Dynamic Level Sets for Visual Tracking

A Thesis
Presented to
The Academic Faculty

by

Marc Niethammer

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Electrical and Computer Engineering
Georgia Institute of Technology
November 4, 2004

Dynamic Level Sets for Visual Tracking

Approved by:

Dr. Allen Tannenbaum, Advisor

Dr. Magnus Egerstedt

Dr. Laurence J. Jacobs

Dr. Waymond Scott

Dr. Anthony Yezzi

Date Approved: November 4, 2004

To my parents

ACKNOWLEDGEMENTS

This thesis would not have been possible without the help and support of many extraordinary people. I am indebted to all of them.

I wish to thank my advisor Allen Tannenbaum for his support, guidance, and his joyfulness, making it a pleasure to work with him and in his lab.

I feel very indebted to Larry Jacobs, for his genuine support, interest and friendship, especially throughout some overcast days.

My thanks go out to my friends and colleagues Eric Pichon, Amanda Wake, Patricio Vela, Delphine Nain, Eli Hershkovits, Chris Alvino, Gallagher Pryor, Lei Zhu, Yan Yang, Yogesh Rathi, and Oleg Michailovich without whom writing this thesis would not have been possible.

Many thanks also go to Konstantin Mischaikow, Bill Kalies and Sigurd Angenent for very helpful discussions, support and for being a pleasure to work with.

I am thankful for the support of my committee members Magnus Egerstedt, Anthony Yezzi, and Waymond Scott.

Special thanks go to Irene, for continuous support, love, and happiness.

Last but not least, I wish to thank my parents, to whom this thesis is dedicated, for years of unconditional love and support, and my sister who all have contributed in their own ways to the successful completion of this thesis. This would not have been possible without you – thanks.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS OR ABBREVIATIONS

$\|A\|$	the matrix norm.
A	a matrix.
A^{-1}	the inverse operator.
$\boldsymbol{x}$	a vector.
$\|\boldsymbol{x}\|$	the vector norm.
$\mathcal{C}$	a curve.
$\|\cdot\|_1$	the 1-norm.
$\|\cdot\|_2$	the 2-norm.
$<\cdot,\cdot>$	the inner product.
$\|\cdot\|_\infty$	the infinity-norm.
$\mathcal{N}$	the unit inward normal.
$\mathcal{T}$	the tangent vector.
δ	the delta function.
Δ	the Laplace operator.
δ_{ik}	the Kronecker delta symbol.
h	the Heaviside function.
$\Im z$	the imaginary part of z.
$\mathbb{C}$	the set of complex numbers.
$\mathbb{C}^n$	the vector space of n-tuples of complex numbers.
$\mathbb{N}$	the set of positive integers.
$\mathbb{R}$	the set of real numbers.
$\mathbb{R}^n$	the vector space of n-tuples of real numbers.
$\mathbb{Z}$	the set of integers.
$\mathcal{C}^\infty(\Omega)$	The space of infinitely differentiable functions on Ω.
$\mathcal{C}^k(\Omega)$	The space of k-times continuously differentiable functions on Ω.
$\mathcal{C}(\Omega)$	The space of continuous functions on Ω.
∇	the gradient operator.
$\Re z$	the real part of z.
$x \perp y$	x orthogonal to y.

SUMMARY

Visual tracking is the problem of following the positions of possibly multiple objects in a video sequence, based on the input of one or many cameras (the optical sensors). It is a fundamental problem of artificial vision and many different solution approaches have been proposed. This thesis deals with visual tracking based on a sequence of planar images in a partial differential equation framework.

Image segmentation can be seen as a static equivalent of tracking. A variety of active contour approaches have been successfully applied to this problem, where a curve deforms to minimize a given energy. Even though this deformation is typically written as an evolution over time, there is usually no relation to physical time. Geodesic active contours (and their higher dimensional analogs) have been particularly popular due to their geometric nature which naturally leads to a level set implementation. However, geodesic active contours have so far been inherently static.

This thesis introduces *geometric dynamic active contours* in the context of visual tracking, augmenting geometric curve evolution with physically motivated dynamics. Adding additional state information to an evolving curve lifts the curve evolution problem to space dimensions larger than two and thus does not allow for the use of classical level set techniques: these only apply to closed curves or surfaces of codimension one.

This thesis therefore develops and explores level set methods for problems of higher codimensions, putting an emphasis on the vector distance function based approach. This formalism is very general, it is interesting in its own right and still a challenging topic.

Two different implementations for geometric dynamic active contours are explored: the *full level set approach* as well as a simpler *partial level set approach*. The full level set approach results in full topological flexibility and can deal with curve intersections in the image plane. However, it is computationally expensive. On the other hand the partial

level set approach gives up the topological flexibility (intersecting curves cannot be represented) for increased computational efficiency. Contours colliding with different dynamic information (e.g., objects crossing in the image plane) will be merged in the partial level set approach whereas they will correctly traverse each other in the full level set approach. Both implementations are illustrated on synthetic and real examples.

Compared to the traditional static curve evolution case, fundamentally different evolution behaviors can be obtained by propagating additional information along with every point on a curve.

CHAPTER 1

INTRODUCTION

Object tracking can be accomplished in many ways including by mechanical, acoustical, magnetic, inertial, or optical sensing, and by radio and microwaves, to mention a few. The ideal tracker should be "tiny, self-contained, complete, accurate, fast, immune to occlusions, robust, tenacious, wireless, and cheap" [139, 66]. As of now such a tracker does not exist; trade-offs are necessary. A method should be chosen based on the application at hand. Optical sensing is unobtrusive and can be simplified by choosing a simple (possibly prespecified) work environment, or by altering the appearance of the objects to be tracked (e.g., by painting them, or by mounting light sources on them). The desired objects to be tracked then become much easier to detect. However, in certain instances (e.g., for an uncooperative object to be followed) this is not possible. Visual tracking is the task of following the positions of possibly multiple objects based on the inputs of one or many cameras (the optical sensors). It is a challenging image segmentation problem with added temporal information. From a controls perspective we distinguish between open loop and closed loop control. Since visual tracking is based on measurements (an image, or image sequence) obtained by an image sensor (a camera), it is intimately linked with closed loop control. In the context of visual tracking, the two tasks of locating and following an object (e.g., for surveillance applications), and influencing objects or the environment (e.g., controlling the movement of a plane, based on visual input) can be distinguished. The latter will most likely encompass the first (possibly resulting in nested control loops). Both tasks can be accomplished by means of feedback mechanisms. Either case needs a good estimate of object position. Having obtained this estimation, the task is either fulfilled (e.g., for surveillance applications), or this information facilitates closing a control loop. This brings to mind a broad range of applications. Indeed, be it for medical or military use, the need for visual tracking is ubiquitous.

Humans and animals perform visual tracking tasks with ease every day: following cars in traffic, watching other people, following the lines of text in a document, etc. These mundane tasks seem simple, but designing robust reliable algorithms and their computer implementation have proven to be quite challenging [4]. We rely on a highly developed brain, assumptions about the world acquired throughout a lifetime, and highly effective visual sensors: our eyes. The design of algorithms which would make a machine behave and perceive similarly to humans in all situations is a daunting task which is far from being solved. However, if only a specific application is of interest, the problem becomes more tractable. Visual tracking is a relatively well defined problem when dealing with well defined environments.

Applications for visual tracking are diverse. Some key areas of research include:

- Vehicle guidance and control: See [134, 90, 14, 48, 36, 93] for applications to autonomous driving [1]. See Sinopoli *et al.* [119] and Sharp *et al.* [117] for visual tracking systems for the navigation and the landing of an unmanned aerial vehicle, respectively.

- Surveillance and identification: See [110, 127, 15] for applications to target tracking and biometric identification.

- Robotics and manufacturing: See Corke [28] and Hutchinson *et al.* [62] for discussions on visual servo control which requires the visual tracking of objects/object features as a preprocessing stage. Here visual tracking is used to increase the bandwidth and accuracy of robots. Visual grasping falls into this category of tasks.

- User interfaces: See [102] for real-time fingertip tracking and gesture recognition, and [4] for virtual environments.

- Video processing: See [71] for automated addition of virtual objects to a movie.

- Medical applications: See [61] for applications to vision guided surgery (surgical instrument tracking) and [8] for medical image tracking.

[1] Exemplary for these research efforts are the European Prometheus and the American PATH programs.

A wide variety of algorithms for visual tracking exists: e.g., feature trackers, blob trackers, contour and surface trackers. See [19, 17, 80, 30, 97, 126] and the references therein. All of these have their own advantages and disadvantages. The seminal paper of Kass *et al.* [74] spawned a huge interest in the area of contour and surface tracking algorithms; these are the kind of algorithms this thesis will focus on.

1.1 Methodologies for Visual Tracking

The following sections group visual tracking approaches. The boundaries from one group to another are fuzzy. The grouping is similar to the one by McLauchlan and Malik [93]. It focuses on contour and surface trackers.

1.1.1 Feature Trackers

Feature trackers aim at finding and following features from image frame to image frame. The most prominent representative of this class of trackers is probably the Kanade-Lucas-Tomasi (KLT) tracker. Various extensions of this algorithm exist [118, 51]. The two most crucial steps for feature trackers are

- selecting features and
- assessing a feature's quality.

Only reliable (high quality) features should be used for the tracking. In the case of the Shi-Tomasi-Kanade tracker [118] feature selection is for example accomplished based on the conditioning of the linear system relating point translation to image measurements (image derivatives). A well conditioned linear system corresponds to a good feature that can be tracked well. To measure the quality of an image feature over time a measure of feature dissimilarity is introduced. A feature is only being tracked if its dissimilarity measure is small enough.

1.1.2 Blob Trackers

Blob trackers do not try to identify individual features (e.g., corners, bright spots, etc.) or to track an outline of an object. Instead, the tracking is performed by separating the object

and the image background. In its most simple form this could be accomplished by intensity based thresholding or background subtraction; however, many other criteria to distinguish the object from its background are conceivable: e.g., statistical properties. An advantage of most blob trackers is that because of their simplicity they allow for real-time tracking. Keeping track and distinguishing multiple blobs can be accomplished by assigning states (properties) to individual blobs: e.g., color, speed, traveling direction (see for example the KidsRoom project [63]).

1.1.3 Contour and Surface Trackers

Active contours or snakes were introduced by Kass *et al.* [74] as energy minimizing splines in the two-dimensional image plane. The energy is classically based on elasticity and rigidity constraints on the contour and the underlying image so that the snake gets attracted to the sought for image features. More recently, area based approaches have been developed (see for example [106]). Two different approaches for active contour based visual tracking exist: the static and the dynamic approaches. In the static approach the contour is oblivious of its own state; no velocity information is propagated. Tracking can then be achieved by solving subsequent static problems [74] (assuming that the movements of the object to be tracked are relatively slow and the contour does not leave the object's capture range from one frame to the next), or by incorporating temporal information (e.g., optical flow) into an energy functional to be minimized [106, 135]. The dynamic approach [129, 107] is based on a dynamical systems perspective, where points on the contour possess an inherent kinetic energy. They are typically associated with a mass, a velocity vector, are related to their neighboring points through elasticity and rigidity constraints, and move based on an underlying potential field. One of the main problems in visual tracking is dealing with image clutter against which visual tracking algorithms should be robust. The simplest way to increase robustness of contour and surface based trackers is to incorporate shape or movement information into the tracking algorithm [16, 18, 24, 23, 27, 26, 34, 32, 64, 82, 99, 112, 124, 133, 138, 146]; this is sensible, since in many cases the appearance of the object

to be tracked is known (e.g., the approximate shape of lips for lip tracking [50], the shape of an eye [146]).

1.2 Scope and Relation to Previous Work

This thesis is concerned with visual object tracking within the class of contour trackers. Its contribution can be separated into two major parts. First, this thesis is concerned with adding dynamics to contour based visual tracking in a geometric framework. Second, it investigates level set methods in the context of geometric dynamic contour evolutions.

Typical geometric active contours [21, 22, 75, 116, 91] are static. However, variational formulations many times falsely appear to be dynamic, because the resulting Euler-Lagrange equations are solved by gradient descent, introducing an *artificial* time parameter. This time parameter simply describes the evolution of the gradient descent. It will usually not be related to physical time; the formalism is inherently static. A two step approach is typically used for visual tracking by static active contours. First, the curve evolves on a static frame until convergence (or for a fixed number of evolution steps). Second, the location of the curve in the next frame is predicted. In the simplest case this prediction is the current location. Better prediction results can be achieved by using optical flow information, for example. In this two step approach, the curve is not moving intrinsically, but instead is placed in the solution's vicinity by an external observer (the prediction algorithm). The curve is completely unaware of its state. The classical level set method [115] is easy to use in this formalism. However, the approach is unnatural: The curve evolution gets decoupled from the actual dynamics of the objects to be tracked. Even though the standard curve evolution equations *seem* to be dynamic (they include a time dependence, e.g., $\mathcal{C}_t = \kappa\mathcal{N}$) they are not.

In contrast, Terzopoulos and Szeliski [129] or Peterfreund [107] view curve evolution from a dynamical systems perspective; both methods are marker particle based and are fast, but they may suffer from numerical problems (e.g., in the case of sharp corners [115]). In the static case, level set methods are known to handle sharp corners, topological changes, and to be numerically robust. In their standard form, they are restricted to codimension one

problems (of closed curves and surfaces) and are thus not suitable for dynamic curve evolution, since adding additional state information (e.g., velocity information) to an evolving curve lifts the curve evolution problem to higher space dimensions and therefore to higher codimensions. Extensions of level set methods to higher codimensions exist and a level set formulation for dynamic curve evolution is desirable [100, 101].

The main motivation for this thesis is to have a natural framework for visual tracking. The results of this thesis relate to dynamic snakes [129] as geodesic or conformal active contours [75, 22] relate to the original snake formulation [74], advocating a different philosophy to dynamic curve evolution. Instead of discretizing evolution equations upfront (early lumping), partial differential equations describe the system as long as possible (late lumping [141]), resulting in a more natural and geometric formulation.

Uniting dynamic active contours with the level set framework has several important implications:

- The formulation will no longer be *parametric*, but will become *geometric*.
- It will no longer be possible to only use the classical level set approach, which is restricted to codimension one problems of closed curves or surfaces.
- Numerical schemes will need to be tailored specifically for the resulting evolution equations.

Level set methods for codimension one problems of closed curves or surfaces are well developed [115]. However, in the context of this thesis, level set methods for problems of higher codimension are of importance. These are a relatively recent development and little is known theoretically for many of these methods.

Ambrosio and Soner [6] extend the codimension one level set approach to arbitrary codimensions. They propose to surround an evolving surface of higher codimension by a family of hypersurfaces (the level sets of a scalar function). Their setup facilitates mathematical analysis. Specifically, they prove existence and uniqueness of a weak solution. On the downside, the approach is not straightforward to use for practical purposes, since extraction of the evolving surface is nontrivial and the approach may suffer from fattening artifacts

in case of topological mergers, thus partially losing one of the main advantages of level set methods over particle based methods. Lorigo *et al.* [86] utilize this method to segment blood vessels in magnetic resonance angiography images. This is a natural application for the theory developed by Ambrosio and Soner, since objects are represented by hyper-tubes.

The alternative approach of representing a manifold of codimension k in $\mathbb{R}^d$ by the intersection of k scalar function in $\mathbb{R}^d$, proposed by Ambrosio and Soner, is not analyzed mathematically in their seminal paper [6], due to theoretical difficulties.

Nevertheless, it enjoys popularity in practice. Bertalmio *et al.* [11] perform region tracking on a two dimensional manifold in $\mathbb{R}^3$ by intersecting two hypersurfaces represented as level sets of two level set functions. Similarly, Osher *et al.* [103] model planar wavefronts of geometric optics, as objects in three-dimensional space (codimension two), and are thus able to deal with self intersections of the wavefronts. The main complication for these approaches is the initialization of the level set functions. Classically, these are signed distance functions, where the zero level set describes the hypersurfaces. However, an initialization based on signed distance functions is not unique (different sets of hypersurfaces can have the same intersection). Especially at a distance from the intersection, it is not clear how the level sets should be extended.

To cope with this problem, Gomes *et al.* [57] evolve vector distance functions to implicitly move manifolds of arbitrary dimension. This amounts to the intersection of n hypersurfaces in an n-dimensional space. The description is thus redundant, but does not suffer from initialization problems, since the description of a manifold in terms of a vector distance function is unique.

Evolutions based on vector distance functions or on the intersection of multiple level sets are computationally expensive, since the evolutions are performed in high dimensional spaces. Furthermore, it is not clear how to devise a narrow-banding technique (where the equations are only solved within a small band around the manifold to be evolved; see for example [115]) in the case of a representation based on intersecting hypersurfaces [56].

This thesis explores two different implementations for geometric dynamic active contour based tracking: the partial, and the full level set approach. The full level set approach is

based on a vector distance function evolution. It possesses full topological flexibility and can deal with curve intersections in the image plane, but is computationally expensive. Handling self intersecting objects is especially important for tracking moving objects which temporarily occlude each other. The partial level set approach only represents curves in the image plane by a level set approach. This allows for the use of well developed numerical algorithms, but gives up the topological flexibility for increased computational efficiency.

1.3 Organization of this Thesis

The remainder of the thesis is structured as follows: Chapter 2 presents background information useful for a deeper understanding of the material presented in subsequent chapters. Chapter 3 develops the dynamic curve evolution theory. Edge-based and area based approaches are considered. Chapter 4 deals with possible extensions to the framework developed in Chapter 3. Specifically, extensions are proposed that simplify the use of the developed dynamic curve evolution theory on real image sequences. Level set implementations are discussed in Chapter 5. Chapter 6 presents simulation results. Conclusions and possible research directions are given in Chapter 7. The thesis contains an appendix for detailed derivations of the curve evolution equations and an explanation of the numerical methods used.

CHAPTER 2

PRELIMINARIES

This chapter gives some background on the mathematical tools and descriptions used in the remainder of this thesis. Section 2.1 briefly discusses different ways to represent an object. Section 2.2 gives a general overview over filtering methods for a relatively general class of systems given in state space form. This facilitates the classification of the infinite dimensional state space description used for curve description in this thesis. As a side effect Section 2.2 can be regarded as a general introduction to the concept of observers on an informal level. Finally, Section 2.3 introduces the machinery of cubical homology which will turn out to be a very useful tool for the discrete geometry considered in Section 5.2.

2.1 Object Representations

Object representation is a crucial design step. Representational flexibility increases, whereas robustness usually decreases, with increased dimensionality of the representation. The simplest way to represent an object is a point (0-dimensional). Partial differential equation based approaches are infinite dimensional, they are considerably more complex. In between lie the finite dimensional (parameterized) representations.

Two possible ways to represent objects are to represent them by their boundary (contour-based) or by the region they cover. For region based approaches, texture information can be incorporated: see the literature on active appearance models [25, 123, 114].

The simplest way to represent the boundary of an object is to represent it by a finite number of points [124]. To obtain the boundary from the set of points an interpolation scheme is necessary. The easiest interpolation scheme being linear interpolation. Frequently splines are used for this purpose [18, 19, 136]. These methods can be regarded as the

simplest form of contour-based parametric models, where a parameterization describes an object boundary[1].

Parametric methods have the advantage that they are usually computationally relatively inexpensive and they allow easily for the incorporation of motion restrictions. The art lies in the choice of parameterization. In light of representing curves as dynamical systems, some parameterizations are better suited than others. In most cases linear or affine parameterizations (i.e., the parameters map in a linear or affine fashion to the coordinates of points on the curve) will be desirable for dynamic curve evolution. However, nonlinear parameterization may be useful to simplify a system's governing equations (in this new parameterized description). We refer the reader to the book by Sonka *et al.* [122] for a more thorough treatment of object representations.

When employing a certain object representation in the context of visual tracking, it is useful to restrict the shape search space whenever possible. This will lead to more robust tracking algorithms. If we are following the center of mass of an object for example, we could derive the dynamics based on a point model. In the plane this would amount to specifying the coordinates of the point. If we know we are looking for circles it would be sufficient to specify the center of the circle and a radius, for an ellipse it is sufficient to specify the minor and major axis including the center. The tracking problem simplifies further, if the object to be tracked only moves in certain ways, e.g., only by translation, by translation and rotation, by affine transformations.

This thesis deals with dynamic curve evolution equations described by partial differential equations. This leads to great representational flexibility. Adding shape constraints for such systems is important, but not straightforward. It is still subject to active research and beyond the scope of this thesis (see for example [82, 24, 23, 34, 32, 33, 112, 133]). The contribution of this thesis is in the novel dynamic description and implementation of curve evolution on which shape constraints should be added on the basis of further research.

[1]Region descriptions can be parametric as well. See for example the literature on active appearance models, where frequently a principal component analysis is employed to derive a parametric appearance model [25].

2.2 Filtering

The dynamical equations governing curve evolution will depend on the chosen object description. If the object description is finite (infinite) dimensional then the system equations will in general by finite (infinite) dimensional as well [2]. Since for computer implementations only the finite dimensional case matters, two different methodologies can be distinguished for the infinite dimensional case: early lumping and late lumping. Early lumping aims at discretizing a system description upfront, thus approximating it by a finite dimensional system. Then all the tools for finite dimensional systems can be applied to this finite dimensional approximation. The late lumping approach on the other hand aims at keeping the infinite dimensional description as long as possible, discretizing only as a very last step before the actual implementation. This then requires and can make use of the whole theoretical machinery for infinite dimensional systems [35].

A dynamical system can be discrete, continuous, deterministic, stochastic or a mixture of the above (as for hybrid systems, where the interplay of continuous and discrete system parts is studied). This thesis is concerned with infinite dimensional continuous time systems. To position this type of system within the class of possible system descriptions, this section gives an overview over some typical state space system descriptions. We restrict ourselves to purely discrete or purely continuous systems, with purely discrete or purely continuous observations.

Based on this restriction, the following four different system descriptions can be distinguished:

(1) Systems with continuous system dynamics and continuous observations.

(2) Systems with continuous system dynamics and discrete observations.

(3) Systems with discrete system dynamics and continuous observations.

(4) Systems with discrete system dynamics and discrete observations.

[2] As it turns out this will not necessarily be the case for the observer, since finite dimensional systems may lead to infinite dimensional observers.

Cases three and four are identical for all practical purposes (assuming that the observation is free of additional dynamics). If the system is to be simulated, the whole system (system dynamics and observation model) needs to be discretized. From an implementation point of view it is easier to work with completely discrete systems. However, the physical meaning of the equations is then usually buried in the discretized equations. Note that by viewing system dynamics together with an observation process, the following sections will introduce the concept of system observers as well[3]. We assume in the following that the given systems are observable.

2.2.1 System Descriptions

Based on the observation above, we need to distinguish three different system models. A continuous system description[4] is given by

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), t), \\ \boldsymbol{y} &= \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{w}(t), t), \end{aligned}$$

where $\boldsymbol{x}$ is the state vector, $\boldsymbol{f}$ and $\boldsymbol{h}$ are the nonlinear vector functions describing the system dynamics and the output respectively, $\boldsymbol{v}$ and $\boldsymbol{w}$ are noise processes, and $\boldsymbol{y}$ represents the output of the system.

A completely discrete system description is given by

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{v}_k, k), \\ \boldsymbol{y}_k &= \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k, k), \end{aligned}$$

where k denotes the discrete time index (e.g., $x(kh) = x_k,\ h > 0,\ h \in \mathbb{R}$). Linear time-invariant systems can easily be transformed from a continuous to a discrete form (if certain assumptions about the signal shape throughout the discretization intervals can be made). Discretization for nonlinear systems is not so straightforward.

[3]The reader is referred to classical texts on control theory for background on state space descriptions, observers, the concept of observability, etc. [49].

[4]The following system descriptions are sufficiently general for the presentational purposes in this thesis. However, not all systems can be written in this way, e.g. differential-algebraic systems.

In between lies the continuous-discrete system description given by

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), t), \\ \boldsymbol{y}_k &= \boldsymbol{h}(\boldsymbol{x}(k), \boldsymbol{u}(k), \boldsymbol{w}_k, k). \end{aligned}$$

2.2.2 The Deterministic Viewpoint

In the deterministic case, $\boldsymbol{v}$ and $\boldsymbol{w}$ are assumed to be zero. Given an initial state $\boldsymbol{x}_0$, the state at any time t can be computed by solving the governing equations for the system. Also, if the state $\boldsymbol{x}_0$ is not known exactly, the state vector $\boldsymbol{x}$ can be observed based on the measurements $\boldsymbol{y}$ if the system is observable. In this deterministic case a relatively simple observer (e.g., a Luenberger observer [89] in the finite dimensional case, or an extension of the Luenberger observer to infinite dimensional systems [141]) is sufficient. In principle, the observer can then converge to the correct state arbitrarily fast (see for example the literature on high-gain observers [131] or [94]). The deterministic problem, becomes more challenging when the system equations themselves are assumed to be known only approximately. This will be the case for most practical problems [40].

2.2.3 The Probabilistic Viewpoint

If noise processes are not neglected, we need to deal with stochastic differential or stochastic difference equations. In the most general setting we would like to devise a scheme to propagate a conditional density function associated with the evolution equation, i.e., how likely is a certain state $\boldsymbol{x}$ given measurements $\boldsymbol{y}$. Given the full conditional density the system's state can easily be determined. For example by a maximum a posteriori (MAP) approach (i.e., by finding the highest peak of the conditional probability function), or by a maximum likelihood (ML) approach (i.e., by calculating the mean state based on the conditional probability density). Depending on the kind of noise process we are looking at, tremendous simplifications can arise. The Gaussian posterior density assumption leads to the Kalman filter in the linear case. However, in most real life situations the noise process is not (or only approximately) Gaussian. Propagation of the full conditional probability

density profile allows for maximum flexibility. Multi-modal distributions pose no problem in this setting and thus allow for the propagation of multiple hypotheses of a systems state. This is an extremely nice property for visual tracking, and can yield highly robust tracking systems [18, 19]. Of course this comes at a price: potentially high computational cost. Whereas in the linear Gaussian case only mean and variance information has to be propagated, a fine enough discretization of the conditional probability density needs to be propagated in the general case. The computational cost increases with the state dimension. A partial remedy to this "curse of dimensionality" are Monte Carlo based sampling approaches [5, 85, 39, 65, 54, 10, 59]. Note, that by propagating a probability density function that includes the influence of measurements we essentially set up an observer.

2.2.3.1 Discrete Systems Dynamics and Discrete Observations

Following [10] we define the measurement vector as

$$\mathbb{Y}_k = \begin{pmatrix} \boldsymbol{y}_0 & \boldsymbol{y}_1 & \dots & \boldsymbol{y}_k \end{pmatrix}^T.$$

Our goal is to estimate the probability distribution of the state $\boldsymbol{x}_k$ conditioned upon the measurements $\mathbb{Y}_k$. Or in mathematical terms: we want to estimate $p(\boldsymbol{x}_k|\mathbb{Y}_k)$. To this end we observe:

$$p(\boldsymbol{x}_k, \boldsymbol{y}_k|\mathbb{Y}_{k-1}) = p(\boldsymbol{x}_k|\mathbb{Y}_k)p(\boldsymbol{y}_k|\mathbb{Y}_{k-1}) = p(\boldsymbol{y}_k|\boldsymbol{x}_k, \mathbb{Y}_{k-1})p(\boldsymbol{x}_k|\mathbb{Y}_{k-1}).$$

Some rearranging yields (this is Bayes' rule)

$$p(\boldsymbol{x}_k|\mathbb{Y}_k) = \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k, \mathbb{Y}_{k-1})p(\boldsymbol{x}_k|\mathbb{Y}_{k-1})}{p(\boldsymbol{y}_k|\mathbb{Y}_{k-1})}$$

which can be simplified to

$$p(\boldsymbol{x}_k|\mathbb{Y}_k) = \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\mathbb{Y}_{k-1})}{p(\boldsymbol{y}_k|\mathbb{Y}_{k-1})} \tag{1}$$

upon assumption that $\boldsymbol{y}_k$ is independent of previous measurements if $\boldsymbol{x}_k$ is known. If furthermore $\boldsymbol{x}_k$ is a Markov process then we can write

$$p(\boldsymbol{x}_{k+1}, \boldsymbol{x}_k|\mathbb{Y}_k) = p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k, \mathbb{Y}_k)p(\boldsymbol{x}_k|\mathbb{Y}_k)$$

and simplify to

$$p(\boldsymbol{x}_{k+1}, \boldsymbol{x}_k|\mathbb{Y}_k) = p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\mathbb{Y}_k).$$

Finally (upon marginalization on $\boldsymbol{x}_k$) we obtain the Chapman-Kolmogorov equation

$$p(\boldsymbol{x}_{k+1}|\mathbb{Y}_k) = \int p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\mathbb{Y}_k)\ d\boldsymbol{x}_k \tag{2}$$

relating the probability distribution at time step $k+1$ to the measurements $\mathbb{Y}_k$. This can be used to update the prior

$$p(\boldsymbol{x}_{k+1}|\mathbb{Y}_{k+1}) = \frac{p(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1})p(\boldsymbol{x}_{k+1}|\mathbb{Y}_k)}{\int p(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1})p(\boldsymbol{x}_{k+1}|\mathbb{Y}_k)d\boldsymbol{x}_{k+1}}. \tag{3}$$

Equations (2) and (3) form the basis for the Bayesian filtering algorithm. The general solution of these equations is difficult and computationally expensive. However, approximate methods have been developed. See [7] for details on solutions based on grid-based filters and particle filters.

