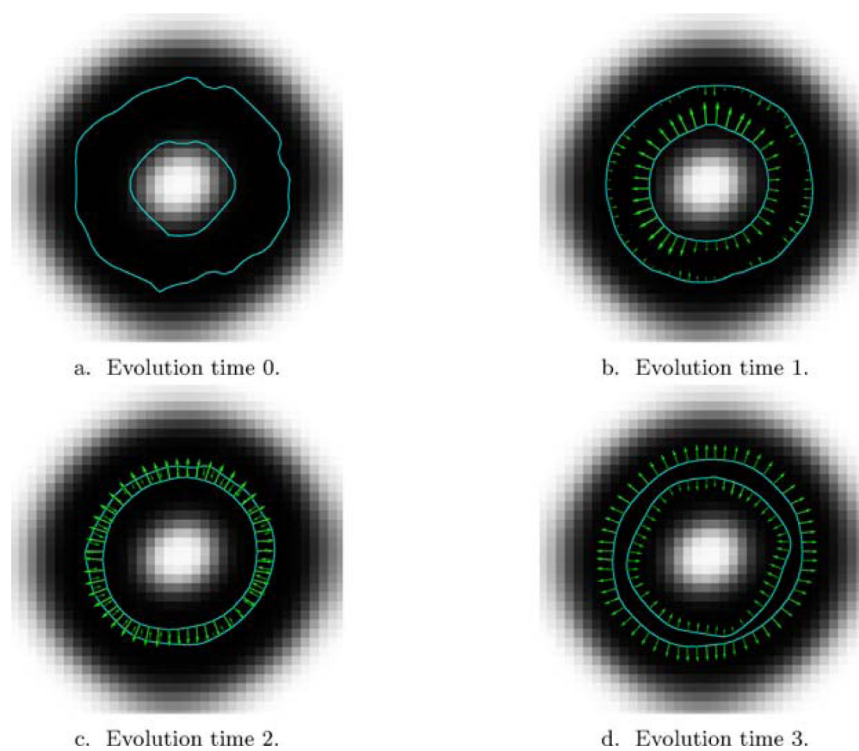


Figure 16.
Two oscillating circles. Time steps 0 to 3.

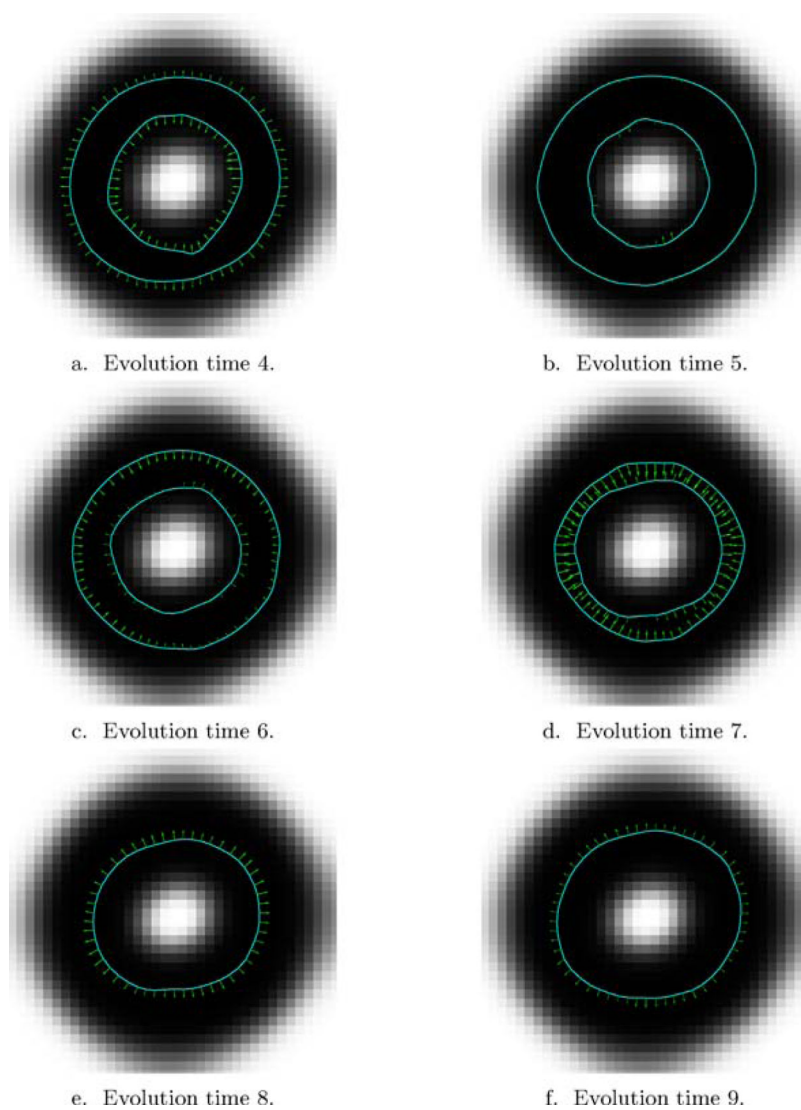


Figure 17.
Two oscillating circles. Time steps 4 to 9.

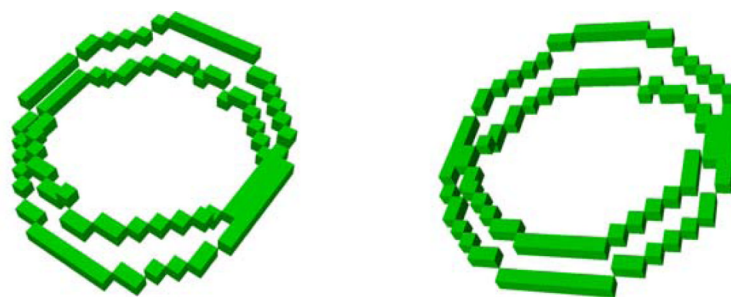


Figure 18.

Projections of the thinned discrete zero level set representation.

Table 1

List of symbols.

Symbol	Description
∇	gradient
D	Jacobian
∇^2	Hessian
Δ	Laplacian
\mathbf{b}	vector distance function velocity function
\mathbf{V}	velocity of the zero level set $\mathbf{b} = -\mathbf{V}$
\mathbf{v}	planar curve velocity
β	normal curve velocity
$\delta(x)$	(scalar) distance function
\mathbf{u}	vector distance function
Ω	computational domain
\mathbf{V}	vertex set of a graph
\mathbf{E}	edge set of a graph
$H_k(X)$	k -th homology group of X
β_k	Betti numbers
\mathbf{M}	manifold
\mathbf{H}	mean curvature vector
\mathbf{C}	planar closed curve
\mathbf{N}	unit inward normal to \mathbf{C}
\mathbf{T}	unit tangent vector to \mathbf{C}
κ	signed curvature to \mathbf{C}
s	arclength
p	curve parameterization
g	potential function