2.2.3.2 Continuous System Dynamics and Discrete Observations

What follows will be for a scalar state equation for notational simplicity. Higher dimensional results exist (see [140, 70, 73, 111, 3] for details). Given the continuous state vector $x(t)$ let

$$x_{k+1} = x((k+1)h), \quad x_k = x(kh),$$

where $h > 0,\ h \in \mathbb{R}$ denotes the discrete time step. Then

$$p(x_{k+1}) = \int p(x_{k+1}|x_k)p(x_k)dx_k.$$

By letting $h \to 0$ the evolution equation for $p(x,t)$ can be written by the Kramers-Moyal expansion as [111]

$$\frac{\partial p(x,t)}{\partial t} = \sum_{n=1}^{\infty} \left(-\frac{\partial}{\partial x}\right) \left(D^{(n)}(x,t)p(x,t)\right), \tag{4}$$

where the $D^{(n)}$ are given by the Taylor expansion of the moments

$$M_n(x,t,\tau)/n! = \int \left(x' - x\right)^n p(x', t+\tau|x,t)dx' = D^{(n)}(x,t)\tau + \mathcal{O}(\tau^2).$$

By Pawula's theorem [111], the Kramers-Moyal expansion requires either an infinite number of terms or stops after one or two terms. If the Kramers-Moyal expansion stops after the second term, Equation (4) becomes the Fokker-Planck equation

$$\frac{\partial p(x,t)}{\partial t} = -\frac{\partial}{\partial x}\left(D^{(1)}(x,t)p(x,t)\right) + \frac{\partial^2}{\partial x^2}\left(D^{(2)}(x,t)p(x,t)\right),$$

where $D^{(1)}$ and $D^{(2)}$ are called drift coefficient and diffusion coefficient respectively. Given discrete observations y_k, the probability density can be corrected based on Bayes' formula [95]

$$p(x,t_k|\mathbb{Y}_k) = \frac{p(y_k|x)p(x,t_k|\mathbb{Y}_{k-1})}{\int p(y_k|x)p(x,t_k|\mathbb{Y}_{k-1})dx}.$$

As an example system consider [95]

$$\begin{aligned} d\boldsymbol{x} &= \boldsymbol{f}(\boldsymbol{x},t)dt + \boldsymbol{g}(\boldsymbol{x},t)d\boldsymbol{w} \\ \boldsymbol{y}_k &= \boldsymbol{h}(\boldsymbol{x},t_k) + \sqrt{R}\boldsymbol{v}, \end{aligned}$$

where $\boldsymbol{v}$ is a white sequence of Gaussian random variables, the $m \times m$ matrix R is diagonal and $\boldsymbol{w}$ is a Wiener process. Then the Fokker-Planck equation is

$$\frac{\partial p(\boldsymbol{x},t)}{\partial t} = -\nabla \cdot (\boldsymbol{f}(\boldsymbol{x},t)p(\boldsymbol{x},t)) + \sum_{i,j}\frac{\partial^2}{\partial x_i \partial x_j}\left(\left(\frac{1}{2}\boldsymbol{g}(\boldsymbol{x},t)\boldsymbol{g}^T(\boldsymbol{x},t)\right)_{ij} p(\boldsymbol{x},t)\right)$$

and

$$p(\boldsymbol{y}_k|\boldsymbol{x}) = \frac{1}{(2\pi)^{m/2}\sqrt{|R|}} e^{-\frac{1}{2}(\boldsymbol{y}_k - \boldsymbol{h}(\boldsymbol{x},t_k))^T R^{-1}(\boldsymbol{y}_k - \boldsymbol{h}(\boldsymbol{x},t_k))}.$$

2.2.3.3 Continuous System Dynamics and Continuous Observations

The probability densities in the discrete-discrete case are governed by Equations (2) and (3). To obtain a completely general continuous description would require the analog to the Kramers-Moyal expansion for the Equation set (2) and (3) (i.e., taking the limit as $h \to 0$). This is difficult. As a remedy the observation is usually chosen as the observation process (see [88, 87, 72, 73, 140, 98] for details).

$$y(t) = \int_0^t h(x(s))ds + W(t),$$

where W is a Wiener process. The optimal filter [88] (in the mean square sense) to recover x from the measurements y is then

$$\hat{x}(t) = \frac{\int xp(x,t)dx}{\int p(x,t)dx},$$

where $p = p(x,t)$ is the unnormalized filtering density which is the solution to

$$dp(x,t) = \mathcal{A}^* p(x,t) + h(x)p(x,t)dY(t),$$

the Zakai equation. Here $\mathcal{A}^*$ is the adjoint (solution) operator to the operator for the Markov process governing the evolution of x (e.g., for the Fokker-Planck equation of Section 2.2.3.2 $\mathcal{A}p(x,t) = -\frac{\partial}{\partial x}\left(D^{(1)}(x,t)p(x,t)\right) + \frac{\partial^2}{\partial x^2}\left(D^{(2)}(x,t)p(x,t)\right)$). The normalized filtering density is governed by the solution to the Kushner equation [98]

$$d\tilde{p}(x,t) = \mathcal{A}^*\tilde{p}(x,t)dt + \tilde{p}(x,t)\left(h(x) - \int h(x)\tilde{p}(x,t)dx\right)dY(t) \tag{5}$$

which is a nonlinear stochastic differential equation, whereas the Zakai equation is a linear stochastic differential equation and is thus usually preferred for computational purposes. A special case is the case where both the state and the observation evolutions are governed by diffusion equations. Then, this is equivalent to the Kalman-Bucy filter (i.e., the continuous time Kalman filter). As a matter of fact, all the cases presented in the previous sections have as special cases the Kalman filter. See [53] for Kalman filtering results for the discrete-discrete, continuous-discrete and continuous-continuous cases. In summary, we observe the following: In the case of discrete observations, the observations update the probability distribution in an update step after evolution of the system for some finite time period. In the completely continuous case this update step is replaced by the continuous error injection term

$$h(x) - \int h(x)\tilde{p}(x,t)dx$$

as can be seen in the Kushner Equation (5). Dealing with nonlinear dynamical systems in complete generality is difficult. As demonstrated above, the governing equations get complex even for systems with a finite number of states; a finite dimensional system may require an infinite dimensional observer. The dynamic curve evolution problem considered in

this thesis is infinite dimensional. Observer design for such systems is still in its infancy [141]. To deal with this problem, the error injection idea presented in this section will be used in Section 4.1 to design an infinite dimensional observer-like system.

2.3 Cubical Homology

This section gives a brief introduction to homology. It will be used in Section 5.2 to define the concept of a simple point in a d-dimensional space.

Homology aims at counting 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. In the cubical homology these units are pixels/voxels and their respective vertices, edges and higher-dimensional faces. Cubical homology is ideally suited for digital images, due to its ability to handle voxels or pixels directly. Whereas homology is by now a standard tool of algebraic topology [92], cubical homology is more recent [68, 69].

Formally, an *elementary cube* Q is given by the finite product [67]

$$Q = I_1 \times I_2 \times \cdots \times I_d \subset \mathbb{R}^d, \tag{6}$$

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 $k \in \mathbb{N}$.

Definition 1

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

$$\mathcal{K}(X) := \{Q \in \mathcal{K} \mid Q \subset X\} \text{ and}$$
$$\mathcal{K}_k(X) := \{Q \in \mathcal{K}(X) \mid \dim Q = k\}.$$

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 of k-chains, $C_k(X)$, by declaring each element of $\mathcal{K}_k(X)$ to be a distinct generator (or basis element). Let C_k denote $C_k(\mathcal{R}^d)$.

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

$$\partial_k : C_k \to C_{k-1} \tag{7}$$

is the group homomorphism defined on every elementary cube $Q \in \mathcal{K}_k$ 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 by

$$Z_k(X) := \ker \partial_k^X = C_k(X) \cap \ker \partial_k \tag{8}$$

$$B_k(X) := \text{image } \partial_{k+1}^X = \partial_{k+1}\left(C_{k+1}(X)\right) \tag{9}$$

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

$$\beta_k(X) := \text{rank}(H_k(X)). \tag{10}$$

The elements of $H_k(X)$ are called the *k-generators of X*. The homology groups are computed as (for example) described in [69, 108]. Software to compute the Betti-numbers of a cubical complex is freely available (`http://www.math.gatech.edu/~chom`). A very useful result to compute the homology of a cubical set is the Mayer-Vietoris theorem:

Theorem 1 (Mayer-Vietoris)

Let X be a cubical space, then given the two cubical sets $A_0 \subset X$ and $A_1 \subset X$, such that $X = A_0 \cup A_1$ and $B = A_0 \cap A_1$, there is a long exact sequence

$$\cdots \to H_k(B) \to H_k(A_0) \oplus H_k(A_1) \to H_k(X) \to H_{k-1}(B) \to \ldots,$$

where an exact sequence is defined as [68]:

Definition 3

A finite (or infinite) sequence of groups and homomorphisms

$$\cdots \rightarrow G_3 \stackrel{\Psi_3}{\rightarrow} G_2 \stackrel{\Psi_2}{\rightarrow} G_1 \rightarrow \ldots$$

is exact *at* G_2 *if*

$$ker\ \Psi_2 = im\ \Psi_3.$$

It is an exact sequence *if it is exact at every group.*

For details regarding the theory of cubical homology and a proof of the Mayer-Vietoris theorem the reader should consult the excellent book by Kaczynski *et al.* [68].

CHAPTER 3

CURVE EVOLUTION THEORY

3.1 Parameterized Dynamic Curve Evolution

This section reviews parameterized dynamic curve evolution [129] and introduces the mathematical setup required to derive the geometric dynamic curve evolution equations of Section 3.2.

Consider the evolution of closed curves of the form $\mathcal{C} : S^1 \times [0, \tau) \mapsto \mathbb{R}^2$ in the plane, where $\mathcal{C} = \mathcal{C}(p, t)$ and $\mathcal{C}(0, t) = \mathcal{C}(1, t)$ [128, 76], with t being the time, and $p \in [0, 1]$ the curve's parameterization (see Figure 1 for an illustration). The classical formulation for dynamic curve evolution as proposed by Terzopoulos and Szeliski [129] is derived by means of minimization of the action integral

$$\mathcal{L} = \int_{t=t_0}^{t_1} L(t, \mathcal{C}, \mathcal{C}_t)\ dt, \tag{11}$$

where subscripts denote partial derivatives (here with respect to the time t). The Lagrangian $L = T - U$ is the difference between the kinetic and the potential energy. The potential energy of the curve is given by

$$\begin{aligned} U &= \int_0^1 U_{el} + U_{rig} + U_{pf}\ dp \\ &= \int_0^1 \frac{1}{2} w_1 \|\mathcal{C}_p\|^2 + \frac{1}{2} w_2 \|\mathcal{C}_{pp}\|^2 + g(\mathcal{C})\ dp, \end{aligned}$$

where g is some potential function (with the desired location of the curve forming a potential well) and U_{el}, U_{rig}, and U_{pf} are the elasticity, rigidity and potential field contributions respectively, with their (possibly position-dependent) scalar weights w_1, and w_2. A common choice for the potential function is

$$g(\boldsymbol{x}) = \frac{1}{1 + \|G * \nabla I(\boldsymbol{x})\|^r}, \tag{12}$$

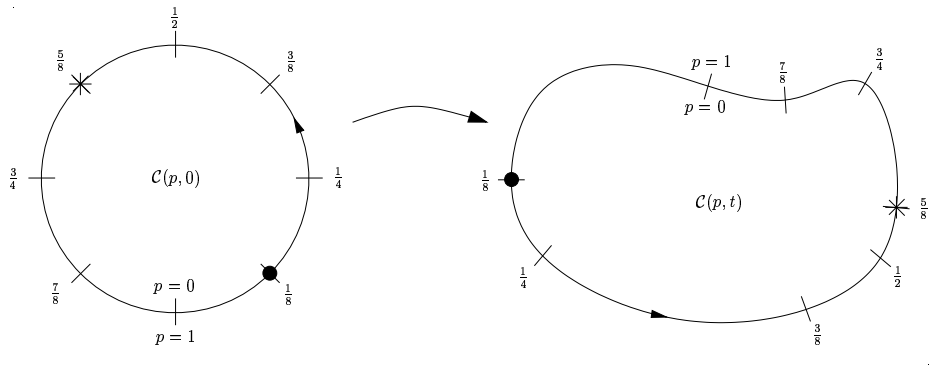


Figure 1: For parameterized curve evolution the parameterization travels with a particle. In general, the parameterization will not stay uniformly spaced. The black disk and the asterisk indicate particles attached to the curve; their assigned value for p will stay the same throughout the evolution.

where $\boldsymbol{x} = [x, y]^T$ are the image coordinates, I is (for example) the image intensity, r is a positive integer, and G is a Gaussian of variance σ^2. The kinetic energy is

$$T = \int_0^1 \frac{1}{2}\mu \|\mathcal{C}_t\|^2 \, dp,$$

where μ corresponds to mass per unit length. The Lagrangian used is then

$$L = \int_0^1 \frac{1}{2}\mu \|\mathcal{C}_t\|^2 - \frac{1}{2} w_1 \|\mathcal{C}_p\|^2 - \frac{1}{2} w_2 \|\mathcal{C}_{pp}\| - g(\mathcal{C}) \, dp. \tag{13}$$

Computing the first variation $\delta\mathcal{L}$ of the action integral (11) and setting it to zero yields the Euler-Lagrange equations for the candidate minimizer [132] in force balance form:

$$\mu \mathcal{C}_{tt} = \frac{\partial}{\partial p}(w_1 \mathcal{C}_p) - \frac{\partial^2}{\partial p^2}(w_2 \mathcal{C}_{pp}) - \nabla g. \tag{14}$$

Note, that the right hand side of Equation (14) corresponds to a force (compare with Newton's second law $ma = F$, where m is the mass, a the acceleration, and F the force), where ∇g is the force exerted by the image on the curve. The capture range of the potential force ∇g will depend on the variance of the Gaussian in Equation (12). If a greater capture range is desired, ∇g can be replaced by a more general force [142], e.g., a force based on the gradient vector flow field. Equation (14) depends on the parameterization p and is therefore not geometric (see Xu *et al.* [143] for a discussion on the relationship between

parametric and geometric active contours). The proposed methodology (see Section 3.2) will be completely independent of parameterization. It will be geometric.

3.2 Geometric Dynamic Curve Evolution

This section presents the geometric dynamic curve evolution equations, which are geometric and evolve according to *physically motivated time*. Minimizing equation (11) using the Lagrangian

$$L = \int_0^1 \left(\frac{1}{2}\mu \|\mathcal{C}_t\|^2 - g \right) \|\mathcal{C}_p\| \; dp,$$

instead of the Lagrangian (13), results in (see A.1 for a derivation)

$$\mu \mathcal{C}_{tt} = -\mu(\mathcal{T} \cdot \mathcal{C}_{ts})\mathcal{C}_t - \mu(\mathcal{C}_t \cdot \mathcal{C}_{ts})\mathcal{T} - \left(\frac{1}{2}\mu \|\mathcal{C}_t\|^2 - g \right) \kappa \mathcal{N} - (\nabla g \cdot \mathcal{N})\mathcal{N}, \tag{15}$$

which is geometric and a natural extension of the geodesic active contour approach [22, 75]. Here, $\mathcal{N}$ is the unit inward normal, $\mathcal{T} = \frac{\partial \mathcal{C}}{\partial s}$ the unit tangent vector to the curve, $\kappa = \mathcal{C}_{ss} \cdot \mathcal{N}$ denotes curvature and s is the arclength parameter [38].

The term $(g\kappa - \nabla g \cdot \mathcal{N})\mathcal{N}$ in Equation (15) is a force exerted by the image potential g on the curve $\mathcal{C}$. Compare this to the evolution equation for geodesic active contours as given in [113, 128] ($\mathcal{C}_t = (g\kappa - \nabla g \cdot \mathcal{N})\mathcal{N}$).

From a controls perspective this can be interpreted as a control law based on g and its spatial gradient ∇g, which is designed to move the curve closer to the bottom of the potential well formed by g.

The control law will not guarantee perfect tracking, since the potential forces associated with g will have to outweigh the dynamical forces. As a remedy, proportional integral (PI) control (optionally with an anti-windup scheme) of the form

$$\begin{aligned} \mu \mathcal{C}_{tt} = -\mu(\mathcal{T} \cdot \mathcal{C}_{ts})\mathcal{C}_t - \mu(\mathcal{C}_t \cdot \mathcal{C}_{ts})\mathcal{T} - \frac{1}{2}\mu \|\mathcal{C}_t\|^2 \kappa \mathcal{N} + \left(\alpha_p g + \alpha_i \int g \; dt \right) \kappa \mathcal{N} - \\ - \left((\mathrm{diag}(\boldsymbol{\beta}_p) \nabla g + \mathrm{diag}(\boldsymbol{\beta}_i) \int \nabla g \; dt) \cdot \mathcal{N} \right) \mathcal{N} \end{aligned} \tag{16}$$

can be used, where α_p, α_i, $\boldsymbol{\beta}_p$, $\boldsymbol{\beta}_i$ are controller parameters (α_p and α_i are scalars, $\boldsymbol{\beta}_p$ and $\boldsymbol{\beta}_i$ are vectors in $\mathbb{R}^2$), and $\mathrm{diag}(\boldsymbol{\beta})$ denotes the diagonal matrix with $\mathrm{diag}(\boldsymbol{\beta})_{kk} = \boldsymbol{\beta}_k$. We

assume that g and ∇g vanish at the desired location of the curve. For the sake of notational simplicity we will continue to use Equation (15) in what follows.

Equations (15,16) describe a curve evolution that is only influenced by inertia terms and information on the curve itself. To increase robustness, the potential energy U can include region-based terms (see for example [106, 145, 144]). This would change the evolution Equations (14,15,16), but such changes pose no problem to the overall approach.

The state-space form of equation (15) is

$$\boldsymbol{x}_t(s,t) = \begin{pmatrix} x_3(s,t) \\ x_4(s,t) \\ f_1(\boldsymbol{x}) \\ f_2(\boldsymbol{x}) \end{pmatrix}, \tag{17}$$

where $\boldsymbol{x}^T = [x_1, x_2, x_3, x_4]^T$, $x_1 = x(s,t)$, $x_2 = y(s,t)$, $x_3 = x_t(s,t)$, $x_4 = y_t(s,t)$, and f_i are scalar functions in $\boldsymbol{x}$ and its derivatives. The evolution describes the movement of a curve in $\mathbb{R}^4$, where the geometrical shape can be recovered by the simple projection

$$\Pi(\boldsymbol{x}) = \begin{pmatrix} x_1(s,t) \\ x_2(s,t) \end{pmatrix}.$$

3.2.1 Interpretation of the Evolution Terms for the Geometric Dynamic Curve Evolution

To get an understanding of Equation (15), it is fruitful to look at the effect of its individual terms. The term

$$-\left(\nabla g \cdot \mathcal{N}\right)\mathcal{N}$$

accelerates the curve $\mathcal{C}$ towards[1] the potential well formed by g. The term

$$-a(g,\mathcal{C}_t)\kappa\mathcal{N} = -\left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)\kappa\mathcal{N}$$

accelerates the curve $\mathcal{C}$ based on its smoothness properties and

$$-\mu\left(\mathcal{T} \cdot \mathcal{C}_{ts}\right)\mathcal{C}_t \tag{18}$$

[1] $-\nabla g$ points towards the potential well.

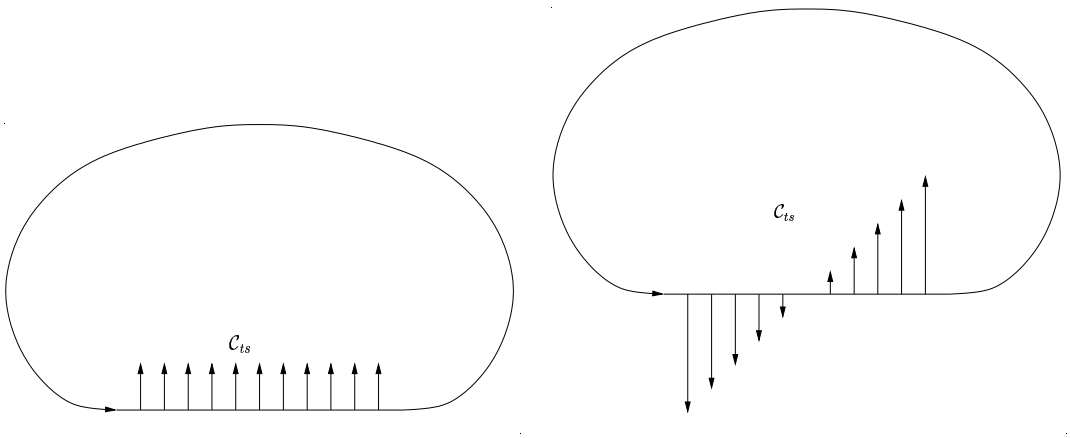


Figure 2: Curve propagation in the normal direction, $\mathcal{C}_{ts}$ constant and linearly increasing.

represents a smoothing term for the tangential velocity. The velocity change $\mathcal{C}_{ts}$ at every point on the curve $\mathcal{C}$ decomposes into its tangential and normal components as

$$\mathcal{C}_{ts} = (\mathcal{C}_{ts} \cdot \mathcal{N})\,\mathcal{N} + (\mathcal{C}_{ts} \cdot \mathcal{T})\,\mathcal{T}.$$

Assume that the tangential and the normal components change approximately linearly close to the point of interest. A Taylor series expansion (at arclength s_0) yields

$$\begin{aligned}
(\mathcal{C}_{ts} \cdot \mathcal{N})(s) &= (\mathcal{C}_{ts} \cdot \mathcal{N})(s_0) + \frac{\partial(\mathcal{C}_{ts} \cdot \mathcal{N})}{\partial s}\bigg|_{s_0} (s - s_0) + \mathcal{O}(s^2) = n_0 + n_1(s - s_0) + \mathcal{O}(s^2), \\
(\mathcal{C}_{ts} \cdot \mathcal{T})(s) &= (\mathcal{C}_{ts} \cdot \mathcal{T})(s_0) + \frac{\partial(\mathcal{C}_{ts} \cdot \mathcal{T})}{\partial s}\bigg|_{s_0} (s - s_0) + \mathcal{O}(s^2) = t_0 + t_1(s - s_0) + \mathcal{O}(s^2).
\end{aligned}$$

In order to appreciate the effect of the term (18), it is sufficient to consider the two fundamental cases depicted in Figures 2 and 3. The normal component (depicted in Figure 2) is irrelevant for the evolution, since $\mathcal{T} \cdot \mathcal{C}_{ts} = 0$ in this case. The tangential component (depicted in Figure 3) will counteract tangential gradients of $\mathcal{C}_{ts}$. The two cases correspond to a linearly and a parabolically increasing velocity $\mathcal{C}_t$ in the tangential direction. In both cases, the term $-\mu\,(\mathcal{T} \cdot \mathcal{C}_{ts})\,\mathcal{C}_t$ will counteract this tendency of tangentially diverging particles on the curve, ideally smoothing out the tangential velocities over the curve $\mathcal{C}$.

The term

$$-\mu\,(\mathcal{C}_t \cdot \mathcal{C}_{ts})\,\mathcal{T}$$

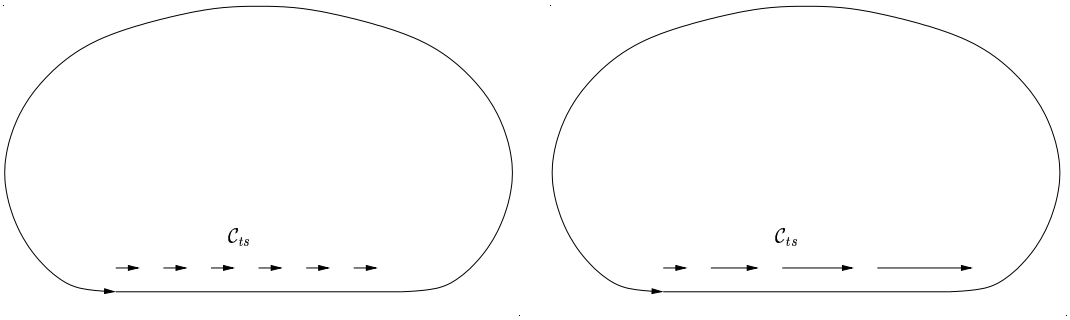


Figure 3: Curve propagation in the tangential direction, $\mathcal{C}_{ts}$ constant and linearly increasing.

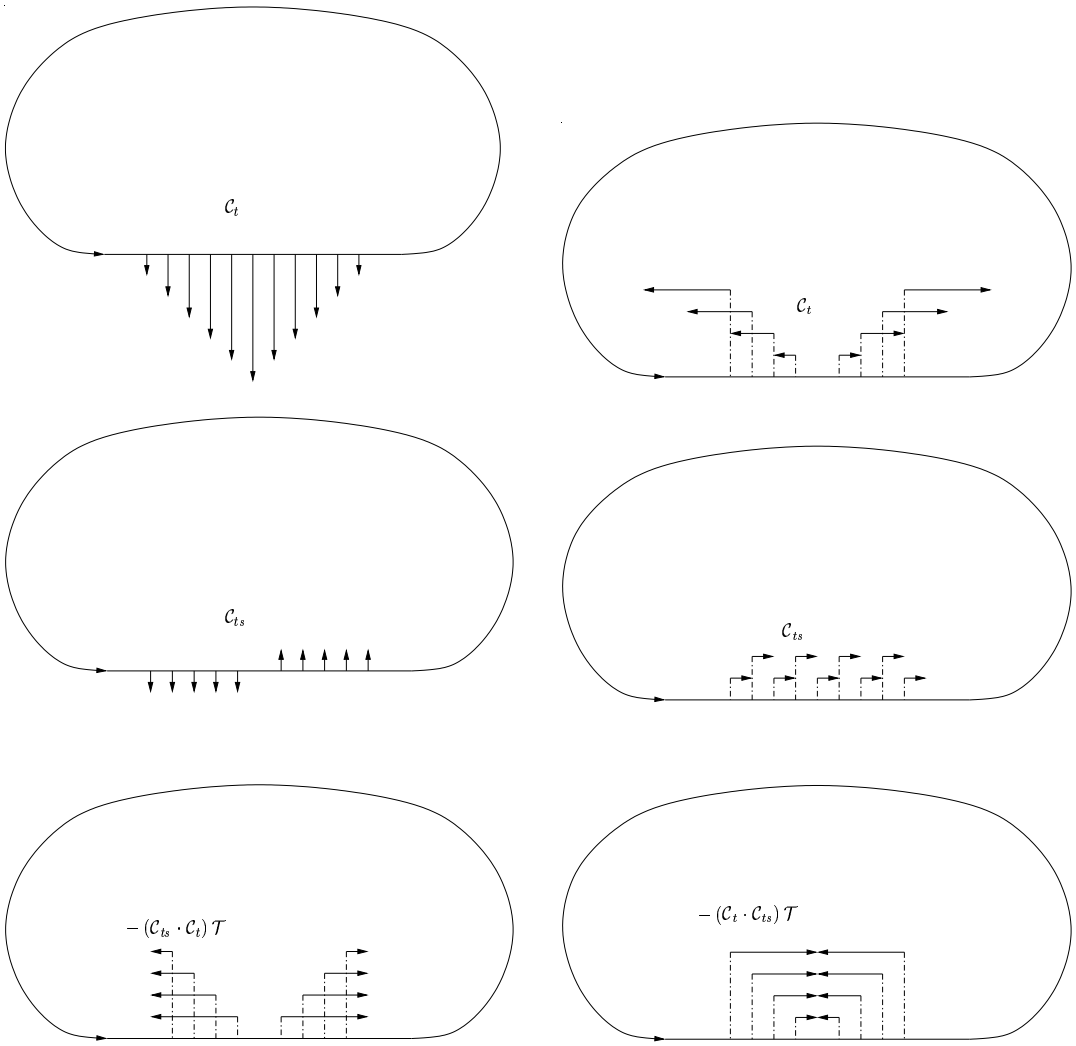
governs the transport of particles along the tangential direction. To understand what is occurring locally, assume looking at a locally linear piece of the curve. Then the velocity decomposes into

$$\mathcal{C}_t = (\mathcal{C}_t \cdot \mathcal{T})\,\mathcal{T} + (\mathcal{C}_t \cdot \mathcal{N})\,\mathcal{N}.$$

It is instructive to look at a triangular velocity shape $\mathcal{C}_t$ in the normal direction (as shown in Figure 4(a)) and in the tangential direction (as shown in Figure 4(b)). The triangular velocity shape in the normal direction induces a tangential movement of particles on the curve. This can be interpreted as a rubberband effect. Assume that the rubberband gets pulled at one point. This will elongate the rubberband. Since the point at which it is pulled stays fixed (no movement except for the displacement due to the pulling) particles next to it flow away from it. The triangular velocity shape in the tangential direction also induces tangential motion of the particles. However, this motion will counteract the initial tangential direction and will thus also lead to a smoothing effect on the change of tangential velocity vector over arclength.

3.3 Simple Area Based Dynamic Curve Evolution

So far, we looked at dynamic curve evolution based on an edge-based potential function. It is simple and fast, but generally not very robust. For the static case, region (or area) based formulations have been proposed, which replace the edge-based potential function, by an



(a) Normal (b) Tangential

Figure 4: Behavior of the curve evolution term $-\mu\,(\mathcal{C}_t \cdot \mathcal{C}_{ts})\,\mathcal{T}$.

expression based on area intensity, statistics, etc. Here, greater robustness is achieved by giving up on flexibility and making stronger assumptions about the objects to be captured.

We can easily derive a dynamic curve evolution equation from an area based static formulation. To do so, simply split the action integral into

$$\mathcal{L} = \int_{t=t_0}^{t_1} \left(\int_{p=0}^{1} \frac{1}{2}\mu \|\mathcal{C}_t\|^2 \|\mathcal{C}_p\| \; dp \right) - g_a \; dt, \tag{19}$$

where g_a denotes the area based part of the action integral[2]. As an example, this section demonstrates this methodology in case of the approach proposed by Yezzi *et al.* [145]. Define (following [145])

$$g_a(u, w) = -\frac{1}{2}\left(u - w\right)^2,$$

where $u = S_u/A_u$ and $w = S_w/A_w$, where

$$\begin{aligned} S_u &= \int_{R^u} I \; dA \qquad & A_u &= \int_{R^u} dA \\ S_w &= \int_{R^w} I \; dA \qquad & A_w &= \int_{R^w} dA, \end{aligned}$$

and R^u and R^w denote the domains inside and outside the curve respectively. Computing the first variation and setting it to zero results in (see A.2 for a derivation)

$$\mu \mathcal{C}_{tt} = -\mu(\mathcal{T} \cdot \mathcal{C}_{ts})\mathcal{C}_t - \mu(\mathcal{C}_t \cdot \mathcal{C}_{ts})\mathcal{T} - \frac{1}{2}\mu\|\mathcal{C}_t\|^2 \kappa \mathcal{N} - (u - w)\left(\frac{I - u}{A_u} + \frac{I - w}{A_w}\right)\mathcal{N}.$$

The formulation (19) suggests keeping a curve based term for the velocity part of the evolution equation. However, this is not necessary since the term g_a can be chosen as desired. See Section (3.4) for a "completely" area based approach.

3.4 Elastic Body Deformation

Given a set A on a domain $\Omega \subset \mathbb{R}^2$ and a displacement $\boldsymbol{u} = (u^x \; , \; u^y)^T$ associated with every point in A (see Figure 5)

$$T = \frac{1}{2}\rho \int_A \|\boldsymbol{u}_t\|^2 \; dA$$

[2]Generally, all static (energy-based) approaches can be cast into the dynamic framework by using the static energy as the potential energy P of the action integral (19), where $P = g_a$.

is the kinetic energy of the body B described by the set A, ρ denotes the density of B. The elastic potential energy can be defined (see [121, 1] for details) for a volume V as

$$U_{el}(\boldsymbol{u}) = \int_V W \; dV - \int_{\partial V} \boldsymbol{T}^T \boldsymbol{u} \; d(\partial V) - \int_V \boldsymbol{f}^T \boldsymbol{u} \; dV,$$

where W is the strain energy density, $\boldsymbol{T}$ are the surface tractions, $\boldsymbol{f}$ is the body force, and $\boldsymbol{u}$ are the displacements in three dimensions. For an isotropic material the strain energy density is [58]

$$W = \mu \epsilon_{ij} \epsilon_{ij} + \frac{\lambda}{2} (\epsilon_{kk})^2,$$

using the usual Einstein summation convention (i.e., summing over indices occurring more than once); ϵ denotes strain, and μ and λ are the Lamé constants. Following Gould [58] the first variation of U_{el} is

$$\delta U_{el}(u_l; v_l) = - \int_V \left(\mu (u_l)_{,ii} + (\lambda + \mu)(u_{i,i})_{,l} + f_l\right) v_l \; dV \tag{20}$$

where the v_l denote the perturbations. Looking at the two-dimensional (planar) plane strain problem, all field variables become independent of z (see [1]). If furthermore there are no body forces, initial displacements or velocities present in the z-direction, Equation (20) reduces to its two dimensional case and becomes

$$\begin{aligned} \delta U_{el}(u_l; v_l) &= - \int_V \left(\mu (u_l)_{,ii} + (\lambda + \mu)(u_{i,i})_{,l} + f_l\right) \delta(z) v_l \; dV \\ &= - \int_A \left(\mu (u_l)_{,ii} + (\lambda + \mu)(u_{i,i})_{,l} + f_l\right) v_l \; dA. \end{aligned}$$

Defining the boundary potential energy as[3]

$$U_{bd} = \int_{\partial A} g \; ds, \tag{21}$$

where s is arclength and g is some potential function, yields the variation

$$\delta U_{bd}(\boldsymbol{u}; \boldsymbol{v}) = \int_{\partial A} (-(\nabla g \cdot \mathcal{N} + \kappa g)\mathcal{N}) \cdot \boldsymbol{v} \; ds = \int_A (-(\nabla g \cdot \mathcal{N} + \kappa g)\mathcal{N}) \, \delta(\partial A) \cdot \boldsymbol{v} \; dA, \tag{22}$$

where $\delta(\partial A)$ is the Dirac Delta function on the set ∂A. Defining the Lagrangian as

$$L = T - (U_{bd} + U_{el}) \tag{23}$$

[3]In this setting it would probably also make a lot of sense to use an area based formulation for the potential energy, e.g. based on the separation of means, statistics, etc.

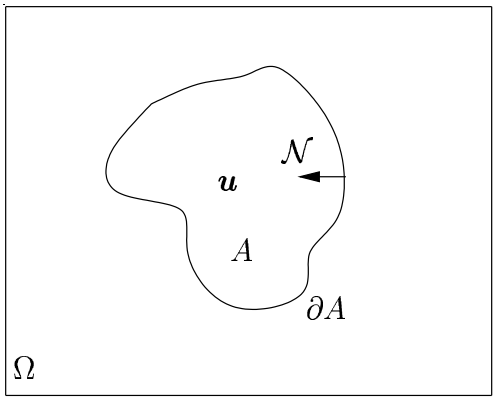


Figure 5: Object domain with object for the elastic body deformation.

and computing the first variation of the action integral

$$\mathcal{L} = \int_t L \; dt, \tag{24}$$

yields (using the notation $\Delta \boldsymbol{u} = (\Delta u_1 \; \Delta u_2)^T$ and Equations (20) and (22))

$$\delta\mathcal{L}(\boldsymbol{u};\boldsymbol{v}) = \int_t \int_A (-\rho \boldsymbol{u}_{tt} + (-\nabla g \cdot \mathcal{N} + \kappa g)\, \delta(\partial A)\mathcal{N}) \cdot \boldsymbol{v} \; dA \; dt - \\ - \int_t \int_A (\boldsymbol{f} + \mu \Delta \boldsymbol{u} + (\lambda + \mu)\nabla(div(\boldsymbol{u}))) \cdot \boldsymbol{v} \; dA \; dt.$$

This results in the sought for evolution equation on A

$$\rho \boldsymbol{u}_{tt} = (-\nabla g \cdot \mathcal{N} + \kappa g)\, \delta(\partial A)\mathcal{N} - \mu \Delta \boldsymbol{u} - (\lambda + \mu)\nabla(div(\boldsymbol{u})). \tag{25}$$

The domain A is dependent on time. The only image dependent term is U_{bd} (as given in Equation (21)). It attracts the contour to the object boundary and ensures boundary smoothness. Augmenting the scheme by any kind of area-based image dependent term will increase robustness. One possibility would be to add the image term from Section 3.3 to the elastic area evolution equation (25). It then becomes

$$\rho \boldsymbol{u}_{tt} = (-\nabla g \cdot \mathcal{N} + \kappa g)\, \delta(\partial A)\mathcal{N} - \mu \Delta \boldsymbol{u} - (\lambda + \mu)\nabla(div(\boldsymbol{u})) - (u - w)\left(\frac{I - u}{A_u} + \frac{I - w}{A_w}\right)\mathcal{N}. \tag{26}$$

3.5 Normal Geometric Dynamic Curve Evolution

To get more insight into the behavior of the curve evolution Equation (15), it is instructive to derive the evolution equations for the tangential and normal velocity components of the curve.

The general version of the geometric dynamic curve evolution equation is given as

$$\begin{aligned}\mu\mathcal{C}_{tt} &= -\mu(\mathcal{T}\cdot\mathcal{C}_{ts})\mathcal{C}_t - \frac{\partial}{\partial s}\left(\left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)\mathcal{C}_s\right) - \nabla g \\ &= -\mu(\mathcal{T}\cdot\mathcal{C}_{ts})\mathcal{C}_t - \mu(\mathcal{C}_t\cdot\mathcal{C}_{ts})\mathcal{T} - \left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)\kappa\mathcal{N} - (\nabla g\cdot\mathcal{N})\mathcal{N}, \qquad (27)\end{aligned}$$

where $\mathcal{N}$ is the unit inward normal and

$$\begin{aligned}\mathcal{T}_s &= \kappa\mathcal{N}, \\ \mathcal{N}_s &= -\kappa\mathcal{T}.\end{aligned}$$

The curve speed $\mathcal{C}_t$ can be decomposed as

$$\mathcal{C}_t = \alpha(p,t)\mathcal{T} + \beta(p,t)\mathcal{N}, \qquad (28)$$

where the parameterization p is independent of time and travels with its particle (i.e., every particle corresponds to a specific value p for all times), and α and β correspond to the tangential and the normal speed functions respectively.

The general (without prespecified special reparameterization ϕ) evolution equations for α and β (see [77] for details on some of the used equations) are derived in what follows. Using an arbitrary curve parameterization p (with $\mathcal{C}(p,0) = \mathcal{C}(p,1)$, $\mathcal{C} = \mathcal{C}(p,t)$, and $p \in [0,1]$) define

$$G(p,t) := \|\mathcal{C}_p\| = (x_p^2 + y_p^2)^{\frac{1}{2}}.$$

Arclength is then given by

$$s(p,t) := \int_0^p G(\xi,t)\, d\xi.$$

Then

$$\frac{\partial}{\partial t}\frac{\partial}{\partial s} = -\frac{1}{G}(\alpha_p - \beta\kappa G)\frac{\partial}{\partial s} + \frac{\partial}{\partial s}\frac{\partial}{\partial t}.$$

It follows that [77]

$$G_t = \alpha_p - \beta\kappa G.$$

The expressions above yield

$$\mathcal{N}_t = -(\beta_s + \alpha\kappa)\mathcal{T},$$

and

$$\mathcal{T}_t = (\beta_s + \alpha\kappa)\mathcal{N}.$$

From this follows

$$\begin{aligned}
\mathcal{C}_{tt} &= (\mathcal{C}_t)_t = (\alpha T + \beta N)_t \\
&= \alpha_t\mathcal{T} + \alpha(\beta_s + \alpha\kappa)\mathcal{N} + \beta_t\mathcal{N} - \beta(\beta_s + \alpha\kappa)\mathcal{T} \\
&= (\alpha_t - \beta\beta_s - \alpha\beta\kappa)\mathcal{T} + (\alpha\beta_s + \alpha^2\kappa + \beta_t)\mathcal{N}
\end{aligned}$$

and

$$\begin{aligned}
\mathcal{C}_{ts} &= (\alpha\mathcal{T} + \beta\mathcal{N})_s \\
&= \alpha_s\mathcal{T} + \alpha\kappa\mathcal{N} + \beta_s\mathcal{N} - \beta\kappa\mathcal{T} \\
&= (\alpha_s - \beta\kappa)\mathcal{T} + (\alpha\kappa + \beta_s)\mathcal{N}.
\end{aligned}$$

Some simple algebra yields

$$\mu\left[(\alpha_t - 2\alpha\beta\kappa + 2\alpha\alpha_s)\mathcal{T} + \left(\frac{3}{2}\kappa\alpha^2 - \frac{1}{2}\kappa\beta^2 + \alpha_s\beta + \alpha\beta_s + \beta_t\right)\mathcal{N}\right] = g\kappa\mathcal{N} - (\nabla g\cdot\mathcal{N})\mathcal{N}.$$

Or after some rearranging

$$\mu\left[(\alpha_t + 2\alpha\alpha_s)\mathcal{T} + (\alpha_s\beta + \alpha\beta_s + \beta_t)\mathcal{N}\right] = (2\alpha\beta\mu\mathcal{T} + \left(\frac{1}{2}\beta^2 - \frac{3}{2}\alpha^2\right)\mu\mathcal{N} + g\mathcal{N})\kappa - (\nabla g\cdot\mathcal{N})\mathcal{N}.$$

Since

$$\mathcal{T} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\mathcal{N},$$

it follows

$$\begin{aligned}
&\mu\begin{pmatrix} \alpha_s\beta + \alpha\beta_s + \beta_t & -\alpha_t - 2\alpha\alpha_s \\ \alpha_t + 2\alpha\alpha_s & \alpha_s\beta + \alpha\beta_s + \beta_t \end{pmatrix}\mathcal{N} = \\
&\qquad \mu\kappa\begin{pmatrix} \left(\frac{1}{2}\beta^2 - \frac{3}{2}\alpha^2\right) + \frac{1}{\mu}g & -2\alpha\beta \\ 2\alpha\beta & \left(\frac{1}{2}\beta^2 - \frac{3}{2}\alpha^2\right) + \frac{1}{\mu}g \end{pmatrix}\mathcal{N} - \begin{pmatrix} \frac{1}{\mu}\nabla g\cdot\mathcal{N} & 0 \\ 0 & \frac{1}{\mu}\nabla g\cdot\mathcal{N} \end{pmatrix}\mathcal{N}. \quad (29)
\end{aligned}$$

This must be true for all $\mathcal{N}$. Equation (29) reduces to the following two coupled partial differential equations:

$$\begin{aligned} \alpha_t &= -(\alpha^2)_s + 2\kappa\alpha\beta, \\ \beta_t &= -(\alpha\beta)_s + \left[\left(\frac{1}{2}\beta^2 - \frac{3}{2}\alpha^2\right) + \frac{1}{\mu}g\right]\kappa - \nabla g \cdot \mathcal{N}. \end{aligned} \tag{30}$$

Here, $-(\alpha^2)_s$ and $-(\alpha\beta)_s$ are the transport terms for the tangential and the normal velocity along the contour, and $g\kappa - \nabla g \cdot \mathcal{N}$ is the well known geodesic active contour image influence term [75, 22]. In contrast to the static geodesic active contour, this term influences the curve's normal velocity rather than directly the curve's position. It can be interpreted as a force. Finally, the terms $2\kappa\alpha\beta$ and $(\frac{1}{2}\beta^2 - \frac{3}{2}\alpha^2)\kappa$ incorporate the dynamic elasticity effects of the curve. Envisioning a rotating circle, the term $(\frac{1}{2}\beta^2 - \frac{3}{2}\alpha^2)\kappa$ can be interpreted as a rubberband (i.e., if we rotate the circle faster it will try to expand, but at the same time it will try to contract due to its then increasing normal velocity; oscillations can occur). Restricting the movement of the curve to its normal direction (i.e., setting $\alpha = 0$) yields (see Appendix A.3 for an alternative derivation)

$$\beta_t = \frac{1}{2}\beta^2\kappa + \frac{1}{\mu}g\kappa - \frac{1}{\mu}\nabla g \cdot \mathcal{N}. \tag{31}$$

This is a much simpler evolution equation. It is identical to the full evolution Equation (30) if the initial tangential velocity is zero. The image term, g, only influences the normal velocity evolution β. It does not create any additional tangential velocity. Thus, if $\alpha = 0\ \forall s$, then $\alpha = 0\ \forall s, t$; the flow with $\alpha = 0$ is contained in (27) as an invariant subsystem.

If there is an initial tangential velocity, and/or if the image influence g contributes to the normal velocity β *and* to the tangential velocity α, the normal evolution equation will not necessarily be equivalent to the full evolution Equation (30). There is always a curve parameterization resulting in an evolution equation free of tangential velocities. Specifically, considering a reparameterization

$$\overline{\mathcal{C}}(q,t) = \mathcal{C}(\phi(q,t),t),$$

where $\phi : \mathbb{R} \times [0,T) \mapsto \mathbb{R}, p = \phi(q,t), \phi_q > 0$ then

$$\frac{\partial \overline{\mathcal{C}}}{\partial t} = \frac{\partial \mathcal{C}}{\partial t} + \frac{\partial \mathcal{C}}{\partial p}\frac{\partial \phi}{\partial t}.$$

The time evolution for $\overline{C}$ can then be decomposed into

$$\overline{\mathcal{C}}_t = \overline{\alpha}\mathcal{T} + \overline{\beta}\mathcal{N} = (\alpha(\phi(q,t),t) + \|\mathcal{C}_p(\phi(q,t),t)\|\phi_t)\mathcal{T} + \overline{\beta}\mathcal{N},$$

where

$$\begin{aligned} \overline{\alpha} &= \alpha(\phi(q,t),t) + \|\mathcal{C}_p(\phi(q,t),t)\|\phi_t \\ \overline{\beta} &= \beta(\phi(q,t),t). \end{aligned}$$

Choosing ϕ as

$$\phi(q,t)_t = -\frac{\alpha(\phi(q,t),t)}{\|\mathcal{C}_p(\phi(q,t),t)\|} \tag{32}$$

results in

$$\overline{\mathcal{C}}_t = \overline{\beta}\mathcal{N},$$

which is a curve evolution equation without a tangential component. For all times the curve $\overline{\mathcal{C}}$ will move along its normal direction. However, the tangential velocity is still present in the update equation for $\overline{\beta}$. Compute $\overline{\mathcal{C}}_{tt}$ and $\overline{\mathcal{C}}_{ts}$ reveals this. With

$$\overline{\mathcal{C}}_t = \overline{\beta}\mathcal{N}$$

it follows

$$\begin{aligned} \overline{\mathcal{C}}_t &= (\beta(\phi(q,t),t)\mathcal{N})_t \\ &= (\beta_p\phi_t + \beta_t)\mathcal{N} - \beta\beta_p\phi_s\mathcal{T} \end{aligned}$$

and

$$\begin{aligned} \overline{\mathcal{C}}_{ts} &= (\beta(\phi(q,t),t)\mathcal{N})_s \\ &= (\beta_p\phi_s)\mathcal{N} - \beta\kappa\mathcal{T}. \end{aligned}$$

With

$$\begin{aligned} \mathcal{T}\cdot\overline{\mathcal{C}}_{ts} &= -\beta\kappa, \\ \overline{\mathcal{C}_t}\cdot\overline{\mathcal{C}_{ts}} &= \beta\beta_p\phi_s, \\ \|\mathcal{C}_t\|^2 &= \beta^2 \end{aligned}$$

it follows that (through substitution of the respective terms in Equation (27))

$$\mu\left((\beta_p\phi_t+\beta_t)\mathcal{N}-\beta\beta_p\phi_s\mathcal{T}\right)=\mu\beta^2\kappa\mathcal{N}-\mu\beta\beta_p\phi_s\mathcal{T}-\left(\frac{1}{2}\mu\beta^2-g\right)\kappa\mathcal{N}-(\nabla g\cdot\mathcal{N})\mathcal{N}.$$

This simplifies to

$$\mu(\beta_p\phi_t+\beta_t)=\left(\frac{1}{2}\mu\beta^2+g\right)\kappa-(\nabla g\cdot\mathcal{N})\mathcal{N}, \tag{33}$$

which depends on the time derivative of the reparameterization function ϕ which in turn depends on the tangential component α. The left hand side of Equation (33) represents a transport term along the curve, the speed of which depends on the time derivative of the reparameterization function ϕ.

If the Lagrangian used to formulate the variational problem does not explicitly depend on time, the overall energy of the system will stay constant for all times. In this context this amounts to

$$\int_0^1\left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2+g\right)\|\mathcal{C}_p\|\;dp=\text{const.}$$

3.5.1 Special Solutions

Studying a simple circular example illustrates the behavior of Equations (30) and (31). Assume $g=\mu=1$. Then $\nabla g=0$. Furthermore, assume a circle evolving with radius R and constant initial velocities

$$\alpha(s,0)=\alpha_0\qquad\beta(s,0)=\beta_0.$$

Then the normal evolution reduces to

$$\begin{aligned}\beta_t &= (\frac{1}{2}\beta^2+1)\frac{1}{R}-\gamma_\beta\beta\\ R_t &= -\beta\end{aligned} \tag{34}$$

and the full evolution becomes

$$\begin{aligned}\alpha_t &= 2\alpha\beta\frac{1}{R}-\gamma_\alpha\alpha\\ \beta_t &= \left[\left(\frac{1}{2}\beta^2-\frac{3}{2}\alpha^2\right)+1\right]\frac{1}{R}-\gamma_\beta\beta\\ R_t &= -\beta,\end{aligned} \tag{35}$$

making use of the relations

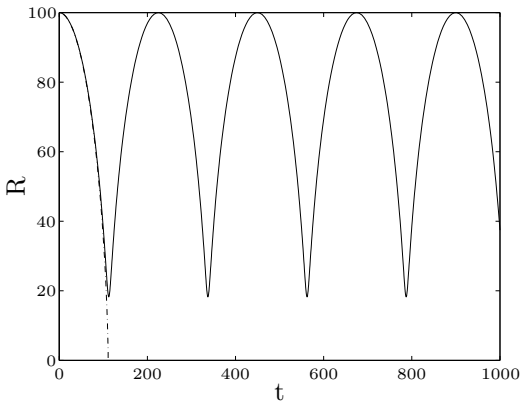
$$\begin{aligned} \alpha_s &= \beta_s = 0 \quad \forall t \qquad \text{(given the constant initial conditions for the circle)} \\ \kappa &= \frac{1}{R}, \end{aligned}$$

and adding an artificial friction term, with γ_α and γ_β being the friction coefficients for the tangential and the normal velocity, respectively. Since the initial velocity conditions are constant on the circle and the circle evolves on a uniform potential field g, the solution is rotationally invariant (with respect to the origin of the circle). Thus it is sensible to evolve R in Equation (35) by using its normal velocity only.

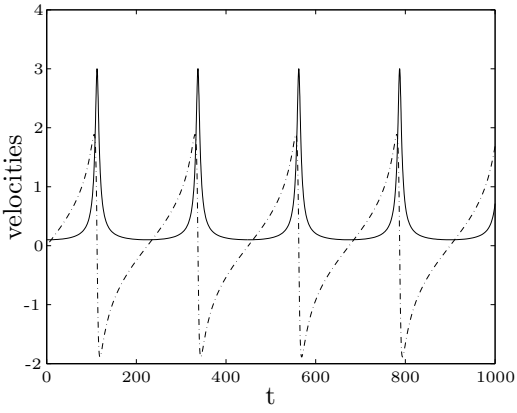
Figures 6(a) to 8(c) show the evolution of the radius, R, the tangential velocity α (if applicable), the normal velocity β for a small initial value of α, a larger initial value of α, and with added friction, respectively.

Figures 6(a), 6(b), and 6(c) show the results for $\alpha_0 = 0.1$, $\beta_0 = 0$, $R_0 = 100$, $\gamma_\alpha = 0$, $\gamma_\beta = 0$. While in the normal evolution case the circle accelerates rapidly and disappears in finite time, this is not the case when tangential velocities are not neglected: then the circle oscillates. It rotates faster if it becomes smaller and slower if it becomes larger. Due to the small initial tangential velocity the radius evolution is initially similar in both cases. The oscillation effect is more drastic with increased initial tangential velocity ($\alpha_0 = 1$). This can be seen in Figures 7(a) and 7(b). Figures 8(a) to 8(c) show the results with added friction ($\gamma_\alpha = \gamma_\beta = 0.1$). Both circles disappear in finite time. The evolutions of the radius look similar in both cases. Due to the large friction coefficients a large amount of energy gets dissipated; oscillations no longer occur.

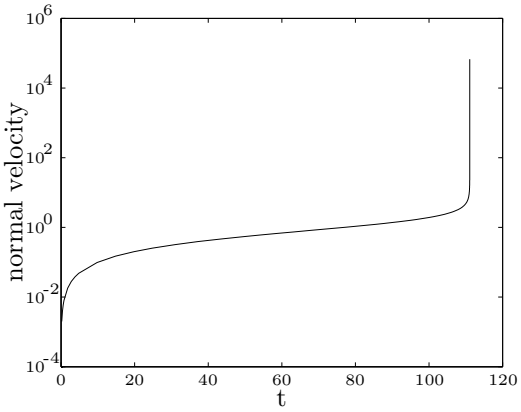
Equations (34) and (35) do not exhibit the same behavior. Depending on the initial value for α, they will have fundamentally different solutions. For $\alpha = \pm\sqrt{\frac{2}{3}}$, and $\beta_0 = 0$ in Equation (35), the solution is (geometrically) stationary, and the circle will keep its shape and rotate with velocity α for all times. Also if $\alpha_0 = 0$, in this example case, both evolutions will be identical.



(a) Evolution of the radius for the normal velocity evolution (dashed line) and the full velocity evolution (solid line).

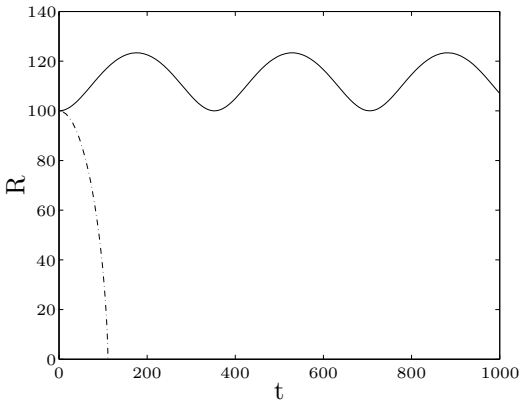


(b) Evolution of normal velocity (dashed line) and the tangential velocity evolution (solid line) for the full velocity approach.

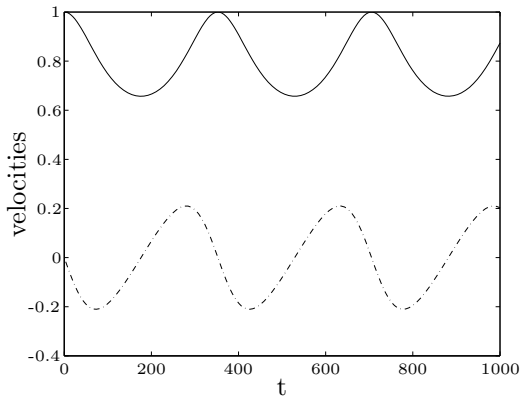


(c) Evolution of normal velocity for the normal velocity evolution.

Figure 6: Geometric dynamic curve evolution for an oscillating circle with parameters: $\alpha_0 = 0.1$, $\beta_0 = 0$, $R_0 = 100$, $\gamma_\alpha = 0$, $\gamma_\beta = 0$.

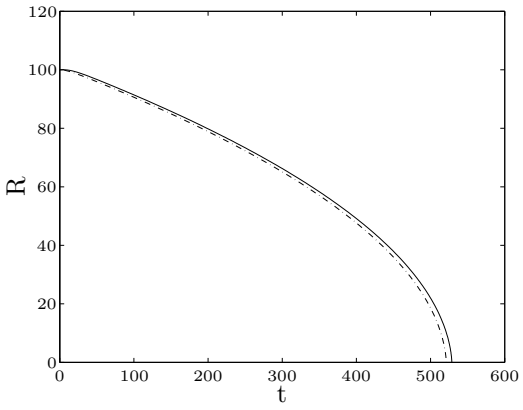


(a) Evolution of the radius for the normal velocity evolution (dashed line) and the full velocity evolution (solid line).

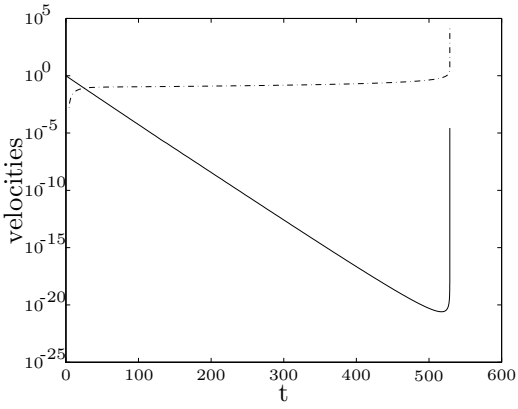


(b) Evolution of normal velocity (dashed line) and the tangential velocity evolution (solid line) for the full velocity approach.

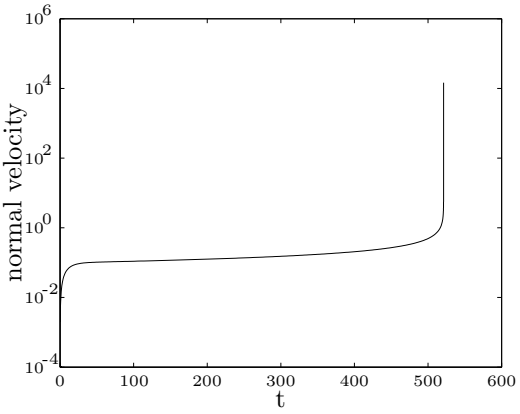
Figure 7: Geometric dynamic curve evolution for an oscillating circle with parameters: $\alpha_0 = 1$, $\beta_0 = 0$, $R_0 = 100$, $\gamma_\alpha = 0$, $\gamma_\beta = 0$.



(a) Evolution of the radius for the normal velocity evolution (dashed line) and the full velocity evolution (solid line).



(b) Evolution of normal velocity (dashed line) and the tangential velocity evolution (solid line) for the full velocity approach.



(c) Evolution of normal velocity for the normal velocity evolution.

Figure 8: Geometric dynamic curve evolution for an oscillating circle with parameters: $\alpha_0 = 1$, $\beta_0 = 0$, $R_0 = 100$, $\gamma_\alpha = 0.1$, $\gamma_\beta = 0.1$.

CHAPTER 4

EXTENSIONS

This chapter discusses possible extensions to the framework developed in Chapter 3. Section 4.1 develops an observer-like evolution equation to improve the performance of the dynamic curve evolutions on real image sequences. Section 4.2 derives a simple methodology for occlusion detection and integrates it into the evolution equations derived in Section 4.1.

4.1 Observers

A system governed by a time-independent Lagrangian (i.e., $L_t \equiv 0$) will preserve energy [132], but this is not necessarily desirable. Indeed, envision a curve evolving on a static image with an initial condition of zero normal velocity everywhere and with an initial position of nonminimal potential energy. The curve will oscillate in its potential well indefinitely. One solution to this problem is to dissipate energy [129], which can be accomplished by simply adding a friction term to Equation (61). However, to increase robustness it is desirable to be able to dissipate and to add energy to the system in a directed way. Specifically, it is desirable to add and dissipate energy in such a way that the evolving curve approximates the boundary contour of the object to be tracked.

An observer is a dynamical system designed to dynamically approximate the state of another system based on measurable outputs. Usually not all the states of a given system are measurable (be it for cost reasons, inaccessibility, etc.). Then an observer reconstructs a system's states based on the available outputs. The problem in this thesis is slightly different. All the states of the system (here, location and velocity) can be measured; this is the fully observable case. The problem then becomes one of blending an incorrect, approximate system state into the correct one. The straightforward approach to do this would simply be to replace the current state of a particle by it's measured "correct" state. However, this is not desirable, since it would destroy the dynamical properties of the system

and would be very sensitive to noise. Constructing an observer for the geometric dynamic active contour is not straightforward, since the governing equations are infinite dimensional and nonlinear (see Sections 2.2 and 3.5). Observers for certain classes of infinite dimensional systems exist. In particular, for systems with nonlinearities free of differential operators (see [141] for a design procedure for such observers). The theory for more general systems is in its infancy; e.g., Vande Wouwer and Zeitz [141] describe a minimum least squares estimator for general nonlinear second order infinite dimensional systems. However, the resulting filter is complex (and its complexity would increase for an increasing number of space dimensions considered). A less principled way for the curve evolution to approximate the dynamic behavior of the tracked objects is to use error injection. This guarantees convergence of the curve to the desired object(s) if the curve is initially in the appropriate basin of attraction and is the basis for the construction of an observer. However, in the case presented here there, is no formal stability and convergence proof.

Estimating the position and velocity vector for every point on the curve $\mathcal{C}$ allows for error injection. Define the line through the point $\boldsymbol{x}(s)$ on the current curve as

$$l(s,p) := \boldsymbol{x}(s) - p\mathcal{N}$$

and the set of points in an interval (a,b) on the line as

$$L(a,b,s) := \{l(s,p), a < p < b)\}.$$

Define

$$\begin{aligned} f(s) &:= \inf\{p : p < 0, \Phi(\boldsymbol{x}) \leq 0 \ \forall \boldsymbol{x} \in L(p,0,s)\}, \\ t(s) &:= \sup\{p : p > 0, \Phi(\boldsymbol{x}) \geq 0 \ \forall \boldsymbol{x} \in L(0,p,s)\}. \end{aligned}$$

The set of estimated contour point candidates Z is the set of potential edge points in $L(f,t,s)$

$$Z(L(f,t,s)) := \{\boldsymbol{x} : \boldsymbol{x} \in L(f,t,s), \exists \epsilon > 0 : \|\nabla(G * I(\boldsymbol{x}))\| > \|\nabla(G * I(\boldsymbol{y}))\| \\ \forall \boldsymbol{y} \in L(f,t,s) \cap B_\epsilon(\boldsymbol{x}), \boldsymbol{y} \neq \boldsymbol{x}\},$$

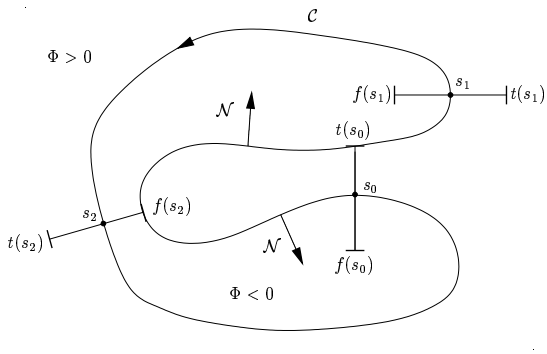


Figure 9: The feature search is performed in the normal direction to the curve. The search region is only allowed to intersect the curve at its origin of search (i.e., s_0, s_1, s_2, ...).

where G is a Gaussian, $B_\epsilon(\boldsymbol{x})$ is the disk around $\boldsymbol{x}$ with radius ϵ, and I is the current image intensity. Many other feature detectors are conceivable. Given some problem specific likelihood function $m(\boldsymbol{z})$ the selected contour point is the likelihood maximum

$$\boldsymbol{x}_c(s) = \underset{\boldsymbol{z} \in Z(L(f,t,s))}{\arg\max} \; m(\boldsymbol{z}),$$

at position

$$p_c = d(\boldsymbol{x}, s) = (\boldsymbol{x}(s) - \boldsymbol{x}_c(s))^T \mathcal{N}.$$

It is sufficient to estimate normal velocity, since the curve evolution equation does not take tangential velocity components into account. The estimation then can be performed (assuming brightness constancy from image frame to image frame for a moving image point) by means of the optical flow constraint *without* the need for regularization. Note, that this estimate is computed on a few chosen points in Z only. The optical flow constraint is given as

$$I_t + uI_x + vI_y = 0,$$

where $u = x_t$ and $v = y_t$ are the velocities in the x and the y direction respectively. Restricting the velocities to the normal direction by setting

$$\begin{pmatrix} u \\ v \end{pmatrix} = \gamma \frac{\nabla I}{\|\nabla I\|}.$$

yields

$$\gamma = -\frac{I_t}{\|\nabla I\|}$$

and thus the desired velocity estimate

$$\begin{pmatrix} u \\ v \end{pmatrix} = -I_t \frac{\nabla I}{\|\nabla I\|^2}.$$

Define

$$\begin{aligned} \overline{\beta} &:= -\gamma \frac{\nabla I}{\|\nabla I\|} \cdot \frac{\nabla \hat{\Phi}}{\|\nabla \hat{\Phi}\|}, \\ \overline{\Phi} &:= -\|\boldsymbol{x}_c - \boldsymbol{x}\| \text{sign}\left(\hat{\Phi}(\boldsymbol{x}_c)\right). \end{aligned}$$

The following observer-like dynamical system

$$\begin{aligned} \hat{\Phi}_t &= \left(m(\boldsymbol{x}_c) K_\Phi (\overline{\Phi} - \hat{\Phi}) + \hat{\beta} + \gamma\kappa\right) \|\nabla \hat{\Phi}\|, \\ \hat{\beta}_t &= m(\boldsymbol{x}_c) K_\beta (\overline{\beta} - \hat{\beta}) + \left(\frac{1}{2}\hat{\beta}^2 + \frac{g}{\mu}\right)\kappa + \frac{1}{\mu}\nabla g \cdot \frac{\nabla \hat{\Phi}}{\|\nabla \hat{\Phi}\|} + \delta \hat{\beta}_{ss}, \end{aligned} \tag{36}$$

dynamically blends the current curve $\hat{\mathcal{C}}$ into the desired curve $\mathcal{C}$ (see Figure 10). Here, K_Φ and K_β are the error injection gains for $\hat{\Phi}$ and $\hat{\beta}$, respectively. Any terms related to image features are computed at the current location $\boldsymbol{x}$ of the contour. The error injection gains are weighted by the likelihood $m(\boldsymbol{x}_c)$ of the correspondence points as a measure of prediction quality. The additional terms $\kappa\gamma$ and $\delta\hat{\beta}_{ss}$ with tunable weighting factors γ and δ are introduced to allow for curve and velocity regularization if necessary. They are computed as

$$\kappa = \nabla \cdot \left(\frac{\nabla \hat{\Phi}}{\|\nabla \hat{\Phi}\|}\right)$$

and

$$\hat{\beta}_{ss} = \mathcal{N}^T \begin{pmatrix} \hat{\beta}_{yy} & -\hat{\beta}_{xy} \\ -\hat{\beta}_{xy} & \hat{\beta}_{xx} \end{pmatrix} \mathcal{N} + \kappa \nabla \hat{\beta} \cdot \mathcal{N}.$$

In case no correspondence point for a point on the zero level set of $\hat{\Phi}$ is found, the evolution equation system (36) is replaced by

$$\begin{aligned} \hat{\Phi}_t &= (\hat{\beta} + \gamma\kappa)\|\nabla \hat{\Phi}\|, \\ \hat{\beta}_t &= \delta \hat{\beta}_{ss} \end{aligned} \tag{37}$$

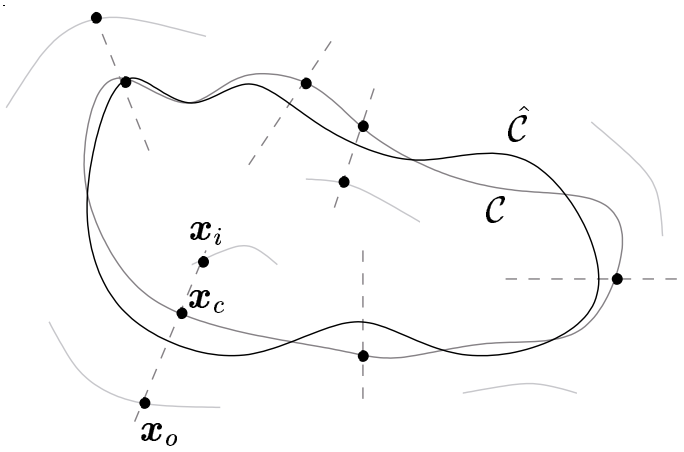


Figure 10: Correspondence point $\boldsymbol{x}_c$, inside correspondence point $\boldsymbol{x}_i$, and outside correspondence point $\boldsymbol{x}_o$ of the curve $\hat{\mathcal{C}}$. $\mathcal{C}$ represents the contour of the object to be tracked.

for this point. Assuming there is no curve and velocity regularization (i.e., $\gamma = \delta = 0$), a straight curve part ($\kappa = 0$) will have the dynamic steady state ($\beta_t = 0$)

$$\hat{\beta} = \frac{1}{m(\boldsymbol{x}_c)} \left(\overline{\beta} + \frac{1}{\mu K_\beta} \nabla g \cdot \frac{\nabla \hat{\Phi}}{\|\nabla \hat{\Phi}\|} \right).$$

For large values of K_β and trustworthy feature points $\boldsymbol{x}_c$ (with $m(\boldsymbol{x}_c) \approx 1$), $\hat{\beta}$ will approach $\overline{\beta}$ as desired.

4.2 Occlusion Detection

The following simple occlusion detection algorithm allows for the assessment of the curve's prediction capability[1]. It is based on ideas in [60]. The inside and the outside correspondence points are defined as (see Figure 10)

$$\begin{aligned} \boldsymbol{x}_i(s) &= \underset{\boldsymbol{z} \in Z(L(f, p_c, s))}{\arg\max} \; m(\boldsymbol{z}), \\ \boldsymbol{x}_o(s) &= \underset{\boldsymbol{z} \in Z(L(p_c, t, s))}{\arg\max} \; m(\boldsymbol{z}). \end{aligned}$$

[1]More sophisticated, and less parametric, occlusion detection algorithms are conceivable; however, this is not the main focus of this thesis, and the algorithm proposed is sufficient to show that the geometric dynamic active contour can handle occlusions when combined with a suitable occlusion detection algorithm.

The occlusion detection strategy is split into the following six subcases for every point on the contour

(0) There is no correspondence point.

(1) Only the correspondence point is present.

(2) The point is moving outward, the correspondence point is present, but not its outside correspondence point.

(3) The point is moving inward, the correspondence point is present, but not its inside correspondence point.

(4) The point is moving outward, both the correspondence point and its outside correspondence point are present.

(5) The point is moving inward, both the correspondence point and its inside correspondence point are present.

Define the following Gaussian conditional probabilities

$$\begin{aligned} Pr(t_{occ}|occ) &= \frac{2}{\sqrt{2\pi}\sigma_t} e^{-\frac{(t_{occ}-\mu_t)^2}{2\sigma_t^2}} \\ Pr(v_a|occ) &= \frac{1}{\sqrt{2\pi}\sigma_v} e^{-\frac{(v_a-\mu_v)^2}{2\sigma_v^2}} \\ Pr(t_{occ}|\overline{occ}) &= \frac{2}{\sqrt{2\pi}}\sigma_{\bar{t}} e^{-\frac{(t_{occ}-\mu_{\bar{t}})^2}{2\sigma_{\bar{t}}^2}} \\ Pr(v_a|\overline{occ}) &= \frac{1}{\sqrt{2\pi}\sigma_{\overline{v}}} e^{-\frac{(v_a-\mu_{\overline{v}})^2}{2\sigma_{\overline{v}}^2}}, \end{aligned}$$

where t_{occ} is the estimated time to occlusion, v_a is the velocity of the point ahead, overlined symbols denote negated values (i.e., $\overline{occ}$ means not occluded), $Pr(t_{occ}|occ)$, $Pr(v_a|occ)$ are the probabilities of t_{occ} and v_a given an occlusion, and $Pr(t_{occ}|\overline{occ})$ and $Pr(v_a|\overline{occ})$ given there is no occlusion respectively. The corresponding standard deviations are σ_t, σ_v, $\sigma_{\bar{t}}$, and $\sigma_{\overline{v}}$; the means are μ_t, μ_v, $\mu_{\bar{t}}$, $\mu_{\overline{v}}$. Using the currently detected correspondence point $\boldsymbol{x}_c$, and its interior $\boldsymbol{x}_i$ and exterior $\boldsymbol{x}_o$ correspondence points the values of t_{occ} and v_a are computed.

The probability for an occlusion is given by Bayes' formula as

$$Pr(occ|v_a, t_{occ}) = \frac{Pr(v_a, t_{occ}|occ)Pr(occ)}{Pr(v_a, t_{occ}|occ)Pr(occ) + Pr(v_a, t_{occ}|\overline{occ})Pr(\overline{occ})}.$$

where $Pr(occ) = 0$ and $Pr(\overline{occ}) = 1$ everywhere initially. The priors at time step $n+1$ are the smoothed posteriors of time step n. In case (0) $Pr(occ|v_a, t_{occ}) = Pr(occ)$ (i.e., the probability is left unchanged), in all other cases

$$Pr(occ|v_a, t_{occ}) = \frac{Pr^i_{occ}Pr(occ)}{Pr^i_{occ}Pr(occ) + Pr^i_{\overline{occ}}Pr(\overline{occ})},$$

where

$$\begin{aligned} Pr^1_{occ} &= Pr(v_a = v_c|occ), \\ Pr^1_{\overline{occ}} &= Pr(v_a = v_c|\overline{occ}), \end{aligned}$$

$$Pr^2_{occ} = \begin{cases} Pr(v_a = v_c|occ) & \text{if } \boldsymbol{x}_c \text{ outside of } \mathcal{C}, \\ 0 & \text{otherwise,} \end{cases}$$

$$Pr^2_{\overline{occ}} = \begin{cases} Pr(v_a = v_c|\overline{occ}) & \text{if } \boldsymbol{x}_c \text{ outside of } \mathcal{C}, \\ 0 & \text{otherwise,} \end{cases}$$

$$Pr^3_{occ} = \begin{cases} Pr(v_a = v_c|occ) & \text{if } \boldsymbol{x}_c \text{ inside of } \mathcal{C}, \\ 0 & \text{otherwise,} \end{cases}$$

$$Pr^3_{\overline{occ}} = \begin{cases} Pr(v_a = v_c|\overline{occ}) & \text{if } \boldsymbol{x}_c \text{ inside of } \mathcal{C}, \\ 0 & \text{otherwise,} \end{cases}$$

$$\begin{aligned} Pr^4_{occ} &= Pr(v_a = v_o|occ)Pr(t_{occ} = t^o_{occ}|occ), \\ Pr^4_{\overline{occ}} &= Pr(v_a = v_o|\overline{occ})Pr(t_{occ} = t^o_{occ}|\overline{occ}), \end{aligned}$$

$$\begin{aligned} Pr^5_{occ} &= Pr(v_a = v_i|occ)Pr(t_{occ} = t^i_{occ}|occ), \\ Pr^5_{\overline{occ}} &= Pr(v_a = v_i|\overline{occ})Pr(t_{occ} = t^i_{occ}|\overline{occ}), \end{aligned}$$

and

$$
\begin{aligned}
v_c &= \overline{\beta}(\boldsymbol{x}_c) & v_o &= \overline{\beta}(\boldsymbol{x}_o) \\
v_i &= \overline{\beta}(\boldsymbol{x}_i) & v &= \overline{\beta}(\boldsymbol{x}) \\
t^i_{occ} &= \frac{\|\boldsymbol{x}-\boldsymbol{x}_i\|}{|v-v_i|} & t^o_{occ} &= \frac{\|\boldsymbol{x}-\boldsymbol{x}_o\|}{|v-v_o|}.
\end{aligned}
$$

To estimate the current rigid body motion, the equation system

$$
\begin{aligned}
(u_r\ ,\ v_r)^T \int_{\mathcal{C}} n^1 \mathcal{N}\ ds &= -\int_{\mathcal{C}} n^1 \beta\ ds \\
(u_r\ ,\ v_r)^T \int_{\mathcal{C}} n^2 \mathcal{N}\ ds &= -\int_{\mathcal{C}} n^2 \beta\ ds,
\end{aligned}
$$

is solved, where $\mathcal{N} = (n^1\ ,\ n^2)^T$ and $\mu_{\overline{v}} = -(u_r\ ,\ v_r)^T\mathcal{N}$ and $\mu_v = 0$.

The evolution equation becomes

$$
\begin{aligned}
\hat{\Phi}_t &= \left(Pr(\overline{occ})\left(m(\boldsymbol{x}_c)K_\Phi(\overline{\Phi}-\hat{\Phi})\right) + \hat{\beta} + \gamma\kappa\right)\|\nabla\hat{\Phi}\| \\
\hat{\beta}_t &= Pr(\overline{occ})\left(m(\boldsymbol{x}_c)K_\beta(\overline{\beta}-\hat{\beta}) + \left(\frac{1}{2}\hat{\beta}^2 + \frac{g}{\mu}\right)\kappa\right) + \\
&\quad Pr(\overline{occ})\frac{1}{\mu}\nabla g\kappa + \delta\beta_{ss}.
\end{aligned}
$$

This is a linear interpolation between the systems (36) and (37) based on the occlusion probability.

CHAPTER 5

LEVEL SET APPROACHES

To implement the derived curve evolution equations, we propose two different approaches: the partial level set method and the full level set method. The full level set approach propagates the curve in a space consistent with the inherent dimensionality of the problem; there is no separation between the representation of curves in the image plane and the state information propagated along with the curves: the complete state is represented implicitly. Geometric dynamic curve evolution would thus be performed in $\mathbb{R}^4$ (codimension three) in the simplest case (since we are looking at planar curves). The codimensionality will increase if additional information is to be attached to the curve (e.g., in the PI controller case). Normal geometric dynamic curve evolution is at least a problem in $\mathbb{R}^3$ (codimension two) in this setting. This method is computationally relatively complex, but allows for full topological flexibility.

The second methodology, the partial level set approach, is based on an implicit description of the geometrical shape of the curves in the image plane by means of a level set method. Additional state information (e.g., the normal speed in the normal geometric dynamic curve evolution setting) gets propagated explicitly with every point on the curves by possibly multiple transport equations and is not included in the level set representation. This method has the advantage of computational efficiency, since the level set evolution is performed in a low dimensional space, but sacrifices object separation: tracked objects that collide will be merged solely on their position information; intersecting curves cannot be represented.

Section 5.1 reviews briefly the classical level set method [115] designed to deal with closed curves or surfaces of codimension one. This will be the level set method of choice for the partial level set approach of Section 5.3. Section 5.2 will deal with level set methods designed for the evolution of objects of codimension larger than one. We will focus on

and extend the vector distance function based level set approach used for the full level set method in Section 5.4. Section 5.3 describes the partial level set approach. Finally, Section 5.4 explores the full level set approach.

5.1 The Classical Level Set Approach

The object $\mathcal{M}$ to be represented (e.g., a closed curve $\mathcal{C}$ or a closed surface $\mathcal{S}$) is given implicitly as the zero level set of the function

$$\Phi(\boldsymbol{x}(t), t) : \mathbb{R}^d \times \mathbb{R}^+ \mapsto \mathbb{R}, \tag{38}$$

where Φ is called the level set function. Typically, Φ is initialized as the signed distance function to $\mathcal{M}$ (assuming $\Phi > 0$ outside of $\mathcal{M}$ and $\Phi < 0$ inside $\mathcal{M}$). Taking the total derivative of equation (38) results in

$$\Phi_t + \nabla\Phi \cdot \boldsymbol{x}_t = 0. \tag{39}$$

This is a d-dimensional transport equation, with $\boldsymbol{x}_t$ being the speed on $\mathcal{M}$. If $\boldsymbol{x}_t$ for each point on $\mathcal{M}$ is known, the solution of Equation (39) guarantees, that the movement of the zero level set of Φ corresponds to the desired movement of $\mathcal{M}$. For the evolution of the level set function Φ the velocity vector $\boldsymbol{x}_t$ has to be defined on the complete domain. Since Equation (17) only gives the velocities on $\mathcal{M}$ itself, extension velocities have to be constructed [115] if it is not clear how to specify $\boldsymbol{x}_t$ on all of $\mathbb{R}^d$. Generally curves are only moved in their normal directions (i.e., $\nabla\Phi \cdot \boldsymbol{x}_t$ gets replaced by $\|\nabla\Phi\|\mathcal{N} \cdot \boldsymbol{x}_t$), since the tangential component of the evolution law only changes the parameterization of $\mathcal{M}$, but not its geometrical shape (see [113] for a proof). The position of $\mathcal{M}$ will be described for all times by

$$\Phi_t + \|\nabla\Phi\|\mathcal{N} \cdot \boldsymbol{x}_t = 0.$$

5.2 Level Set Approaches for Higher Codimensions

Typically, the level set formulation is used to represent closed codimension one objects (a closed curve in $\mathbb{R}^2$, a closed surface in $\mathbb{R}^3$, etc., see Section 5.1). A surface that is not closed will not separate its embedding space into two distinct regions. Also, there is the possibility

to represent objects which are not orientable. Imagine a Möbius strip in $\mathbb{R}^3$. This is a codimension one object, however it is neither closed nor allows for an orientation.

Recently there has been an increased interest in evolving objects of codimension larger than one, with applications ranging from geometric optics [20, 96, 103], to image processing [86], and to geometric optimization [109]. Theoretical aspects of these evolutions have been investigated in [6, 43, 44, 45, 46, 57, 120].

The usual level set approach [105, 115] 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 [6] 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.* [86] 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 [105], since curves can develop interiors. (See [20] for an example of fattening and [9] 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 (e.g., one could use k signed distance functions with the intersection of their respective zero level sets representing the desired manifold) on the surface of the 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 [31] is not available for such systems of equations). Nevertheless, this approach has been very successfully applied, in particular to problems in geometric optics [20, 96, 103], where the handling of topological changes is of no importance (as they do not occur in this setting [103]). 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) [20].

An alternative approach, closely related to the evolution of an unsigned distance function, is the evolution of a vector distance function, which is the approach focused on in this thesis. 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 [55]. In this thesis, 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).

Section 5.2.1 describes the main level set approaches, focusing on the vector distance function based approach. Section 5.2.2 discusses the vector distance function based level set

method. Section 5.4.1 applies the methodology developed in Section 5.2.2 to the evolution of a normal geometric dynamic active contour, which will constitute the full level set approach.

5.2.1 Overview of Level Set Methods for Higher Codimensions

This section outlines some of the main methods for the evolution of manifolds of codimension greater than one via various level set approaches. It then describes in some detail the vector distance function method which will subsequently be applied 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 [105]. Ambrosio and Soner extend the theory of level set evolutions to mean curvature flows in arbitrary codimensions in their seminal paper [6]. 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.* [86]) 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 $v : \mathbb{R}^d \mapsto \mathbb{R}^+$

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

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

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

where

$$P_{\boldsymbol{p}} = I - \frac{\boldsymbol{p}\boldsymbol{p}^T}{\|\boldsymbol{p}\|^2}, \ \boldsymbol{p} \neq \mathbf{0},$$

is the orthogonal projection operator along $\boldsymbol{p}$ and ∇^2 denotes the Hessian. The $d - k$ smallest eigenvalues of $J(\boldsymbol{x})$ orthogonal to $\nabla v(\boldsymbol{x})$ are related to the geometry of $\mathcal{M}$.

According to [6], the evolution

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

where

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

and

$$\lambda_1(P_{\boldsymbol{p}} A P_{\boldsymbol{p}}) \leq \lambda_2(P_{\boldsymbol{p}} A P_{\boldsymbol{p}}) \leq \cdots \leq \lambda_{d-1}(P_{\boldsymbol{p}} A P_{\boldsymbol{p}})$$

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

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

The uniqueness and existence results obtained in [6] can be extended to a general normal velocity

$$V = H + \Pi^N_{\mathcal{M}} \boldsymbol{g}(\boldsymbol{x}, t),$$

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

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

This approach is mathematically well founded, however, fattening of the evolving manifold can occur [9, 20] 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 (see [20, 104, 11] for the case of codimension two, [103] for the case of codimension three, and [56] for the arbitrary codimension case) which is the second method proposed by Ambrosio and Soner [6]. Given the k scalar functions $\alpha^i : \mathbb{R}^d \mapsto \mathbb{R}$, $0 < i \leq k,\ i \in \mathbb{N}$, the evolving object is implicitly described by the simultaneous evolution of

$$\alpha^i_t + \boldsymbol{x}_t \cdot \nabla \alpha^i = 0,$$

where $\boldsymbol{x}_t$ 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 (in general) performed globally (over the whole computational domain) if well-behaved handling of topological changes is required (narrow-banding approaches [2] 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 [55, 57]. Given a manifold $\mathcal{M}$,

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

is defined as the distance from point $\boldsymbol{x} \in \mathbb{R}^d$ to the manifold $\mathcal{M}$. The vector distance function $\boldsymbol{u}(\boldsymbol{x})$ is then given as the derivative of the squared distance function (see [6, 109] for details on the squared distance function)

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

Thus

$$\boldsymbol{u}(\boldsymbol{x}) = \nabla\eta(\boldsymbol{x}) = \delta(\boldsymbol{x})\nabla\delta(\boldsymbol{x}).$$

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

$$\mathcal{M} = \boldsymbol{u}^{-1}(\boldsymbol{0}).$$

This amounts to the intersection of the d hypersurfaces

$$\boldsymbol{u}_i = 0, 1 \leq i \leq d,\ i \in \mathbb{N}.$$

The description is redundant, but unique.

The evolution equation for the manifold $\mathcal{M}(p,t)$, parameterized by p, becomes (using the notation of [55])

$$\mathcal{M}_t(p,t) = \Pi^N_{\mathcal{M}(p,t)}(\mathcal{D}(\mathcal{M}(p,t),t)) = \boldsymbol{V}(\mathcal{M}(p,t),t),$$

where $\mathcal{D}(\boldsymbol{x},t) = \boldsymbol{x}_t$ is a vector field defined on $\mathbb{R}^d \times \mathbb{R}^+$, determining the evolution speed at $\boldsymbol{x}$. Here, $\Pi^N_{\mathcal{M}(p,t)}$ is the operator projecting the velocity $\boldsymbol{x}_t$ on $\mathcal{M}$ into the normal space of $\mathcal{M}$. 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 has to be constructed in the subspace of $\mathbb{R}^d$ that contains the evolution of the manifold (in Section 5.4.1 the domain will be defined based on image dimensions, and the expected velocities). This speed function should

1) maintain the vector distance function throughout the evolution, and

2) move the manifold $\mathcal{M}$ as desired.

It can be shown (see [57]) that the characteristic equation for the vector distance function $\boldsymbol{u}(\boldsymbol{x})$ is

$$(D\boldsymbol{u})^T\boldsymbol{u} = \boldsymbol{u}, \tag{40}$$

where $D\boldsymbol{u}$ denotes the Jacobian of $\boldsymbol{u}$. Taking the time derivative of Equation (40) and using the fact that $(D\boldsymbol{u})^T = D\boldsymbol{u}$ yields (see [57] for a derivation)

$$D\boldsymbol{b}\boldsymbol{u} = (\boldsymbol{I} - D\boldsymbol{u})\boldsymbol{b}, \tag{41}$$

where $\boldsymbol{b}$ is the desired velocity for the vector distance function evolution with initial condition

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

The overall evolution is then given by

$$\boldsymbol{u}_t + (D\boldsymbol{u})^T\boldsymbol{b} = \boldsymbol{0}.$$

The vector distance function to a given manifold $\mathcal{M}$ is unique, and its initialization is straightforward. Furthermore, narrow-band implementations are feasible, which significantly reduce the computational cost. A fattening phenomenon (similar in spirit to the one observed in the approach of [6]) can be observed for vector distance function based level set approaches. This is not surprising due to the intimate connection between distance functions and vector distance functions.

In what follows only closed one-dimensional curves in $\mathbb{R}^d$ are considered, 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 is a sufficiently good representation for the evolutions based on the equations of dynamic active contours. The full level set approach for dynamic active contours will be presented in Section 5.4.1.

5.2.2 Details on Vector Distance Function Evolutions

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 [55]. Objects can also change dimension throughout the vector distance function evolution. This is in clear 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), i.e., a closed curve, a closed surface, etc. 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.

The following steps constitute the proposed vector distance function based curve evolution algorithm. 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., $\boldsymbol{u}^{-1}(\mathbf{0}) = \mathcal{M}$).

(2) Redistance the vector distance function outside of the zero band.

(3) Compute $\boldsymbol{b}$ on the band determined in (1).

(4) Extend the $\boldsymbol{b}$ vector 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.

5.2.2.1 Initializing the Vector Distance Function

Assume a given 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. Then working on a discrete grid, we explicitly initialize[1] the vector distance function $\boldsymbol{u}$ on a set, $\mathcal{B}$, γ-close to the piecewise linear approximation $\mathcal{A}$:

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

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.* [125]. Since there is no inside and outside in the case considered here, it is not possible to use this approach to initialize over the whole domain Ω (see Section 5.2.2.3 for more details).

5.2.2.2 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 will be the extension of quantities (like the velocities). However, frequently it is sufficient to

[1] For an efficient implementation this set has to be approximated, since the distance of a point $\boldsymbol{x}$ to $\mathcal{A}$ is unknown beforehand.

perform these operations away from the zero level set [42], 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 and do not compute extension velocities on the set $\hat{\mathcal{B}}$ which is γ-close to the zero level set (in analogy to Section 5.2.2.1). Since we have the current vector distance function $\boldsymbol{u}$ at our disposal, determining this set is straightforward

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

Section 5.2.2.7 shows that a more precise approximation for the zero level set is needed for the explicit reinitialization of the vector distance function field on $\hat{\mathcal{B}}$. The value for γ needs to be chosen conservatively to make sure that the real zero level set is contained in $\hat{\mathcal{B}}$.

5.2.2.3 Redistancing the Vector Distance Field Away from the Zero Band

Inspired by [41], the vector distance function is reinitialized only outside the zero band obtained as in Section 5.2.2.2. Gomes and Faugeras [55] propose to minimize the functional

$$\frac{1}{2}\int_{\Omega} \|D\boldsymbol{u}^T\boldsymbol{u} - \boldsymbol{u}\|^2 \, d\boldsymbol{x}. \tag{42}$$

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

Instead of using this flow, we thus decide to perform an alternative redistancing by the intermediate step of computing a distance function. Based on this reinitialized distance field, the vector distance field is computed. This has the advantage that standard algorithms can be used and iteration free solutions (e.g., by using a fast marching approach) are possible[2]. The current distance function $\delta(\boldsymbol{x})$ relates to the vector distance function by

$$\delta(\boldsymbol{x}) = \|\boldsymbol{u}(\boldsymbol{x})\|. \tag{43}$$

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

The distance function can be reinitialized on $\hat{\mathcal{B}}^c$ for example by a fast marching approach [115] or by a partial differential equation approach along the lines of Sussman *et al.* [125]. However, since we are not dealing with a signed distance function here, the approach by Sussman *et al.* [125] simplifies to solving the equation

$$\delta_t = 1 - \|\nabla\delta\|. \tag{44}$$

Given the reinitialized distance function δ, the reinitialized vector distance function can be determined by solving[3] for $\boldsymbol{u}$ in

$$\boldsymbol{u} = \delta\nabla\delta.$$

5.2.2.4 Computing the Velocity Field on the Zero Band

It is sufficient to compute a velocity vector for every point in $\hat{\mathcal{B}}$. We know from [55] that $D\boldsymbol{u}(\boldsymbol{x})\boldsymbol{t} = \boldsymbol{0}$ for every element $\boldsymbol{t}$ of the tangent space to the curve. Thus, since the evolution equation is

$$\boldsymbol{u}_t - (D\boldsymbol{u})^T \boldsymbol{b} = \boldsymbol{0}$$

there is no need to project the velocity vector $\boldsymbol{b}$ onto the normal space of the curve. For example, for a mean curvature evolution, the velocity vector $\boldsymbol{b}$ is given by the mean curvature vector which can be computed [55] (for a one-dimensional curve) as

$$\mathcal{H}(\boldsymbol{x}) = -\left(\Delta\boldsymbol{u}(\boldsymbol{x})\right).$$

Section 5.4.1 contains the corresponding expressions for the normal dynamic active contour.

5.2.2.5 Extending the Velocity Field

We now show that the velocity field $\boldsymbol{b}$ satisfies Equation (41), if it is normal to $\mathcal{M}$ and extended normal to $\mathcal{M}$. Assume $\boldsymbol{b}$ is extended normally to $\mathcal{M}$, i.e., given any point $\boldsymbol{x} \in \Omega$, $\boldsymbol{b}$ will be constant in the $\boldsymbol{u}$ direction. Then,

$$D\boldsymbol{b}\boldsymbol{u} = \boldsymbol{0}$$

[3] Choose the gradient direction dictated by the numerical scheme.

and Equation (41) reduces to

$$\boldsymbol{b} = D\boldsymbol{u}\boldsymbol{b}.$$

We know that

$$D\boldsymbol{u}(\boldsymbol{x}) = D(\delta(\boldsymbol{x})D\delta(\boldsymbol{x})) = D\delta(\boldsymbol{x})D\delta(\boldsymbol{x}) + \delta(\boldsymbol{x})D(D\delta(\boldsymbol{x})).$$

But

$$\delta(\boldsymbol{x})D(D\delta(\boldsymbol{x}))\boldsymbol{b} = \boldsymbol{0},$$

since the gradient of $\delta(\boldsymbol{x})$ is constant along the normal direction, and $\boldsymbol{b}$ and the normal direction are linearly dependent. The remaining equation is

$$\boldsymbol{b} = \left(\nabla\delta(\boldsymbol{x})^T\boldsymbol{b}\right)\nabla\delta(\boldsymbol{x}),$$

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

By Section 5.2.2.4, only the normal direction of $\boldsymbol{b}$ matters for the evolution of $\boldsymbol{u}$. Thus, instead of solving Equation (41) directly, it is sufficient to extend the components of $\boldsymbol{b}$ along the normal direction to $\mathcal{M}$, where $\boldsymbol{b}$ does no longer need to be linearly dependent with $\boldsymbol{u}$ (since only its normal component matters for the evolution).

5.2.2.6 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 is not desirable.

We assume we are given a velocity field $\boldsymbol{b}$ in Ω as discussed in Section 5.2.2.5. Given the velocity field $\boldsymbol{b}$ in Ω, we evolve the vector distance function $\boldsymbol{u}$ using the negative velocity field $-\boldsymbol{b}$, i.e.,

$$\boldsymbol{u}_t - (D\boldsymbol{u})^T \boldsymbol{b} = \boldsymbol{0}. \tag{45}$$

To obtain increased numerical accuracy back and forth error compensation and correction [41] can be implemented. If $\mathcal{L}_h$ is the numerical solution operator of Equation (45) the backward error compensation method is given by the following three steps:

(1) Solve forward: $\tilde{\boldsymbol{u}}^{n+1} = \mathcal{L}_h \boldsymbol{u}^n$.

(2) Solve backward: $\boldsymbol{u}_1^n = \mathcal{L}_h^{-1} \tilde{\boldsymbol{u}}^{n+1}$.

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

We show an example of backward error compensation in Section 6.2, 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).

5.2.2.7 Redistancing the Vector Distance Function Field

Due to numerical errors, the evolving vector field $\boldsymbol{u}$ will drift away from the class of vector distance functions over time. Specifically, close to the zero level set, the vectors will no longer be perpendicular to the curve being evolved. Then the approach of Subsection 5.2.2.3 will no longer suffice.

In this case the vector field has to be reinitialized on the zero band (or alternatively, it needs to get reinitialized throughout the evolution). The question is how to solve the characteristic equation

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

numerically. Inspired by (44) it seems reasonable to evolve

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

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

$$\boldsymbol{u}_t + (D\boldsymbol{u})\, \boldsymbol{u} = \boldsymbol{u}. \tag{46}$$

Equation (46) has a particularly simple structure. It is a multi-dimensional transport equation with a right hand source term. Unfortunately, solving Equation (46) numerically is not straightforward. Only in the most trivial cases, e.g., 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 (46) becomes

$$u_t = u(1 - u_x). \tag{47}$$

This is 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 for implicit representations of one single point. Otherwise, the solution needs to be discontinuous, and needs to 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 (47), 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 [84]. 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 the shock-speed can be derived based on the conservation form of

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

which yields Equation (47) for $f(u) = \frac{1}{2}u^2$.

We obtain (see Figure 11)

$$\int_{x_1}^{x_1+\Delta x} u(x, t_1 + \Delta t)\, dx - \int_{x_1}^{x_1+\Delta x} u(x, t_1)\, dx = \int_{t_1}^{t_1+\Delta t} f(u(x_1, t))\, dt - \int_{t_1}^{t_1+\Delta t} f(u(x_1 + \Delta x, t))\, dt + \int_{t_1} t_1 + \Delta t \int_{x_1} x_1 + \Delta x u(x, t)\, dx\, dt. \tag{48}$$

Since the value for u is constant in each of the triangles of Figure 11, Equation (48) 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 $\Delta x = s\Delta t$,

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

resulting in an expression for the shock speed s:

$$s = \frac{f(u_r) - f(u_l)}{u_r - u_l}. \tag{49}$$

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 Equation (49) 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 will not always be the case. Redistancing performed in this way 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 12 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 [115]. In one dimension, the evolution equation is given as

$$\Phi_t + F\sqrt{\Phi_x^2} = \Phi_t + F\|\Phi_x\| = 0, \tag{50}$$

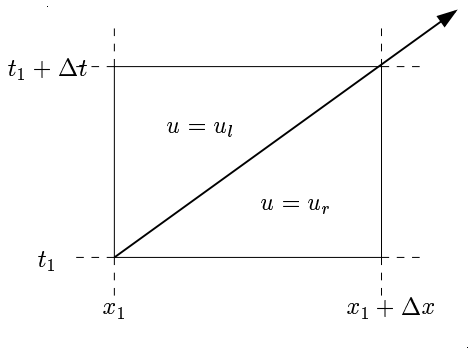


Figure 11: Shock propagation over an infinitesimal time interval Δt.

where F is the speed function. Differentiation of Equation (50) gives

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

Substituting $w = \Phi_x$ results in

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

which is in conservation form for w, the slope. Here, $H(\cdot)$ is the flux function.

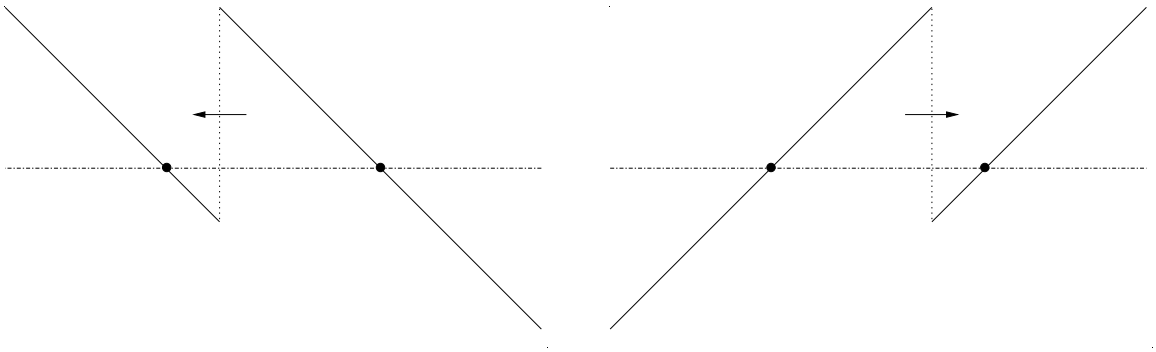
However, a numerical solution to Φ instead of w is sought, solving

$$\Phi_t + H(\Phi_x) = 0, \tag{51}$$

a Hamilton-Jacobi equation, using a numerical approximation to $H(\cdot)$ (the numerical flux function) for the computation. Equation (51) evolves a continuous function Φ as opposed to a discontinuous vector field $\boldsymbol{u}$.

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.

We propose 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.



(a) Shock propagating to the left.

(b) Shock propagating to the right.

Figure 12: Shock propagation for the vector distance function redistancing flow performed on a vector distance function representing two points.

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

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

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 $\hat{\mathcal{B}}$ can be reinitialized by explicitly computing $\text{dist}(\boldsymbol{x}, \mathcal{S}_c),\ \forall \boldsymbol{x} \in \hat{\mathcal{B}}$.

Discrete Connectivity Based Redistancing of the Vector Distance Function The discrete connectivity based approach hinges on a discrete approximation of the zero level set. To construct this discrete approximation, begin with a discrete representation of the zero band (as proposed in Section 5.2.2.2). This discrete approximation is then thinned. Ideally, obtaining 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 [130][4] whose union does not violate the property of every point

[4]For the purpose considered in this thesis, 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.

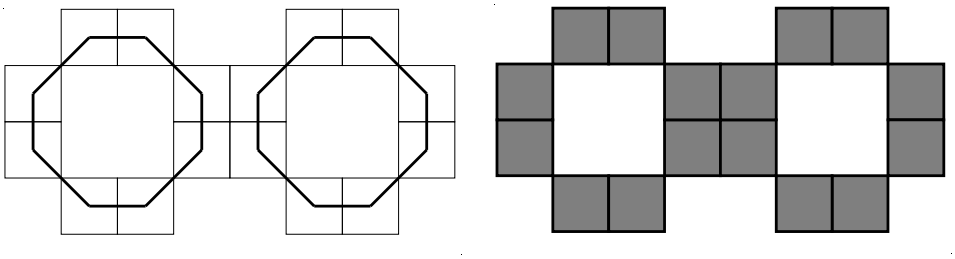


Figure 13: Example of a thinning of the zero band that does not yield two clearly separable discrete closed simple curves.

(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 Figure (13).

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 the 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 [79, 52]. However, this is the case relevant to the application considered. For now, assume the existence of a test for the simple point property in higher dimensions (Section 5.2.3 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 $\overline{\mathcal{N}}_{3^d-1}$ of a point $\boldsymbol{p}$ as

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

and base the decision for a simple point on this neighborhood. As a consequence, the thinning algorithm will no longer be topology preserving. Instead it will be allowed to break undesired connectivities (see Figure 14 for a two-dimensional example) based on the

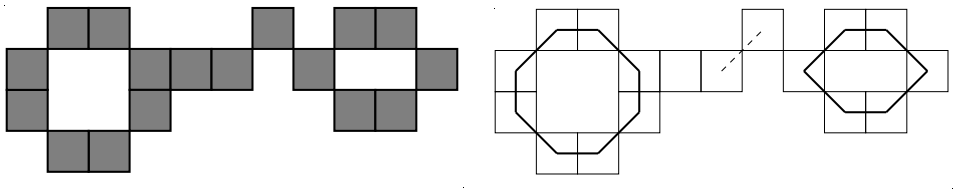


Figure 14: Undesired topological connection remaining after topological thinning.

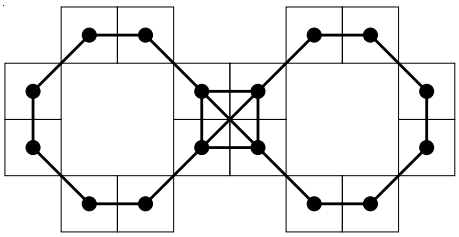


Figure 15: All possible edges and vertices based on the discrete connectivity.

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.

Given this 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 Figure (15) for an illustration.) We are interested in simple cycles of this graph that are consistent with the definition for discrete, simple closed curves. Figure (16) 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.

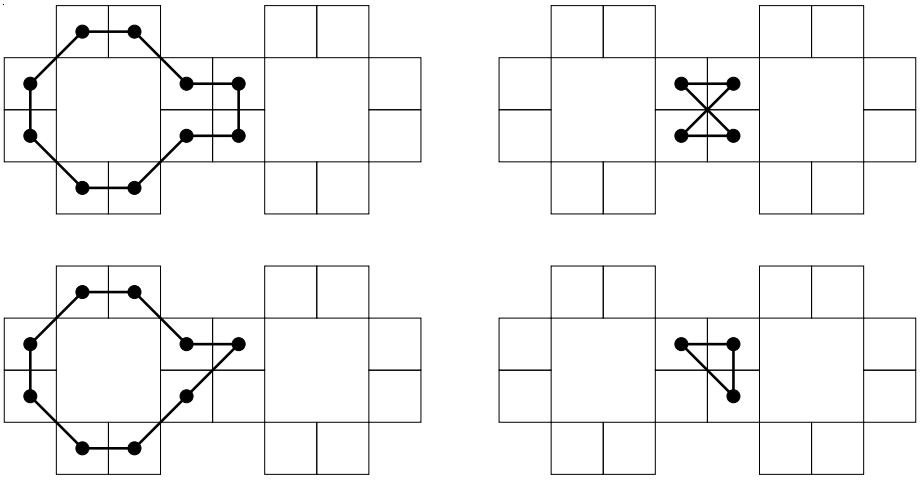


Figure 16: Some possible simple cycles that are not valid representatives for a discrete closed curve.

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 $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ (the vertices of the edge e), define the two line segments

$$\begin{aligned} \boldsymbol{x}(p) &= \boldsymbol{x}_0 + (\boldsymbol{x}_1 - \boldsymbol{x}_0)\, p, \\ \boldsymbol{x}'(p) &= \boldsymbol{p}_0 + (\boldsymbol{p}_1 - \boldsymbol{p}_0)\, p, \end{aligned}$$

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

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

The distance is then defined as (see Figure 17 for an illustration):

$$d\left(\boldsymbol{x}, \boldsymbol{x}'\right) = d(e) :=$$
$$\int_0^1 \| \left(\boldsymbol{x} - \boldsymbol{x}'\right)(p)\|^2 dp = \|\boldsymbol{u}(\boldsymbol{x}_0)\|^2 + \boldsymbol{u}(\boldsymbol{x}_0)^T \left(\boldsymbol{u}(\boldsymbol{x}_1) - \boldsymbol{u}(\boldsymbol{x}_0)\right) + \frac{1}{3}\|\boldsymbol{u}(\boldsymbol{x}_1) - \boldsymbol{u}(\boldsymbol{x}_0)\|^2.$$

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

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

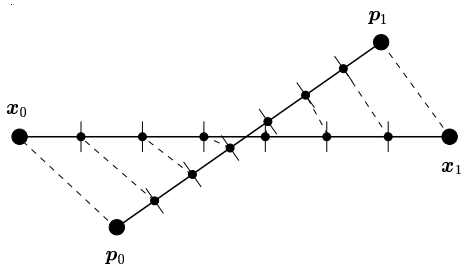


Figure 17: Illustration of the line quality measure.

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 smoothest of the cycles in $\mathcal{S}_e$ is the desired, discrete, simple closed-curve candidate whose edges are removed from $\mathcal{E}_u$.

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

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

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

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

The piecewise linear approximation of these cycles is still a relatively crude approximation. 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 $\boldsymbol{p}_0$ and $\boldsymbol{p}_1$ of a line segment, define

$$\boldsymbol{p}_m := \frac{1}{2}\left(\boldsymbol{p}_0 + \boldsymbol{p}_1\right) - \boldsymbol{u}\left(\boldsymbol{p}_0 + \boldsymbol{p}_1\right).$$

The line segment $(\boldsymbol{p}_0, \boldsymbol{p}_1)$ is then replaced by the two line segments $(\boldsymbol{p}_0, \boldsymbol{p}_m)$ and $(\boldsymbol{p}_m, \boldsymbol{p}_1)$.

This process can be repeated multiple times if necessary. To increase curve smoothness approximate this over-sampled piecewise linear approximation to the zero level set by a least squares quadratic spline [37] 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. Given a line segment L defined by its two endpoints $\boldsymbol{p}_0$ and $\boldsymbol{p}_1$, define the vector distance $\boldsymbol{u}$ of a point $\boldsymbol{p}$ to the line segment L as:

$$\boldsymbol{u}(\boldsymbol{p}, L) = \begin{cases} \boldsymbol{p} - \boldsymbol{p}_0 - \left((\boldsymbol{p} - \boldsymbol{p}_0)^T \boldsymbol{t}\right) \boldsymbol{t} & \text{if } (\boldsymbol{p} - \boldsymbol{p}_0)^T \boldsymbol{t} \in [0, \|\boldsymbol{p}_1 - \boldsymbol{p}_0\|] \\ \min(\|\boldsymbol{p}_0 - \boldsymbol{p}\|, \|\boldsymbol{p}_1 - \boldsymbol{p}\|) & \text{otherwise.} \end{cases},$$

where

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

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

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

5.2.3 Detecting Simple Points by Cubical Homology

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 [12, 79, 52, 78]. The approach in this thesis is based on cubical homology (as introduced in Section 2.3) and is not restricted to a specific space dimension.

We are concerned with the detection of simple points in $\mathbb{Z}^d$. Specifically, given the cubical set $\mathcal{N}(C)$

$$\mathcal{N}(C) := C \cup \left(\bigcup_i U_i \right),$$

where C and U_i are unitary cubes, and $U_i \in (3^d - 1) -$ discrete neighborhood of C, we want to know if we can remove the unitary cube C without changing the topology. Defining the sets

$$\begin{aligned} A_0 &:= \mathcal{N} \setminus (C \setminus (\mathcal{N} \cap \partial C)) \\ A_1 &:= C, \end{aligned}$$

with

$$\mathcal{N} = A_0 \cup A_1 \quad \text{and} \quad B = A_0 \cap A_1 = \mathcal{N} \cap \partial C,$$

there is a long exact sequence (Mayer-Vietoris)

$$\cdots \to H_k(B) \to H_k(A_0) \oplus H_k(A_1) \to H_k(\mathcal{N}) \to H_{k-1}(B) \to \ldots$$

For a point C (i.e., a discrete point represented by a unitary cube) to be simple it needs to be true that

$$H_*(\mathcal{N}) \cong H_*(A_0).$$

Note, that since the cubical set $\mathcal{N}$ is the union of unitary cubes in the discrete $(3^d - 1)$-neighborhood of C,

$$H_*(B) \cong H_*(A_0).$$

Thus, the Mayer-Vietoris sequence becomes

$$\cdots \to H_k(A_0) \to H_k(A_0) \oplus H_k(A_1) \to H_k(A_0) \to H_{k-1}(A_0) \to \ldots \tag{52}$$

Also, since A_1 is a unitary cube,

$$H_k(A_1) = \begin{cases} c\mathbb{Z} & \text{for } k = 0 \\ \emptyset & \text{otherwise,} \end{cases}$$

where c is an element of the equivalence class of the vertices of C. From Equation (52) it follows that

$$H_0(A_0) \to H_0(A_0) \oplus H_0(A_1) \to H_0(A_0).$$

Since this has to be an exact sequence, it follows that

$$H_0(A_0) \cong H_0(A_1) \cong c\mathbb{Z},$$

and similarly

$$H_k(A_0) \to H_k(A_0) \oplus H_k(A_1) \to H_k(A_0) \quad \text{for } k > 0$$

and thus

$$H_k(A_0) \cong H_k(A_1) \cong \emptyset \quad \text{for } k > 0.$$

Thus, the cube C can be removed from $\mathcal{N}$ without changing the topology if

$$\begin{aligned} \beta_0 &:= rank\,(H_0(A_0)) = 1 \\ \beta_k &:= rank\,(H_k(A_0)) = 0 \quad \text{for } k > 0. \end{aligned}$$

It is sensible to only look at the discrete $(3^d - 1)$-neighborhood of C, since topology preservation in this neighborhood implies overall topology preservation in our setting. To see that let's assume X to be a cubical set (composed of the union of unitary cubes), such that $\mathcal{N} \subset X$. Then defining

$$\begin{aligned} \overline{A}_0 &:= X \setminus \mathcal{N} \\ \overline{A}_1 &:= \mathcal{N}, \end{aligned}$$

with

$$\begin{aligned} X &= \overline{A}_0 \cup \overline{A}_1 \\ B &= \overline{A}_0 \cap \overline{A}_1 = \emptyset. \end{aligned}$$

yields (again, by Mayer-Vietoris)

$$\emptyset \to H_k(\overline{A}_0) \oplus H_k(\overline{A}_1) \to H_k(X) \to \emptyset.$$

But this implies (since the sequence is exact) that

$$H_k(\overline{A}_0) \oplus H_k(\overline{A}_1) \cong H_k(X),$$

or

$$H_k(X \setminus \mathcal{N}) \oplus H_k(\mathcal{N}) \cong H_k(X). \tag{53}$$

But since the two homology groups on the left hand side of Equation (53) only share the empty set, it is not possible to change the topology of $\mathcal{N}$ without changing the topology of X at the same time.

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 a d-xel changes the topology of the $(3^d - 1)$-neighborhood from $\beta_0 = 1,\ \beta_i = 0,\ i > 0$ to $\beta_0 = 1,\ \beta_1 = 1,\ \beta_i = 0,\ i > 1$, this removal is allowed.

5.3 Partial Level Set Approach

For the partial level set approach, a planar curve $\mathcal{C}$ is represented implicitly as described in Section 5.1 by means of the level set function Φ. The evolution of interest not only depends on the geometrical shape of the curve, but also on its current state (e.g., its normal velocities). This additional state information needs to be transported along with the curve. In the partial level set approach this is accomplished by solving one additional transport equations for each state that needs to be propagated:

$$s_t^i + \boldsymbol{x}_t \cdot \nabla s^i = 0,$$

where s^i represents the i-th additional state and $\boldsymbol{x}_t$ is the velocity the level set function Φ moves with. In the case where only velocity state information is present, the two functions $u : \mathbb{R}^2 \times \mathbb{R}^+$ and $v : \mathbb{R}^2 \times \mathbb{R}^+$, representing the x and the y components of the velocity vector respectively ($\boldsymbol{x}_t = [u, v]^T$), are propagated along with the zero level set of Φ by solving two additional transport equations (for normal curve propagation, one additional transport

equation will be sufficient; the overall scheme will be identical). Specifically the following algorithm is proposed for numerical implementations[5]

1) Compute the current velocities at every point of the contour based on equation (17).

2) Update the velocity fields u and v using the results from step 1.

3) Propagate Φ_t, u, and v by one time step using the velocities from step 1. For u and v this amounts to solving

$$\begin{aligned} u_t + \boldsymbol{x}_t \cdot \nabla u &= 0 \\ v_t + \boldsymbol{x}_t \cdot \nabla v &= 0 \end{aligned}$$

respectively. Note, that it is important to propagate u and v in the direction $\boldsymbol{x}_t$ opposed to the normal direction with respect to the curve in this case.

This approach can be used to propagate any kind of information along with the contour. By distributing marker particles on the contour one could use this method for region tracking (compare Bertalmio and Sapiro [11] where region tracking is performed by intersecting two hypersurfaces, using ideas presented in Section 5.2). The advantage of a partial level set approach compared to the full level set approach discussed in Section 5.4 is its low computational complexity. For normal geometric dynamic curve evolution the complexity will only be about twice as high as for a normal, static level set evolution. The price to pay is the loss in topological flexibility: contours are not allowed to intersect each other and get merged solely based on their shape in the image plane. The state information is not used to influence the topology changes.

Section 5.3.1 sets up the framework for the partial level set approach for the normal geometric dynamic curve evolution. Section 5.3.2 discusses the mathematical properties of the partial level set evolution for the normal geometric dynamic curve evolution. Results are presented in Section 6.1.

[5]This implementation needs to be tied into a whole numerical scheme. This is beyond the scope of this thesis. See [115, 104, 84] for details.

5.3.1 Normal Geometric Dynamic Curve Evolution

The normal geometric dynamic curve evolution (see Section (3.5)) is given as

$$\begin{aligned} \mathcal{C}_t &= \beta(s,t)\mathcal{N} \\ \beta_t &= \frac{1}{2}\beta^2\kappa + \frac{1}{\mu}g\kappa - \frac{1}{\mu}\nabla g \cdot \mathcal{N}. \end{aligned} \tag{54}$$

Since the evolution of the curve's shape is independent of the tangential velocity, the level set evolution equation for an arbitrary velocity $\boldsymbol{x}_t$ can be written as

$$\Phi_t + \|\nabla\Phi\|\mathcal{N} \cdot \boldsymbol{x}_t = 0, \tag{55}$$

where

$$\mathcal{N} = -\frac{\nabla\Phi}{\|\nabla\Phi\|}.$$

Here, $\boldsymbol{x}_t = \tilde{\beta}\mathcal{N}$, where

$$\tilde{\beta}(\boldsymbol{x},t) = \beta(p,t) \tag{56}$$

is the spatial normal velocity at the point $\boldsymbol{x}$. This simplifies Equation (55) to

$$\Phi_t - \tilde{\beta}\|\nabla\Phi\| = 0. \tag{57}$$

Substituting Equation (56) into Equation (31) and using the relation

$$\kappa = \nabla \cdot \left(\frac{\nabla\Phi}{\|\nabla\Phi\|}\right)$$

yields

$$\tilde{\beta}_t - \tilde{\beta}\nabla\tilde{\beta}\frac{\nabla\Phi}{\|\nabla\Phi\|} = \left(\frac{1}{2}\tilde{\beta}^2 + \frac{1}{\mu}g\right)\kappa + \frac{1}{\mu}\nabla g \cdot \frac{\nabla\Phi}{\|\nabla\Phi\|}. \tag{58}$$

The left hand side of Equation (58) is the material derivative for the normal velocity. If we use extension velocities, Equation (58) simplifies to

$$\tilde{\beta}_t = \left(\frac{1}{2}\tilde{\beta}^2 + \frac{1}{\mu}g\right)\kappa + \frac{1}{\mu}\nabla g \cdot \frac{\nabla\Phi}{\|\nabla\Phi\|}.$$

Since the extensions are normal to the contours, normal propagation of the level set function will guarantee a constant velocity value along the propagation direction (up to numerical errors). Specifically $\nabla\tilde{\beta}\perp\nabla\Phi$ in this case and thus

$$\nabla\Phi \cdot \nabla\tilde{\beta} = 0.$$

For an alternative derivation[6], we change our Lagrangian, and extend it over a range of level sets. For each time t, and $0 \leq r \leq 1$ let

$$\mathcal{C}^{(r)}(t) := \{(x, y) \in \mathbb{R}^2 : \Phi(x, y, t) = r\}.$$

Using the Lagrangian

$$L = \int_0^1 \int_{\mathcal{C}^{(r)}(t)} \left(\frac{1}{2}\mu\tilde{\beta}^2 - g \right) \, ds \, dr$$

the action integral becomes

$$\mathcal{L} = \int_t \int_0^1 \int_{\mathcal{C}^{(r)}(t)} \left(\frac{1}{2}\mu\tilde{\beta}^2 - g \right) \, ds \, dr \, dt,$$

which is

$$\begin{aligned} \mathcal{L} &= \int_0^1 \int_t \int_{\mathcal{C}^{(r)}(t)} \left(\frac{1}{2}\mu\tilde{\beta}^2 - g \right) \, ds \, dt \, dr \\ &= \int_t \left(\int_0^1 \int_{\mathcal{C}^{(r)}(t)} \left(\frac{1}{2}\mu\tilde{\beta}^2 - g \right) \, d\mathcal{H}^1 \lfloor_{\mathcal{C}^{(r)}(t)} \, dr \right) dt \\ &= \int_t \int_\Omega \left(\frac{1}{2}\mu\tilde{\beta}^2 - g \right) \|\nabla\Phi\| \, dx \, dy \, dt, \end{aligned} \tag{59}$$

where $\mathcal{H}^1$ is the one-dimensional Hausdorff measure and we applied the coarea formula [13]. This casts the minimization problem into minimization over an interval of level sets in a fixed coordinate frame (x and y are time independent coordinates in the image plane). Using Equation (57), $\tilde{\beta}$ becomes

$$\tilde{\beta} = \frac{\Phi_t}{\|\nabla\Phi\|}. \tag{60}$$

Substituting (60) into Equation (59) yields

$$\mathcal{L}[\Phi] := \int_t \int_\Omega \left(\mu \frac{\Phi_t^2}{2\|\nabla\Phi\|} - g\|\nabla\Phi\| \right) \, dx \, dy \, dt,$$

which is the new Φ-dependent action integral to be minimized. Then, $\delta\mathcal{L} = 0$ if and only if

$$\frac{\partial}{\partial t}\left(\frac{\Phi_t}{\|\nabla\Phi\|} \right) = \nabla \cdot \left(\left(\frac{g}{\mu} + \frac{\Phi_t^2}{\|\nabla\Phi\|^2} \right) \frac{\nabla\Phi}{\|\nabla\Phi\|} \right).$$

The curve evolution is thus governed by the equation system:

$$\begin{aligned} \tilde{\beta}_t &= \nabla \cdot \left(\frac{\nabla\Phi}{\|\nabla\Phi\|} \left(\frac{g}{\mu} + \frac{1}{2}\tilde{\beta}^2 \right) \right), \\ \Phi_t &= \tilde{\beta}\|\nabla\Phi\|. \end{aligned} \tag{61}$$

[6]This will yield directly the normal evolution equation, without the detour of deriving Equation (30).

Expanding Equation (61) yields again

$$\tilde{\beta}_t = \left(\frac{1}{2}\tilde{\beta}^2 + \frac{1}{\mu}g\right)\kappa + \frac{1}{\mu}\nabla g \cdot \frac{\nabla\Phi}{\|\nabla\Phi\|} + \tilde{\beta}\nabla\tilde{\beta}\cdot\frac{\nabla\Phi}{\|\nabla\Phi\|}.$$

The equation system (61) constitutes a conservation law for the normal velocity $\tilde{\beta}$. The propagation of the level set function Φ is described (as usual) by a Hamilton-Jacobi equation.

5.3.2 Mathematical Properties

This section presents some mathematical properties of the partial level set approach for the normal geometric dynamic curve evolution.

5.3.2.1 Energy Bounds

Consider the energy functional[7]

$$E = \int_\Omega \left\{\frac{1}{2}\frac{\Phi_t^2}{\|\nabla\Phi\|} + g(x,y,t)\|\nabla\Phi\|\right\}\, dx\, dy = \int_\Omega \left\{\frac{1}{2}\tilde{\beta}^2 + g(x,y,t)\right\}\|\nabla\Phi\|\, dx\, dy.$$

It changes with time according to

$$\begin{aligned}\frac{dE}{dt} &= \int_\Omega \left\{\tilde{\beta}\tilde{\beta}_t + g_t\right\}\|\nabla\Phi\| + \left(\frac{1}{2}\tilde{\beta}^2 + g\right)\frac{\nabla\Phi}{\|\nabla\Phi\|}\cdot\nabla\Phi_t\, dx\, dy \\ &= \int_\Omega g_t\|\nabla\Phi\| - \int_\Omega v\|\nabla\Phi\|\nabla\cdot\left\{\left(\frac{1}{2}\tilde{\beta}^2 + g\right)\frac{\nabla\Phi}{\|\nabla\Phi\|}\right\} + \left(\frac{1}{2}\tilde{\beta}^2 + g\right)\frac{\nabla\Phi}{\|\nabla\Phi\|}\cdot\nabla\left(v\|\nabla\Phi\|\right)\, dx\, dy \\ &= \int_\Omega g_t\|\nabla\Phi\|\, dx\, dy\end{aligned}$$

Note that

$$(\nabla\Phi)_t = \nabla(\Phi_t), \tag{62}$$

since x and y are independent of t.

5.3.2.2 Evolution of Graphs

Let Γ_t be the graph of a function $y = h(x,t)$. Then

$$\beta = \frac{h_t}{\sqrt{1+h_x^2}}, \quad N = \frac{1}{\sqrt{1+h_x^2}}\begin{pmatrix} -h_x \\ 1 \end{pmatrix}, \qquad \kappa = \frac{h_{xx}}{(1+h_x^2)^{3/2}}$$

[7] $\mu = 1$ in what follows. This does not change the qualitative behavior of the evolution equation: μ can always be subsumed in the expression for g, if desired.

and time derivatives following the motion of the curve are given by

$$\frac{D}{\partial t} = \frac{\partial}{\partial t} - \beta \frac{h_x}{\sqrt{1+h_x^2}} \frac{\partial}{\partial x} = \frac{\partial}{\partial t} - \frac{h_t h_x}{1+h_x^2} \frac{\partial}{\partial x}.$$

Hence the law of motion $\frac{D}{\partial t}\beta = (\beta^2/2 + g)\kappa - \mathcal{N} \cdot \nabla g$ for the curve Γ_t implies a partial differential equation for h

$$\frac{\partial}{\partial t}\left(\frac{h_t}{\sqrt{1+h_x^2}}\right) - \frac{h_t h_x}{1+h_x^2}\frac{\partial}{\partial x}\left(\frac{h_t}{\sqrt{1+h_x^2}}\right) = \left\{\frac{1}{2}\frac{h_t^2}{1+h_x^2} + g(x,h,t)\right\}\frac{h_{xx}}{(1+h_x^2)^{3/2}} - \frac{g_y - h_x g_x}{\sqrt{1+h_x^2}}. \quad (63)$$

Observe that

$$\frac{\partial}{\partial x}\frac{h_x}{\sqrt{1+h_x^2}} = \frac{h_{xx}(1+h_x^2) - h_x^2 h_{xx}}{(1+h_x^2)^{3/2}} = \frac{h_{xx}}{(1+h_x^2)^{3/2}},$$

and hence

$$\frac{h_t h_x}{1+h_x^2}\frac{\partial}{\partial x}\left(\frac{h_t}{\sqrt{1+h_x^2}}\right) = \frac{h_x}{\sqrt{1+h_x^2}}\frac{\partial}{\partial x}\left(\frac{1}{2}\frac{h_t^2}{1+h_x^2}\right) = \frac{\partial}{\partial x}\left\{\frac{1}{2}\frac{h_x h_t^2}{(1+h_x^2)^{3/2}}\right\} - \frac{1}{2}\frac{h_t^2 h_{xx}}{(1+h_x^2)^{5/2}}.$$

Also

$$g\frac{h_{xx}}{(1+h_x^2)^{3/2}} = \frac{\partial}{\partial x}\left\{\frac{g h_x}{\sqrt{1+h_x^2}}\right\} - \frac{h_x}{\sqrt{1+h_x^2}}\left(g_x + g_y h_x\right),$$

so that

$$g\frac{h_{xx}}{(1+h_x^2)^{3/2}} + \frac{h_x g_x}{\sqrt{1+h_x^2}} = \frac{\partial}{\partial x}\left\{\frac{g h_x}{\sqrt{1+h_x^2}}\right\} - \frac{h_x^2 g_y}{\sqrt{1+h_x^2}}.$$

Applying all this to Equation (63) one gets

$$\frac{\partial}{\partial t}\left(\frac{h_t}{\sqrt{1+h_x^2}}\right) - \frac{\partial}{\partial x}\left\{\frac{1}{2}\frac{h_x h_t^2}{(1+h_x^2)^{3/2}} + \frac{g h_x}{\sqrt{1+h_x^2}}\right\} = -g_y\sqrt{1+h_x^2}. \quad (64)$$

Expanding (63),

$$\frac{\partial}{\partial t}\left(\frac{h_t}{\sqrt{1+h_x^2}}\right) - \frac{h_t h_x}{1+h_x^2}\frac{\partial}{\partial x}\left(\frac{h_t}{\sqrt{1+h_x^2}}\right) = \left\{\frac{1}{2}\frac{h_t^2}{1+h_x^2} + g(x,h,t)\right\}\frac{h_{xx}}{(1+h_x^2)^{3/2}} - \frac{g_y - h_x g_x}{\sqrt{1+h_x^2}}$$

yields:

$$\frac{h_{tt}(1+h_x^2) - h_t h_x h_{xt}}{(1+h_x^2)^{3/2}} - \frac{h_t h_x}{1+h_x^2}\frac{h_{xt}(1+h_x^2) - h_x h_t h_{xx}}{(1+h_x^2)^{3/2}} = \left\{\frac{1}{2}\frac{h_t^2}{1+h_x^2} + g(x,h,t)\right\}\frac{h_{xx}}{(1+h_x^2)^{3/2}} - \frac{g_y - h_x g_x}{\sqrt{1+h_x^2}}$$

and after simplifying and multiplying with $(1+h_x^2)^{3/2}$ this leads to

$$h_{tt}(1+h_x^2) - h_t h_x h_{xt} - h_t h_x h_{xt} + \frac{h_x^2 h_t^2}{1+h_x^2} h_{xx} = \left\{ \frac{1}{2} \frac{h_t^2}{1+h_x^2} + g(x,h,t) \right\} h_{xx} - (g_y - h_x g_x)\left(1+h_x^2\right),$$

i.e.,

$$h_{tt} \underbrace{(1+h_x^2)}_{a} = \underbrace{2h_t h_x}_{-2b} h_{xt} + \underbrace{\left\{ \frac{\frac{1}{2} - h_x^2}{1+h_x^2} h_t^2 + g(x,h,t) \right\}}_{-c} h_{xx} - (g_y - h_x g_x)\left(1+h_x^2\right). \qquad (65)$$

Since

$$ac - b^2 = -\left(1+h_x^2\right)\left\{ \frac{\frac{1}{2} - h_x^2}{1+h_x^2} h_t^2 + g(x,h,t) \right\} - h_t^2 h_x^2 = -\frac{1}{2} h_t^2 - g(x,h,t)\left(1+h_x^2\right) < 0,$$

the evolution equation is hyperbolic.

5.3.2.3 Evolution of Graphs again

Equation (65) can be derived based on the variational principle. Then using

$$\beta = \frac{h_t}{\sqrt{1+h_x^2}}, \text{ and } ds = \sqrt{1+h_x^2}\, dx,$$

the action integral becomes

$$S = \int_0^T \int_a^b \left\{ \frac{v^2}{2} - g \right\} ds\, dt = \int_0^T \int_a^b \left\{ \frac{1}{2} \frac{h_t^2}{1+h_x^2} - g(x, h(x,t), t) \right\} \sqrt{1+h_x^2}\, dx\, dt.$$

The Lagrangian is

$$L(x,t,h,h_x,h_t) = \frac{1}{2} \frac{h_t^2}{\sqrt{1+h_x^2}} - g(x,h,t)\sqrt{1+h_x^2}.$$

Computing the first variation yields the Euler-Lagrange equation,

$$\frac{d}{dt}\left(\frac{\partial L}{\partial h_t}\right) + \frac{d}{dx}\left(\frac{\partial L}{\partial h_x}\right) = \frac{\partial L}{\partial h}.$$

We obtain

$$\frac{d}{dt}\left(\frac{h_t}{\sqrt{1+h_x^2}}\right) - \frac{d}{dx}\left(\frac{\frac{1}{2}h_t^2 + g(1+h_x^2)}{(1+h_x^2)^{3/2}} h_x\right) = \sqrt{1+h_x^2} \frac{\partial g}{\partial y}(x,h,t).$$

After some algebraic manipulations this yields

$$h_{tt}\underbrace{(1+h_x^2)^2}_{a} =$$
$$\underbrace{2h_th_x(1+h_x^2)}_{-2b}h_{tx}+\underbrace{\left(g(h_x^2+1)+h_t^2\left(\frac{1}{2}-h_x^2\right)\right)}_{-c}h_{xx}+(g_x+g_yh_x)h_x(1+h_x^2)^2-g_y(1+h_x^2)^3 \tag{66}$$

which is the same as (64). Since

$$ac-b^2=-(1+h_x^2)^2\left(g(1+h_x^2)+\frac{1}{2}h_t^2\right)<0,$$

Equation (66) is hyperbolic as long as g and h_t are not both zero at a given point.

5.3.2.4 Evolution of Nearly Straight Curves

Assume $g(x,y,t)\approx G+\frac{K}{2}(y-at)^2$ and $h(x,t)=at+\varepsilon k(x,t)+O(\varepsilon^2)$, then (65) reduces to

$$\varepsilon k_{tt}=\left\{\frac{1}{2}a^2+G\right\}\varepsilon k_{xx}-K(h-at)+O(\varepsilon^2),$$

and hence, after dropping the $O(\varepsilon^2)$ terms, to

$$k_{tt}=c_a^2k_{xx}-Kk \text{ where } c_a=\frac{1}{2}a^2+G.$$

This is a telegraph equation [47].

5.3.2.5 Evolution Equation Type

As shown in Section 5.3.2.2, the graph evolution Equation (66) is hyperbolic. It remains to determine the evolution type for the level set evolution equation.

We can write the equation as

$$L[\Phi]=\Phi_{tt}+2\sum_{i\in\{x,y\}}a^i\Phi_{it}-\sum_{i,k\in\{x,y\}}a^{ik}\Phi_{ik}+\cdots=0, \tag{67}$$

which in turn can be written as

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & a^{11} & a^{12} \\ 0 & a^{21} & a^{22} \end{pmatrix}}_{A^0} \begin{pmatrix} \Phi_{tt} \\ \Phi_{xt} \\ \Phi_{yt} \end{pmatrix} + \underbrace{\begin{pmatrix} 2a^1 & -a^{11} & -a^{12} \\ -a^{11} & 0 & 0 \\ -a^{12} & 0 & 0 \end{pmatrix}}_{A^1} \begin{pmatrix} \Phi_{tx} \\ \Phi_{xx} \\ \Phi_{yx} \end{pmatrix} +$$

$$+ \underbrace{\begin{pmatrix} 2a^2 & -a^{21} & -a^{22} \\ -a^{21} & 0 & 0 \\ -a^{22} & 0 & 0 \end{pmatrix}}_{A^2} \begin{pmatrix} \Phi_{ty} \\ \Phi_{xy} \\ \Phi_{yy} \end{pmatrix} + \cdots = 0, \quad (68)$$

since $\Phi_{xt} = \Phi_{tx}$ and $\Phi_{yt} = \Phi_{ty}$. For Equation (68) to be symmetric hyperbolic, one of the matrices A^i or a linear combination

$$\sum_{i=0}^{2} \xi^i A^i$$

has to be definite [29]. In this discussion, we assume that $\|\nabla\Phi\| \neq 0$ everywhere. Then with

$$\begin{aligned} a^{11} &= -\left(\frac{1}{2}\tilde{\beta}^2 + g\right)\left(\frac{\Phi_x^2}{\|\nabla\Phi\|^2} - 1\right) - \tilde{\beta}^2\frac{\Phi_x^2}{\|\nabla\Phi\|^2}, \\ a^{12} &= -\frac{\Phi_x\Phi_y}{\|\nabla\Phi\|^2}\left(\frac{3}{2}\tilde{\beta}^2 + g\right), \\ a^{21} &= a^{12}, \\ a^{22} &= -\left(\frac{1}{2}\tilde{\beta}^2 + g\right)\left(\frac{\Phi_y^2}{\|\nabla\Phi\|^2} - 1\right) - \tilde{\beta}^2\frac{\Phi_y^2}{\|\nabla\Phi\|^2}, \end{aligned}$$

we obtain the eigenvalues of A^0

$$\lambda(A^0) = \left\{1, -\tilde{\beta}^2, \frac{1}{2}\tilde{\beta}^2 + g\right\},$$

with the corresponding eigenvectors

$$\lambda_1 = 1 \rightarrow \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \lambda_2 = -\tilde{\beta}^2 \rightarrow \begin{pmatrix} 0 \\ \Phi_x \\ \Phi_y \end{pmatrix}, \quad \lambda_3 = \frac{1}{2}\tilde{\beta}^2 + g \rightarrow \begin{pmatrix} 0 \\ -\Phi_y \\ \Phi_x \end{pmatrix}.$$

The eigenvectors point into the t direction and into the normal and tangential directions to the curve respectively. None of the matrices A^i and none of the linear combinations are

definite, since two of the eigenvalues are positive (or zero) and one is negative (or zero). Thus the equation is not hyperbolic symmetric. In particular, it is not hyperbolic in the normal direction to the curve. The equation can be written in the form of Equation (67) as

$$\Phi_{tt} - 2\beta\frac{\Phi_x}{\|\nabla\Phi\|}\Phi_{tx} - 2\beta\frac{\Phi_y}{\|\nabla\Phi\|}\Phi_{ty} - a^{11}\Phi_{xx} - 2a^{12}\Phi_{xy} - a^{22}\Phi_{yy} - \|\nabla\Phi\|^2 = 0.$$

If $\tilde{\beta} = 0$ at a point $\boldsymbol{x}$, the level set function Φ does not change at $\boldsymbol{x}$ and the evolution equation becomes

$$\Phi_{tt} = \nabla g \cdot \nabla\Phi + g\frac{\Phi_y^2\Phi_{xx} - 2\Phi_x\Phi_y\Phi_{xy} + \Phi_x^2\Phi_{yy}}{\|\nabla\Phi\|} = \nabla g \cdot \nabla\Phi + g\|\nabla\Phi\|\kappa.$$

5.3.2.6 Special Solutions

Let $g = g(r,t)$ be radially symmetric, and let Γ_t be a a circle with radius $R(t)$. Then $v = -R'(t)$, $\kappa = 1/R(t)$ and thus the circle will evolve governed by the equation

$$R''(t) = \left(\frac{1}{2}R'(t)^2 + g(R,t)\right)\frac{-1}{R} - g_r(R,t)$$

or,

$$RR'' + \frac{1}{2}R'^2 = -g(R,t) - Rg_r(R,t),$$

(divide by $\sqrt{R}$)

$$\frac{2}{3}\left(R^{3/2}\right)'' = R^{1/2}R'' + \frac{1}{2}R^{-1/2}(R')^2 = -\frac{g + Rg_r}{R^{1/2}}.$$

One can also write down the variational principle which reduces to

$$\delta\int_0^T \left\{\frac{1}{2}R'^2 - g(R,t)\right\} 2\pi R\, dt = 0.$$

The Lagrangian is $L(R,R',t) = \frac{1}{2}RR'^2 - Rg(R,t)$; the momentum is $p_R = RR'$, and the corresponding Hamiltonian is

$$H(R,p_R,t) = \frac{p_R^2}{2R} + Rg(R,t).$$

The Hamiltonian equations are

$$\frac{dR}{dt} = \frac{p_R}{R}, \quad \frac{dp_R}{dt} = \frac{p_R^2}{R^2} - g - Rg_r.$$

If g does not depend on time, then

$$H = \frac{p_R^2}{2R} + Rg(R) = \frac{1}{2}R\left(\frac{dR}{dt}\right)^2 + Rg(R)$$

is a conserved quantity.

If

$$g = G + \frac{K}{4}(1 - R^2)^2 = G + \frac{K}{4} - \frac{K}{2}R^2 + \frac{K}{4}R^4,$$

then

$$g + Rg_r = (Rg)_R = G + \frac{K}{4} - \frac{3K}{2}R^2 + \frac{5K}{4}R^4,$$

and one obtains the system.

$$R' = \frac{p_R}{R}, \quad p_R' = \frac{p_R^2}{R^2} - G - \frac{K}{4} + \frac{3K}{2}R^2 - \frac{5K}{4}R^4.$$

5.4 Full Level Set Approach

Unlike the partial level set approach of Subsection 5.3, a full level set approach evolves a completely implicit representation of the curve based on Equation (17). This allows for full topological flexibility.

The method of choice for the full level set approach in this thesis is the vector distance function approach. As discussed in Section 5.2, the mathematical theory for vector distance functions is still in its infancy, however initialization of a vector distance function is straightforward and (unlike methods based on the intersection of the level sets of multiple scalar functions) a narrow-band approach is possible, reducing the computational cost significantly.

Section 5.4.1 presents the full level set approach for the normal geometric dynamic curve evolution, the demo problem already employed for the partial level set approach.

5.4.1 Normal Geometric Dynamic Curve Evolution

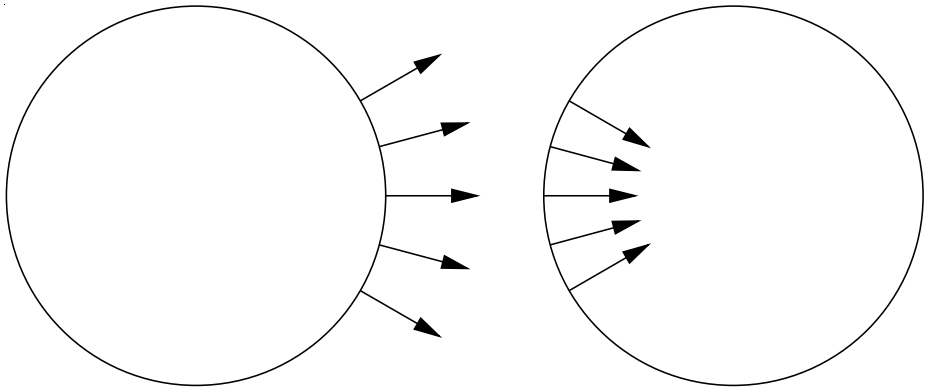
As shown in Section 3.5, the governing equations for normal geometric dynamic curve evolution are:

$$\begin{aligned} \mathcal{C}_t &= \beta\mathcal{N}, \\ \beta_t &= \left(\frac{1}{2}\beta^2 + \frac{1}{\mu}g\right)\kappa - \frac{1}{\mu}\nabla g \cdot \mathcal{N}, \end{aligned} \tag{69}$$

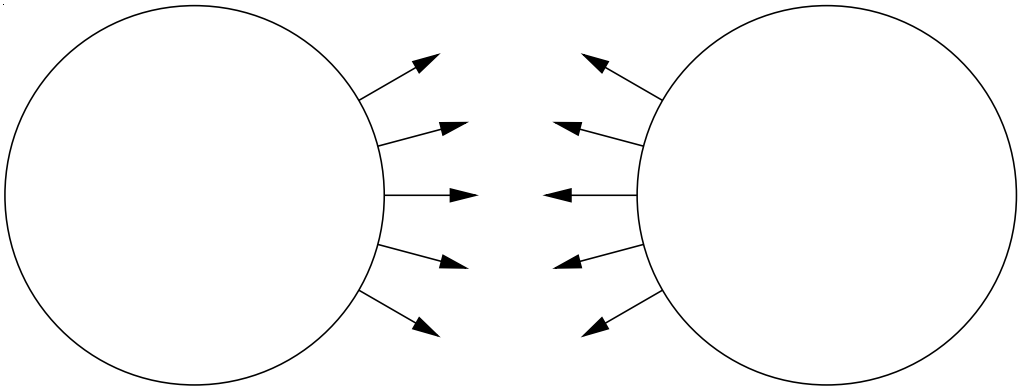
with β being the speed in the normal direction.

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) 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 Figure (18) 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.

The partial level set implementation (see Section 5.3) is not concerned with the propagation of 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 Equation (69) as long as the level sets, used for the representation of the curves' positions in the image plane, do not merge or split; curves sliding past each other cannot be represented in this setting. If this is desired, a full level set approach, i.e., a level set method operating in the full four dimensional space, needs to be employed. Then a completely implicit representation of the curve based on Equation (69) is evolved. This allows 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



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



(b) Curves should not merge, but merge if only the normal velocity gets propagated.

Figure 18: Merging curves scenarios.

description requires them to do. The projection onto the image plane will then show curves intersecting each other. 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. Here, the two contours will get fused.

We will treat the spatial and the velocity evolution combined within the vector distance function setting (as a full level set approach), thus providing a sample problem for the vector distance function approach. Equation (69) was derived based on curves evolving in the plane. However, for an implicit evolution by a vector distance function approach planar quantities (e.g., the mean curvature vector in the image plane) need to be computed based on the vector distance function values $\boldsymbol{u}$ which implicitly describe a curve in $\mathbb{R}^4$. In short, Equation (69) needs to be written in terms of $\boldsymbol{u}$.

With

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

an evolution equation analogous to Equation (69) is

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

which is (upon substitution of Equation (69))

$$\begin{aligned} \mathcal{C}_t &= \boldsymbol{v} \\ \boldsymbol{v}_t &= \left(\frac{1}{2}\|\boldsymbol{v}\|^2 + \frac{1}{\mu} g\right) \kappa \mathcal{N} - \frac{1}{\mu} \left(\nabla g \cdot \mathcal{N}\right) \mathcal{N} - \frac{1}{2} \left(\|\boldsymbol{v}\|^2\right)_s \mathcal{T}. \end{aligned} \tag{70}$$

To implement Equation (70) in a level set framework, the tangential, $\mathcal{T}$, and the normal, $\mathcal{N}$, vectors need to be written in terms of $\boldsymbol{u}$. Also, a replacement for the mean curvature vector in the plane, $\mathcal{H}_p = \kappa \mathcal{N}$ has to be found. Since the object dimension is one, exactly one eigenvalue of $D\boldsymbol{u}$ will be zero, while the others will be one [55]. 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

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

where

$$\boldsymbol{t} = \underset{\forall \boldsymbol{v}_i \in \mathcal{V}}{argmin} \|\lambda_i\| (\boldsymbol{v}_i), \quad \mathcal{V} = \left\{ \boldsymbol{v} \in \mathbb{R}^4 : \frac{1}{2} \left(D\boldsymbol{u} + (D\boldsymbol{u})^T \right) \boldsymbol{v} = \lambda \boldsymbol{v},\ \lambda \in \mathbb{R} \right\}.$$

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

$$\mathcal{T} = \frac{\mathcal{P}(\mathcal{T}_f)}{\|\mathcal{P}(\mathcal{T}_f)\|}, \quad \text{where} \quad \mathcal{P}\left(\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix}^T \right) = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}.$$

We define

$$\boldsymbol{r} := \mathcal{T}^{\perp} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \mathcal{T},$$

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

$$\mathcal{H}_p = \mathcal{T}_s = \left(\frac{1}{\|\mathcal{T} \cdot \mathcal{T}_f\|} (D\mathcal{T})\, \mathcal{T}_f \cdot \boldsymbol{r} \right) \boldsymbol{r}$$

and the squared velocity variation along the curve as

$$\left(\beta^2\right)_s = \frac{1}{\|\mathcal{T} \cdot \mathcal{T}_f\|} \nabla \left(\|\boldsymbol{v} \cdot \boldsymbol{r}\|^2 \right) \mathcal{T}_f,$$

where the quantities that should be normal to the planar curve are replaced by their projections onto $\boldsymbol{o}$ to increase numerical robustness. We cannot guarantee that $\boldsymbol{v}$ will indeed be normal to the curve throughout the evolution due to numerical errors. Defining the reprojected velocity to be

$$\boldsymbol{v}_R := (\boldsymbol{v} \cdot \boldsymbol{r})\, \boldsymbol{r},$$

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

$$\begin{aligned} \mathcal{C}_t &= \boldsymbol{v}_R \\ \boldsymbol{v}_t &= \left(\frac{1}{2} \|\boldsymbol{v}_R\|^2 + \frac{1}{\mu} g \right) \mathcal{H}_p - \frac{1}{\mu} (\nabla g \cdot \boldsymbol{r})\, \boldsymbol{r} - \frac{1}{2\|\mathcal{T} \cdot \mathcal{T}_f\|} \left(\nabla \left(\|\boldsymbol{v}_R\|^2 \right) \mathcal{T}_f \right) \mathcal{T}. \end{aligned} \tag{71}$$

In principle $\boldsymbol{v}$ 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 Equation (71) to

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

The newly introduced term will ensure that the tangential velocity components will vanish for $t \to \infty$ (based on the amplification factor $K > 0$). The level set evolution equation is then

$$\boldsymbol{u}_t - (D\boldsymbol{u})^T \begin{pmatrix} \mathcal{C}_t \\ \boldsymbol{v}_t \end{pmatrix} = \mathbf{0}.$$

CHAPTER 6

SIMULATION RESULTS

This chapter presents simulation results for the theory developed previously. Section 6.1 presents simulation results for the partial level set method. Section 6.2 shows results obtained using the full level set approach.

6.1 Results for the Partial Level Set Approach

We present two sets of results for the partial level set method. The first set (in Section 6.1.1) shows results for the normal geometric dynamic contour evolution on synthetically created blob sequences. Error injection is not used. Results for two real image sequences (a fish and a car) are given in Section 6.1.2. Both use error injection. The car sequence also uses the occlusion detection proposed in Section 4.2.

6.1.1 Tracking of Blob Shapes

To get insight into the behavior of the normal geometric dynamic active contour, it is instructive to evolve it on simple synthetically created geometric objects: in this case, black blobs. The overall objective is visual tracking, i.e., following moving objects in an image sequence over time. The algorithm should work on static (one fixed image) as well as dynamic (a movie) image sequences. Figure 19 shows nine frames for the curve evolution on a static blob image. The curve does not settle at the bottom of the potential well of the object (i.e., the edge of the blob object). Instead it oscillates indefinitely. This is sensible behavior. Apart from possible numerical dissipation, the numerical scheme (see Section B.2) is free of dissipational effects. Thus the curve moves back and forth in the potential well without losing energy. Figure 20 shows the simulation results for the same curve evolution with added dissipation (a dissipational term proportional to the curve's velocity will be used for all the following results in this section). The difference is striking. The curve settles quickly at the correct location and then stops moving. Figure 21 shows

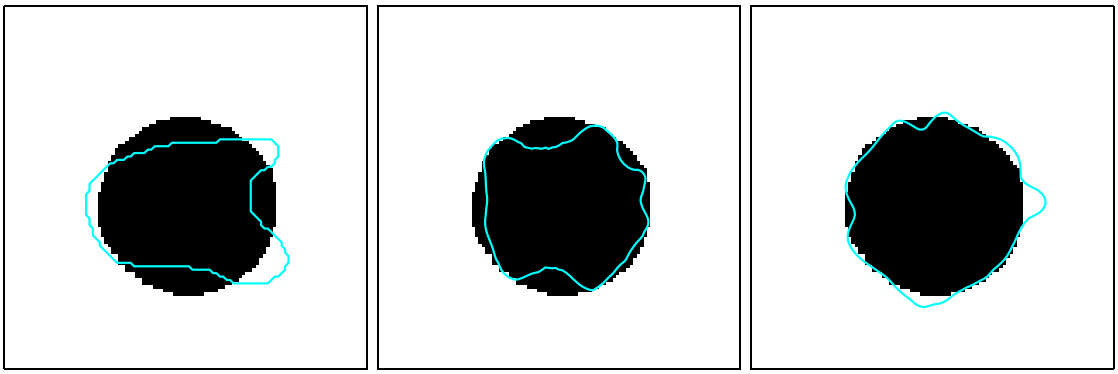
two moving blobs merging and subsequently splitting. The curve evolution scheme manages to keep track of these topological changes. In this case, the potential force is replaced by a gradient vector flow field to ensure that the contour would flow into object concavities[1]. Figure 22 shows four blobs emerging from a single point. This is a difficult example. Even though the curve approximates the outlines of the objects very well at the beginning, the algorithm has no way of "looking inside" the object, where a gap forms. Once the blobs are further apart the curve could move around all four blobs and break into four distinct parts. This would be the desired behavior. However, this is not what happens. Due to the inherent smoothness constraint of the curve evolution algorithm, the curve gets stuck: in between perfectly fulfilling the smoothness constraint (the closer to a circle the better) and obtaining a good approximation to the moving objects.

As demonstrated, the partial level set approach can handle topological changes. To handle the four blob scenario more sophisticated functionals for the potential energy need to be employed. Figure 23 shows the results for the area based approach proposed in Section 3.3. As desired, the evolving curve wraps around all four of the blobs. By using the error injection approach proposed in Section 4.1 undesired oscillations (as for example observed in Figure 19) can be eliminated. Since the error injection approach is based on feature detection along normals to an evolving curve, it positions itself in between purely edge based and purely area based approaches. This is the method employed for real image sequences in Section 6.1.2.

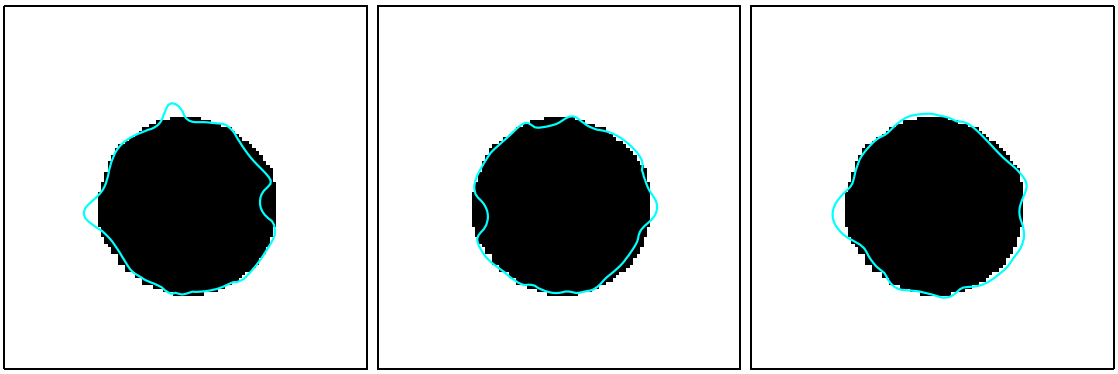
6.1.2 Results on Real Image Sequences

The error injection based algorithm proposed in Section 4.1 is tested on two real video sequences. Figure 25 shows three frames of a fish sequence and Figure 26 shows three frames of a car sequence respectively. In both cases occlusions occur. For the fish sequence no occlusion detection is performed, to demonstrate the behavior of the normal geometric curve evolution algorithm alone, on an image sequence with a short-time partial occlusion.

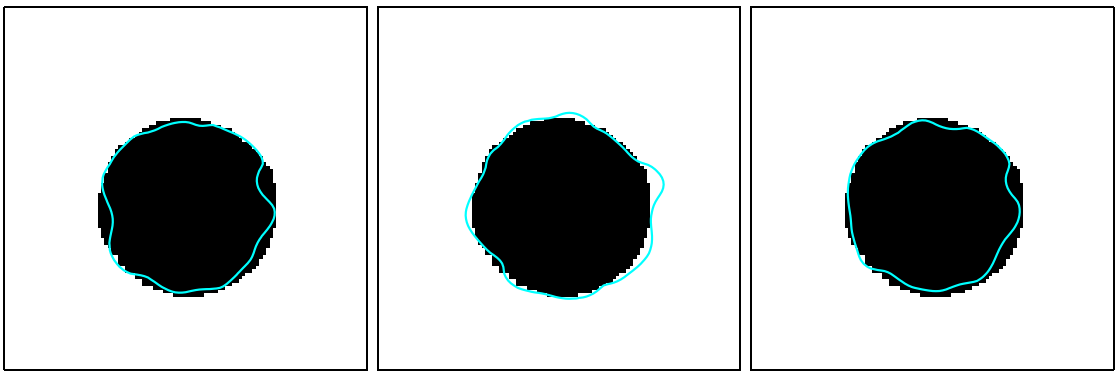
[1] See [142] for details on computing a gradient vector flow field. The gradient vector flow field extends the gradient away from the location of edges into areas of small gradient magnitude, thus enlarging the region of attraction.



(a) Frame 0. (b) Frame 5. (c) Frame 10.

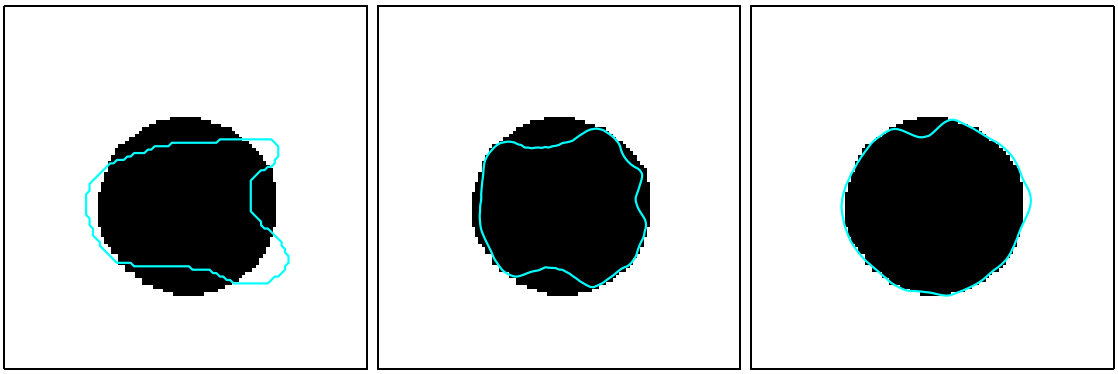


(d) Frame 15. (e) Frame 20. (f) Frame 25.

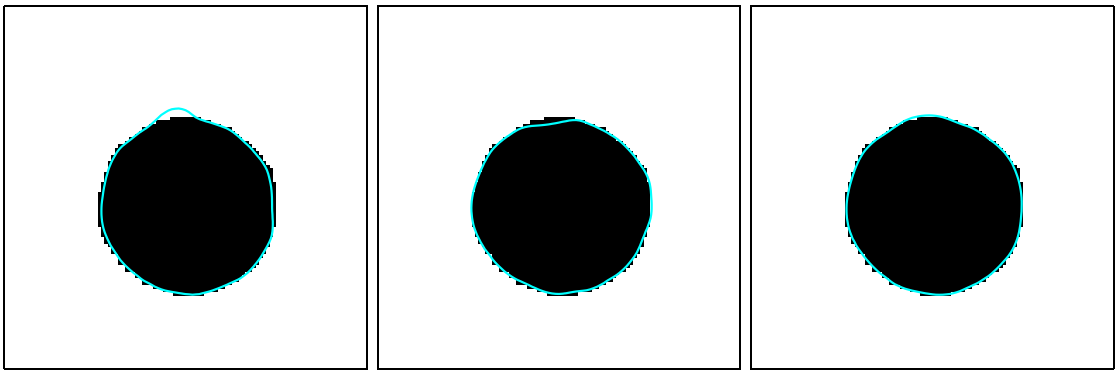


(g) Frame 30. (h) Frame 35. (i) Frame 40.

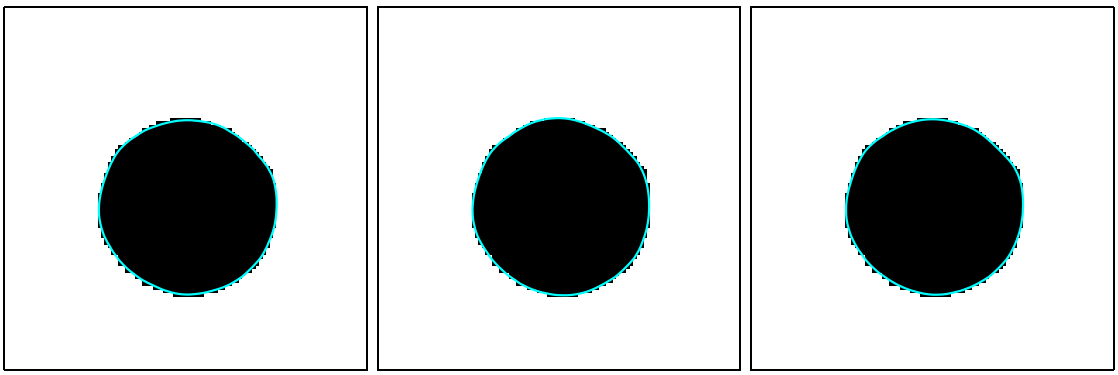
Figure 19: Normal geometric dynamic curve evolution on a static blob image without using dissipation results in oscillations.



(a) Frame 0. (b) Frame 5. (c) Frame 10.

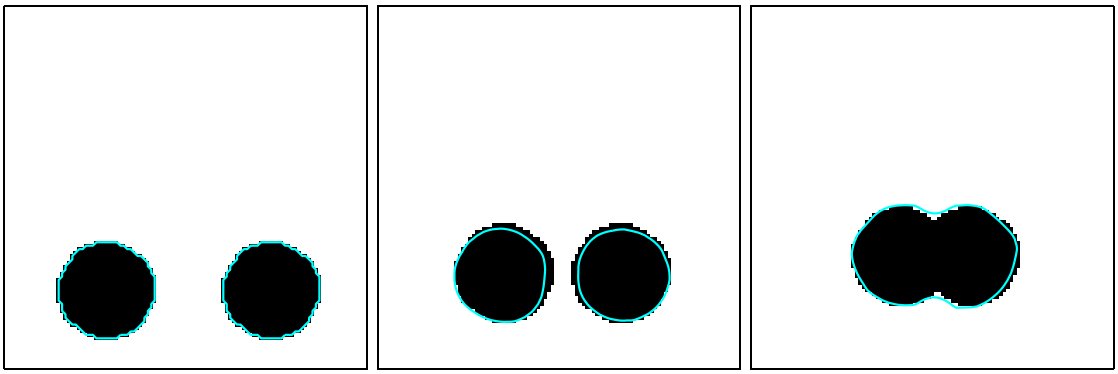


(d) Frame 15. (e) Frame 20. (f) Frame 25.

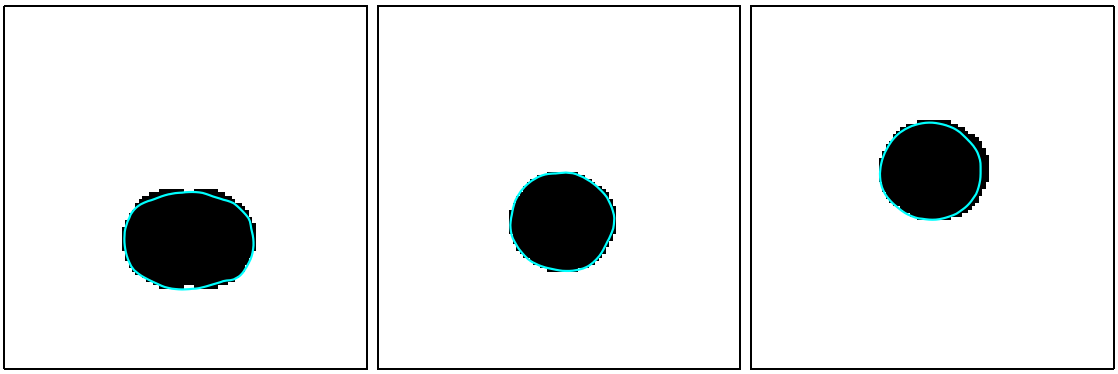


(g) Frame 30. (h) Frame 35. (i) Frame 40.

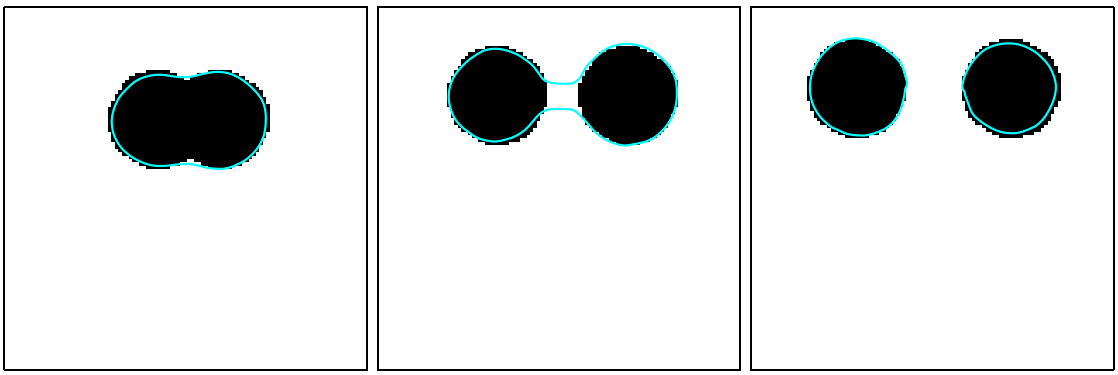
Figure 20: Normal geometric dynamic curve evolution on a static blob image using dissipation prevents prolonged oscillations.



(a) Frame 0. (b) Frame 10. (c) Frame 20.

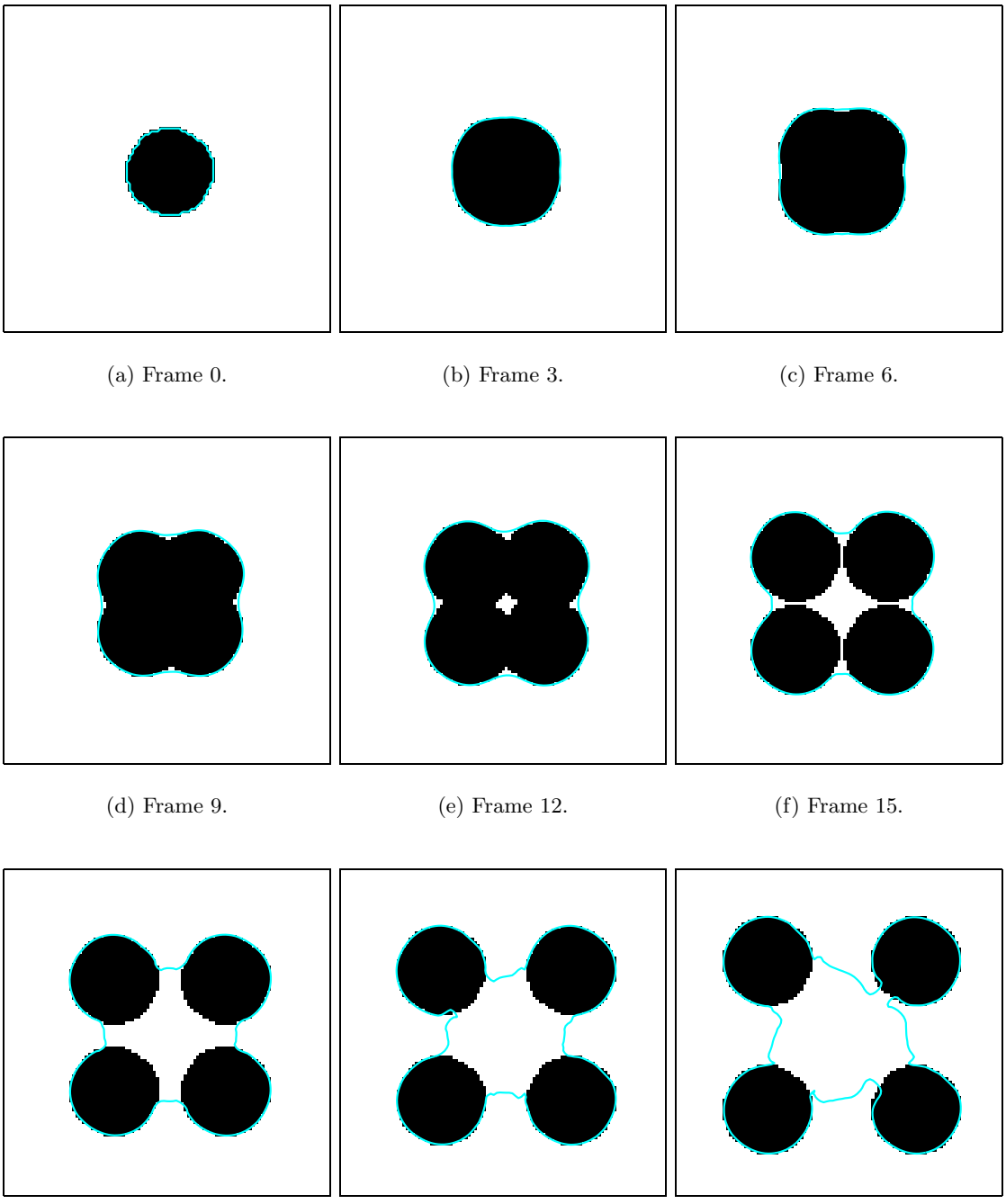


(d) Frame 30. (e) Frame 40. (f) Frame 70.



(g) Frame 100. (h) Frame 115. (i) Frame 119.

Figure 21: The numerical scheme for the partial level set approach allows naturally for merging and splitting of curves.

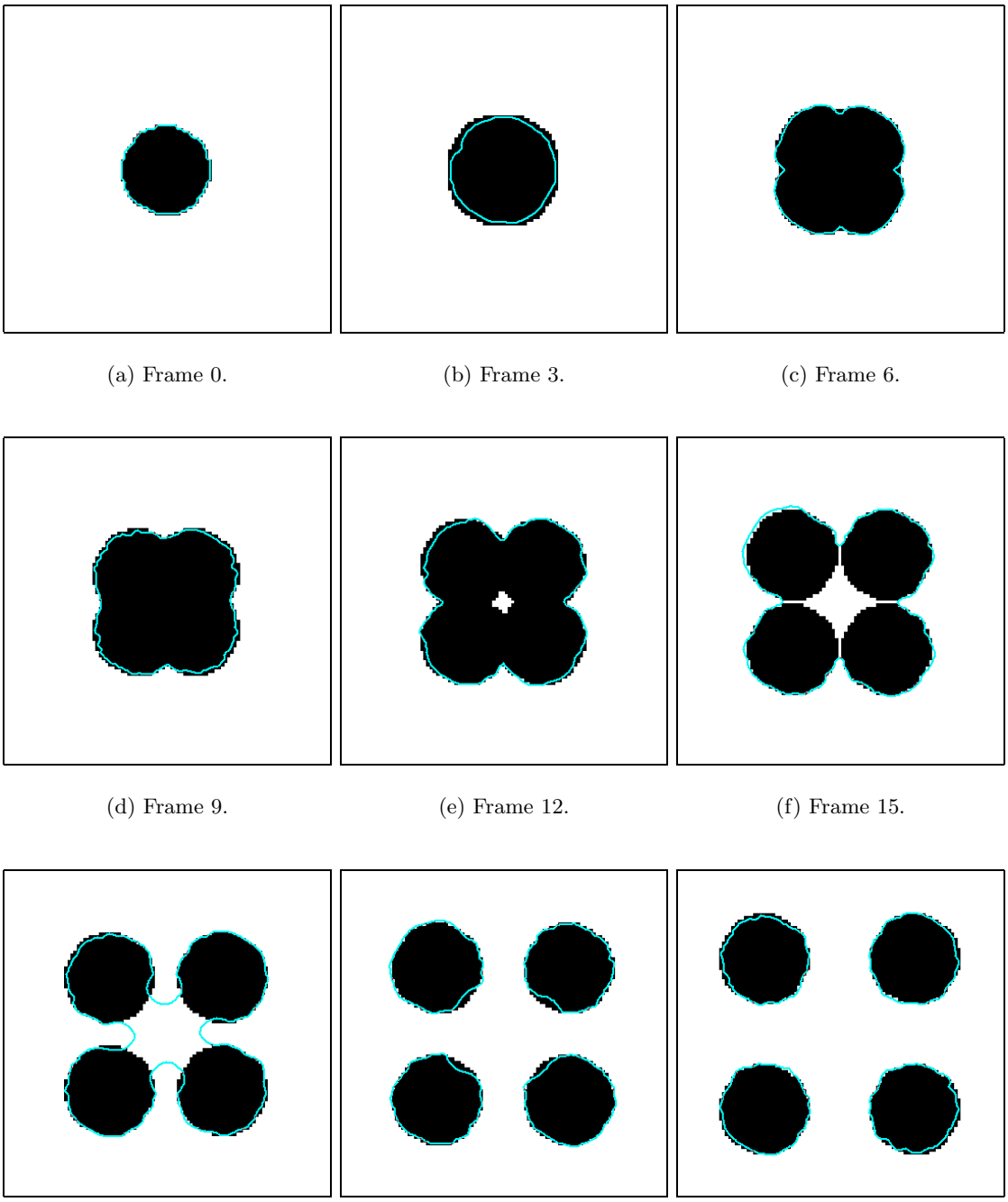


(a) Frame 0. (b) Frame 3. (c) Frame 6.

(d) Frame 9. (e) Frame 12. (f) Frame 15.

(g) Frame 18. (h) Frame 21. (i) Frame 24.

Figure 22: The inherent smoothness constraint of the curve evolution equation does not always allow for the desired segmentation.



(a) Frame 0. (b) Frame 3. (c) Frame 6.

(d) Frame 9. (e) Frame 12. (f) Frame 15.

(g) Frame 18. (h) Frame 21. (i) Frame 24.

Figure 23: The area based evolution captures four blobs moving away from each other.

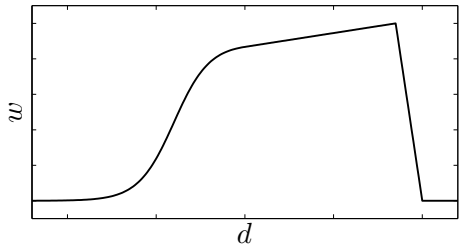


Figure 24: Illustration of the shape of the weighting function $w(\boldsymbol{x})$ for the fish sequence.

Define[2]

$$q(x) := \frac{1}{1+e^{-(p_1+x)}}, \qquad r := q(0) + \frac{e^{-p_1}}{q(0)^2}p_2,$$

$$w(\boldsymbol{x}) := \begin{cases} q(d(\boldsymbol{x})) & \text{if } d(\boldsymbol{x}) \leq 0 \\ q(0) + \frac{e^{-p_1}}{q(0)^2}d(\boldsymbol{x}) & \text{if } 0 < d(\boldsymbol{x}) \leq p_2 \\ r - \frac{r}{p_3-p_2}(d(\boldsymbol{x}) - p_2) & \text{if } p_2 < d(\boldsymbol{x}) \leq p_3 \\ 0 & \text{otherwise.} \end{cases}$$

The used likelihood function for the fish sequence is

$$m(\boldsymbol{z}) = e^{-\left(\frac{(g(\boldsymbol{z})-\mu_g)^2}{2\sigma_g^2} + \frac{(I(\boldsymbol{z})-\mu_I)^2}{2\sigma_I^2}\right)} w(\boldsymbol{z}).$$

The function depends on the image intensity I, the potential function g, and the distance d to the contour. For the car sequence we define

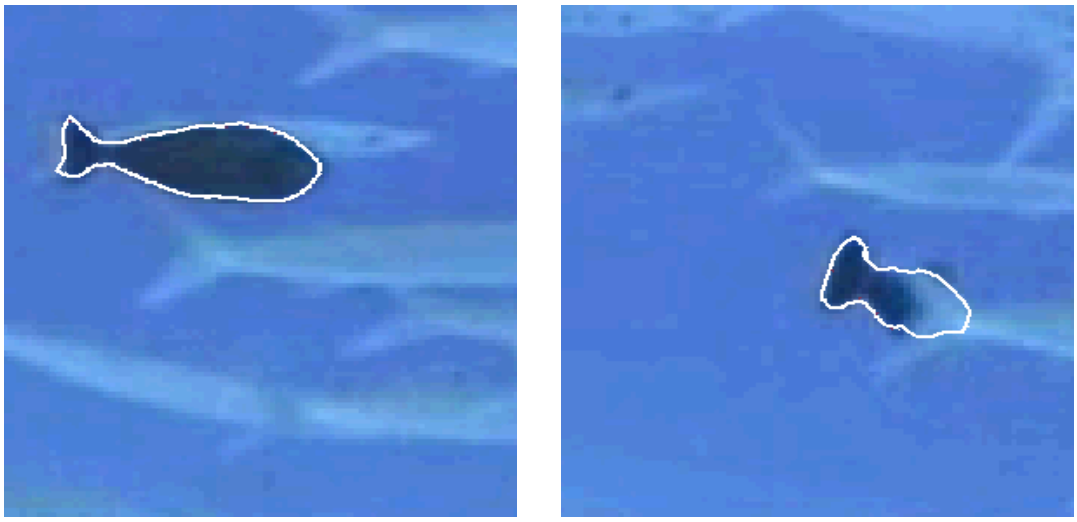
$$\begin{aligned} a(\boldsymbol{x}) &:= \arccos\left(\frac{\nabla(G*I)}{\|\nabla(G*I)\|} \cdot \mathcal{N}\right), \\ a_n(\boldsymbol{x}) &:= \min\left(|a(\boldsymbol{x})|, \pi - |a(\boldsymbol{x})|\right). \end{aligned}$$

This is a measure of angle difference between edge orientation at correspondence points and the normal of the curve. Ideally both should be aligned. The likelihood for a contour point candidate $\boldsymbol{z} \in Z$ is then computed as

$$m(\boldsymbol{z}) = e^{-\left(\frac{(|d(\boldsymbol{z})|-\mu_d)^2}{2\sigma_d^2} + \frac{(g(\boldsymbol{z})-\mu_g)^2}{2\sigma_g^2} + \frac{(a_n(\boldsymbol{z})-\mu_a)^2}{2\sigma_a^2}\right)},$$

and the occlusion detection of Section (4.2) is performed.

[2]This is simply a monotonic function which increases like a sigmoid up to $x = p_1$, linearly increases for $x \in (p_1, p_2]$, linearly decreases to zero for $x \in (p_2, p_3]$ and is zero everywhere else. See Figure 24 for an illustration.



(a) Frame 0

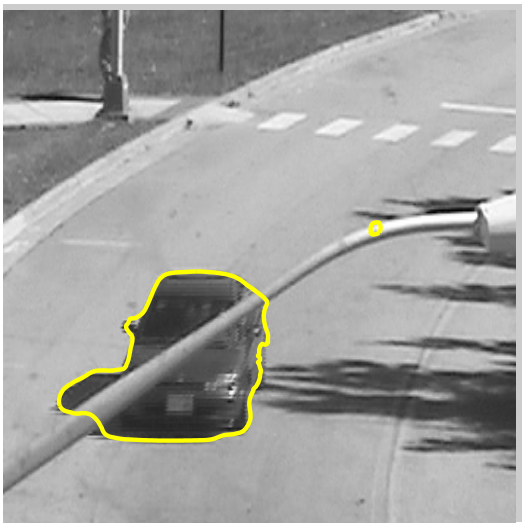
(b) Frame 80

(c) Frame 90

Figure 25: Three frames of a fish sequence.

(a) Frame 0

(b) Frame 14



(c) Frame 55

Figure 26: Three frames of a car sequence.

In both cases occlusions are handled. For the fish sequence the occlusion is dealt with implicitly. The occluding fish moves over the tracked fish quickly, so that the inertia effects keep the fish at a reasonable location. For the car example the occlusion (the lamp post) is treated explicitly by means of the proposed occlusion detection algorithm. In both cases the likelihood functions do not incorporate any type of prior movement or shape information. Doing so would increase robustness, but limit flexibility. Finally, since this active contour model is based on edge features, the active contour captures the sharp edge of the shadow in the car sequence. Presumably this could be handled by including more global area-based terms or shape information in the model.

6.2 Results for the Full Level Set Approach

The results for the partial level set based method presented in Section 6.1 demonstrate the behavior of a normal geometric dynamic active contour on synthetic and real image sequences. A partial level set approach cannot represent intersecting curves in the image plane. Topological merging or splitting is only performed based on the geometrical shape of the represented curves in the image plane. The full level set aims at altering the behavior for topological changes. In the context of dynamic active contours it should allow for intersecting curves in the image plane and should allow for curve merging taking into account the curves' positions in the image plane and their additional states (e.g., velocity information). This section illustrates this behavior for normal geometric dynamic curve evolution (see Section 5.4) implemented using the vector distance function based full level set approach on a simple synthetic image. Furthermore, simple numerical test cases are presented to illustrate the behavior of vector distance function evolutions (see Sections 5.2 and 5.4 for the theoretical background).

Figure 27 shows a one dimensional toy problem, where an implicit representation of a point gets moved back and forth. It demonstrates the superiority of the vector distance function approach 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

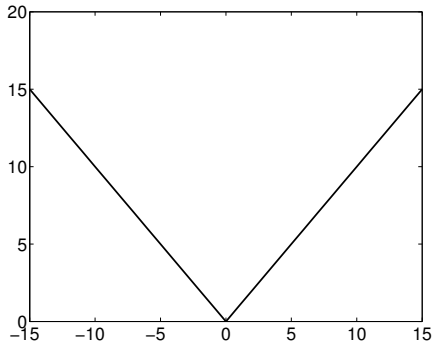
without error compensation). On the other hand error compensation does not significantly improve the result for the vector distance function evolution in this example. In both cases (with or without error compensation) the vector distance function scheme accurately follows the desired movement.

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.

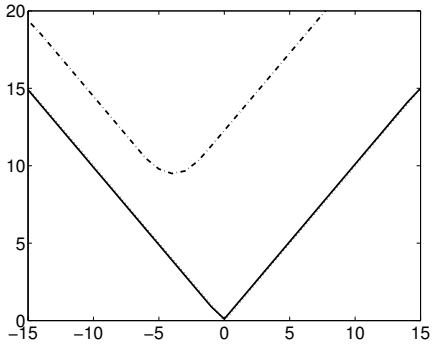
Figure 28 shows a two-dimensional result for a vector distance function based curvature flow. This is merely a result to demonstrate the feasibility of vector distance function evolutions, since this problem could be solved by a standard level set approach using a signed distance function. The results show a nice regularization and the expected behavior (the object becomes more and more circular over time). Finally, Figures 29 and 30 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 31-32 show a circular object oscillating in a potential well based on the evolution equations derived in Section 5.4.1. 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.

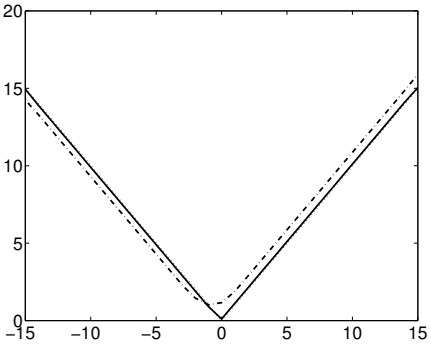
To test the behavior for topological changes, Figures 33-35 show two circles oscillating simultaneously in the same potential well used previously. They slide over each other while their velocities are significantly different from each other, but merge once their velocities become numerically indistinguishable. Note that this is a demanding example since we chose



(a) Initial curve.

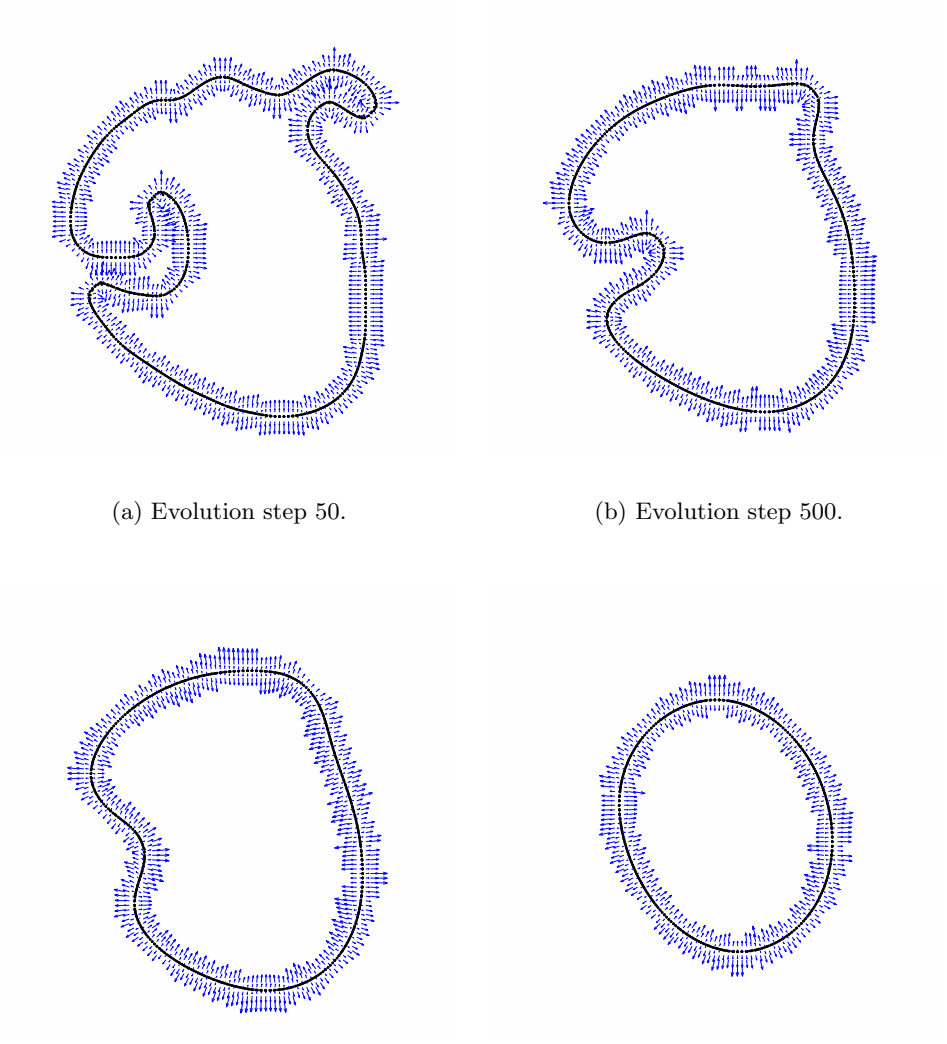


(b) Evolution without error compensation.



(c) Evolution with error compensation.

Figure 27: One-dimensional distance function and vector distance function evolutions. Dash-dotted curves are the results for the distance function evolution. The dotted curve is the initial curve for comparison and the solid curve is the result for the vector distance function evolution. The results represent evolutions for 400 timesteps of 0.1. The evolutions were performed with a constant velocity field of 1 which flips sign every 100 iterations.



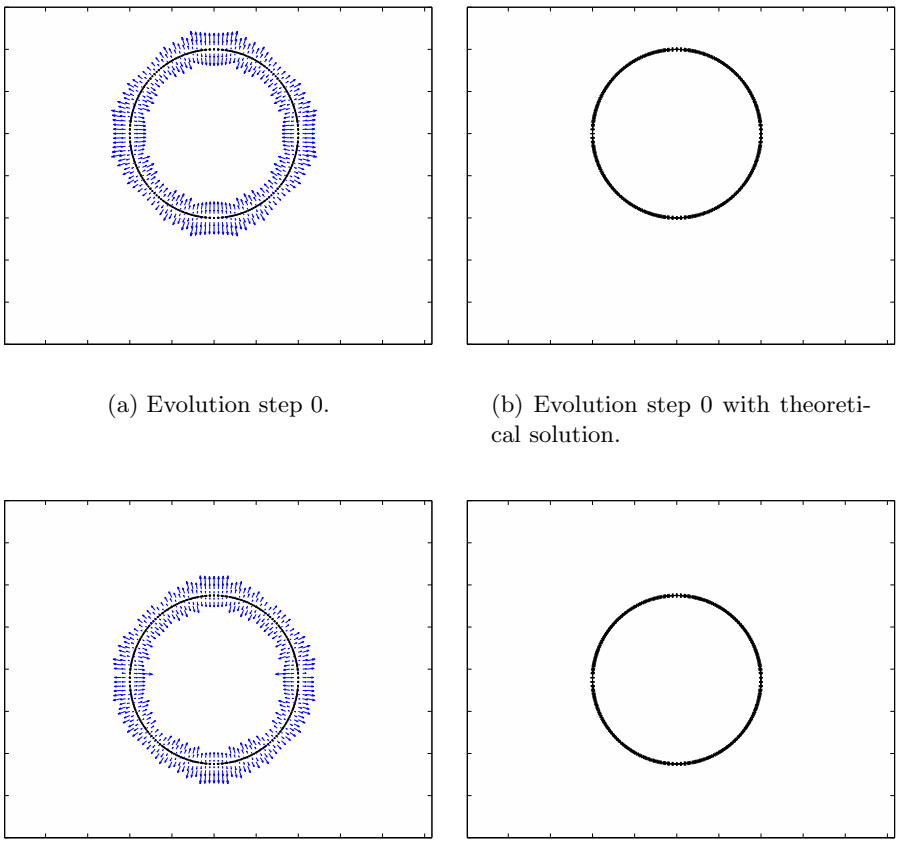
(a) Evolution step 50.

(b) Evolution step 500.

(c) Evolution step 1000.

(d) Evolution step 4000.

Figure 28: Vector distance function based curvature flow.



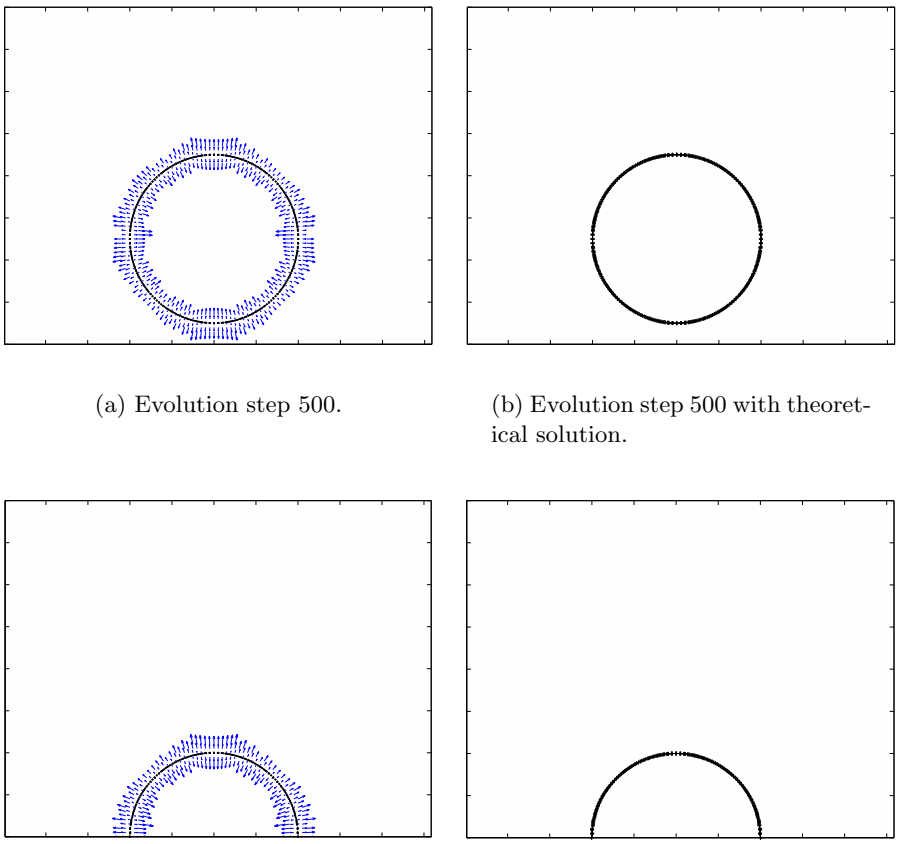
(a) Evolution step 0.

(b) Evolution step 0 with theoretical solution.

(c) Evolution step 250.

(d) Evolution step 250 with theoretical solution.

Figure 29: Circular object subject to a uniform flow field with theoretical solution. Vector distance function evolution steps 0 and 250.



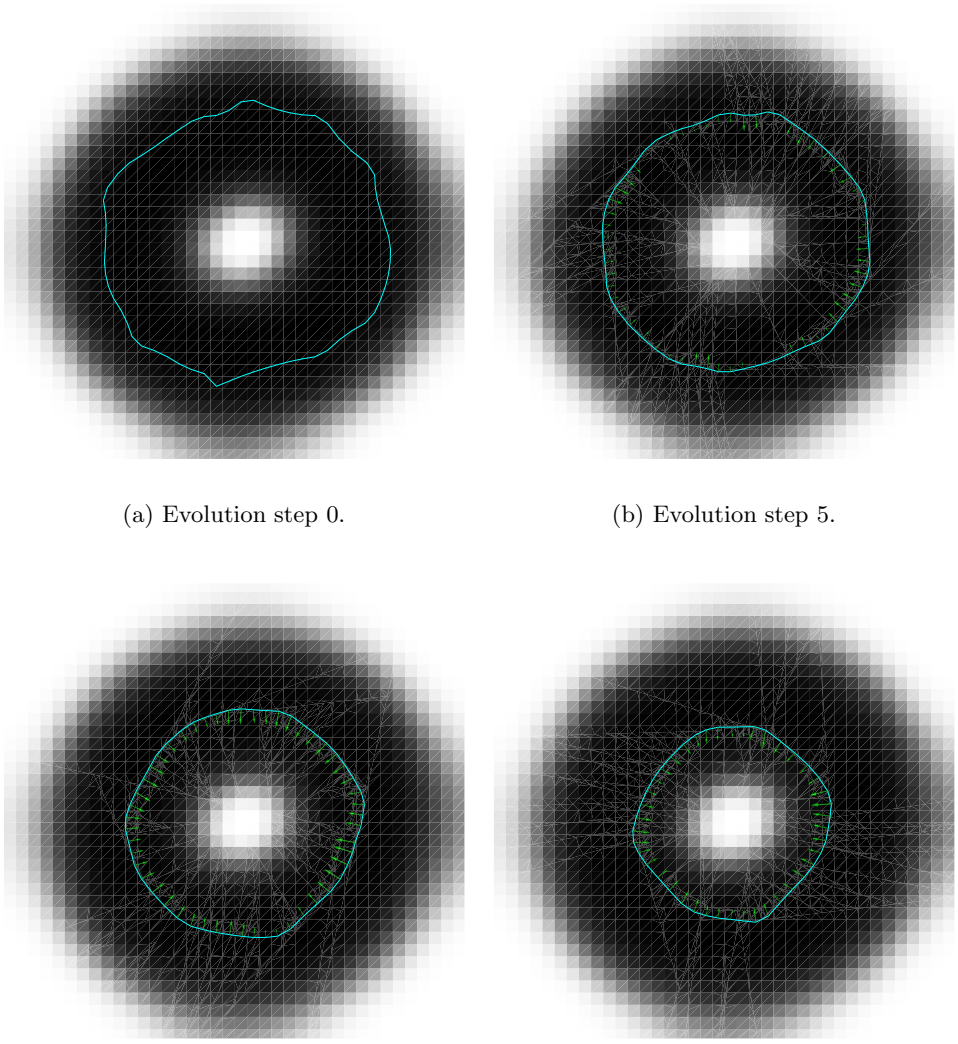
(a) Evolution step 500.

(b) Evolution step 500 with theoretical solution.

(c) Evolution step 1000.

(d) Evolution step 1000 with theoretical solution.

Figure 30: Circular object subject to a uniform flow field with theoretical solution. Vector distance function evolution steps 500 and 1000.



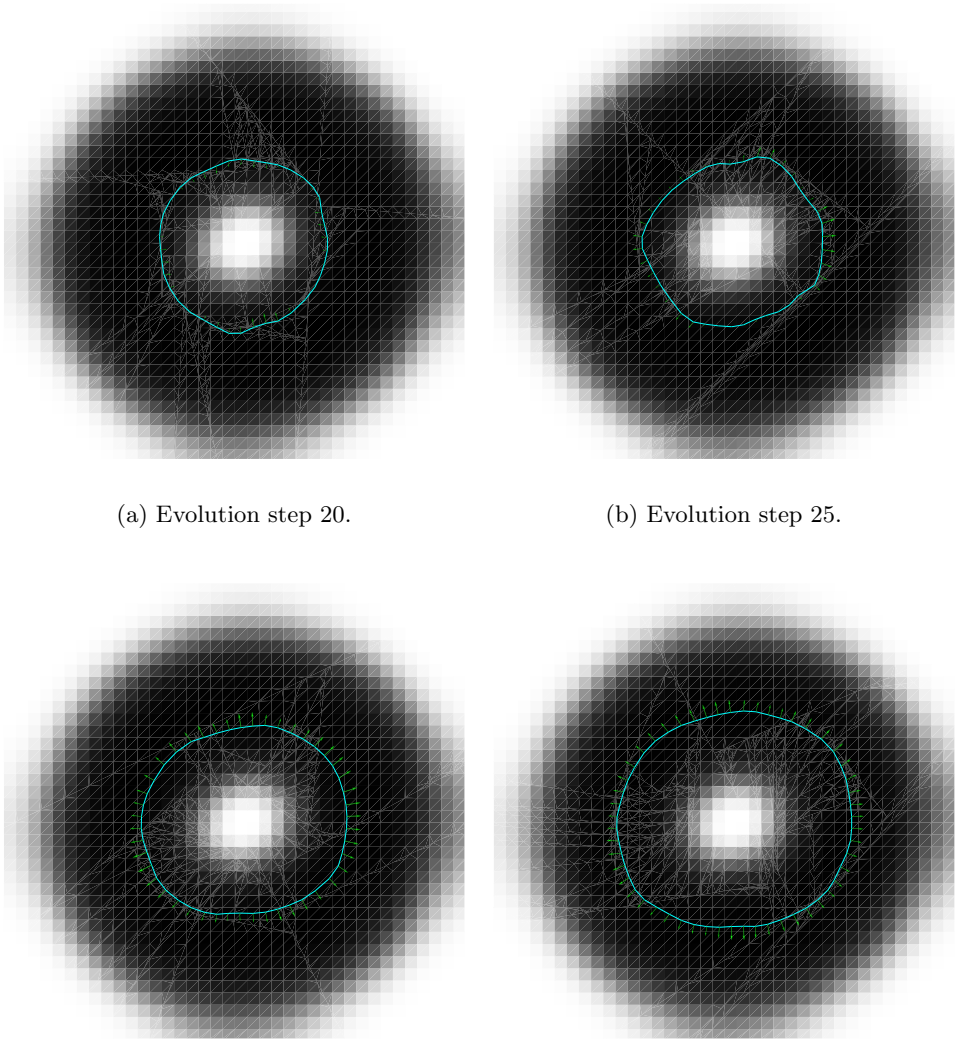
(a) Evolution step 0.

(b) Evolution step 5.

(c) Evolution step 10.

(d) Evolution step 15.

Figure 31: Normal geometric dynamic curve evolution using a vector distance function approach showing an oscillating circular curve. Evolution steps 0-15.



(a) Evolution step 20.

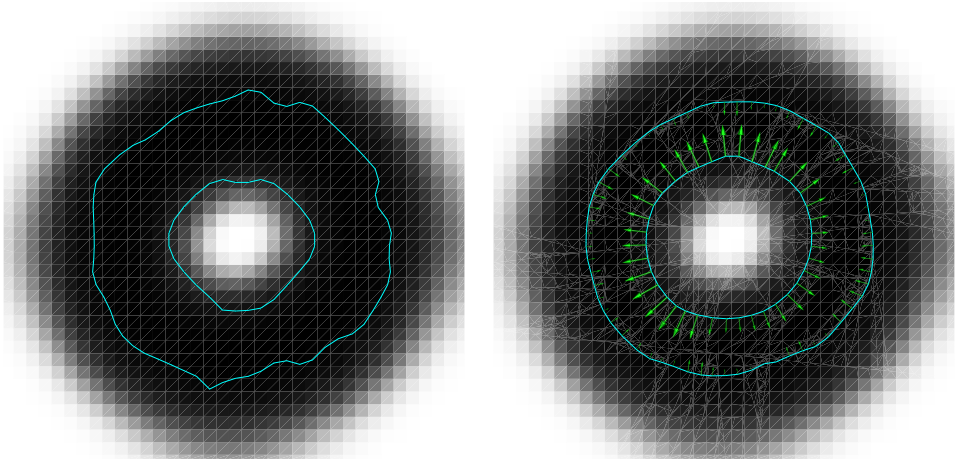
(b) Evolution step 25.

(c) Evolution step 30.

(d) Evolution step 35.

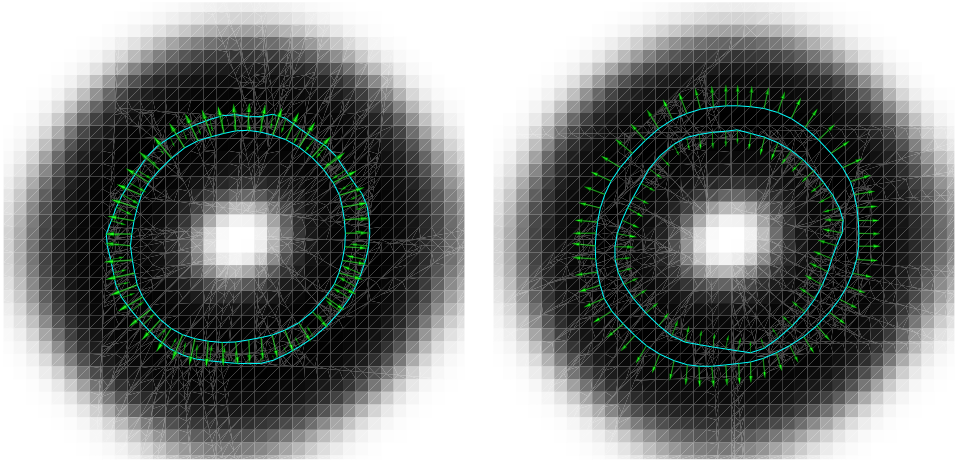
Figure 32: Normal geometric dynamic curve evolution using a vector distance function approach showing an oscillating circular curve. Evolution steps 20-35.

to discretize the velocity with the same accuracy as the spatial dimension (see the pixel size in the images). To illustrate this, Figure 36 shows the three-dimensional projections for a representative thinned discrete approximation to the zero level set.



(a) Evolution time 0.

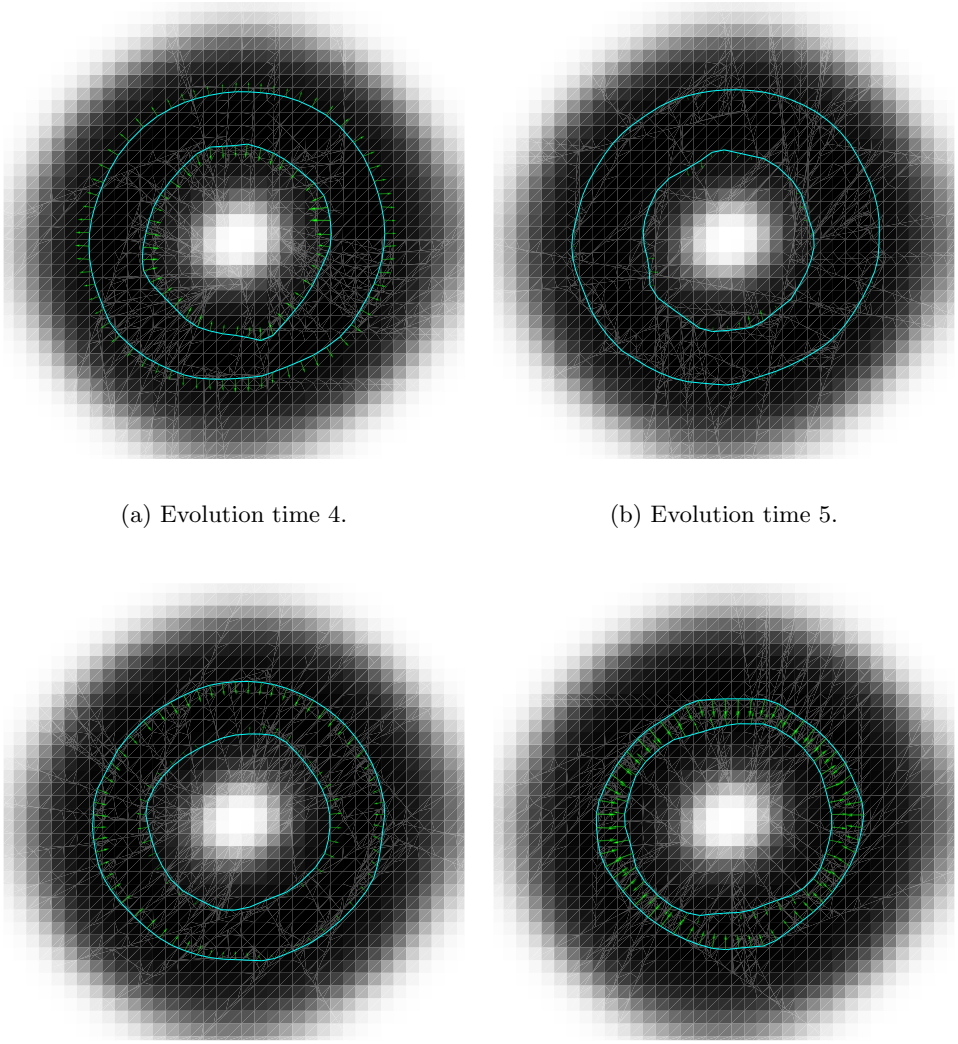
(b) Evolution time 1.



(c) Evolution time 2.

(d) Evolution time 3.

Figure 33: Normal geometric dynamic curve evolution using a vector distance function approach showing two oscillating circles. Time steps 0 to 3.



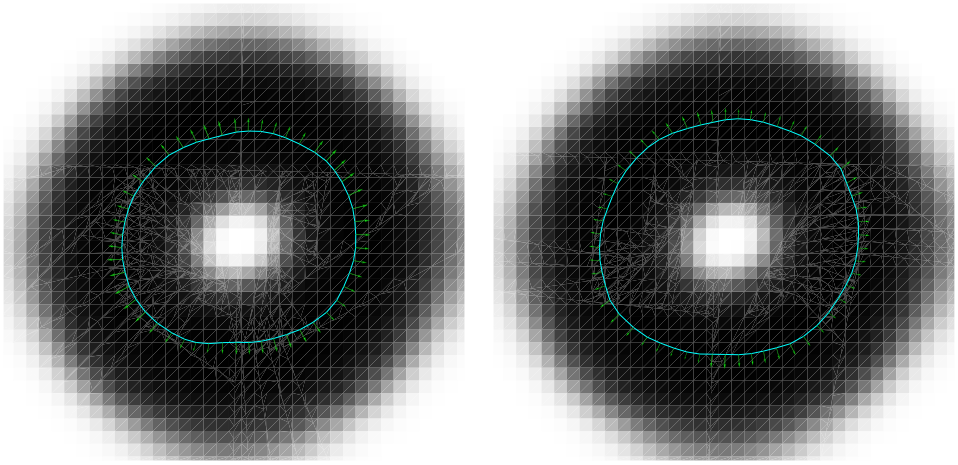
(a) Evolution time 4.

(b) Evolution time 5.

(c) Evolution time 6.

(d) Evolution time 7.

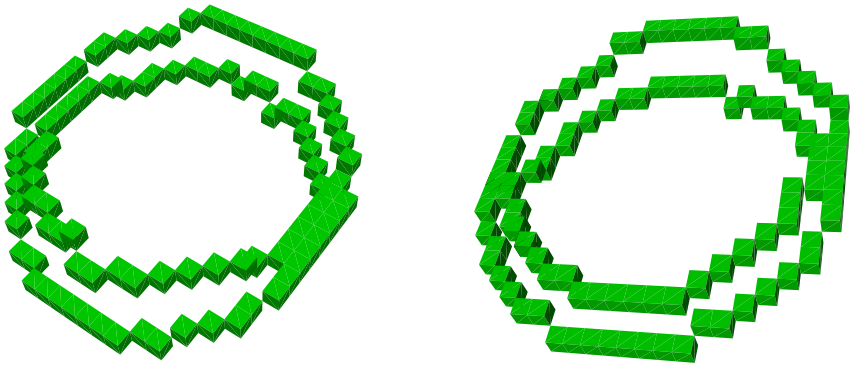
Figure 34: Normal geometric dynamic curve evolution using a vector distance function approach showing two oscillating circles. Time steps 4 to 7.



(a) Evolution time 8.

(b) Evolution time 9.

Figure 35: Normal geometric dynamic curve evolution using a vector distance function approach showing two oscillating circles. Time steps 8 to 9.



(a) Projection in the u direction at evolution time 3.

(b) Projection in the v direction at evolution time 3.

Figure 36: Two projections of the thinned discrete zero level set representation for the vector distance function based normal geometric dynamic curve evolution.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

The contribution of this thesis is twofold: The active contour approach was extended to include dynamics in the context of visual tracking and level set methods were developed specifically for the resulting evolution equations.

Compared to traditional active contour methods, the approach developed in this thesis incorporates the propagation of state information with every point on the evolving contour into the evolution process. For the advocated normal dynamic geodesic active contour this is the normal velocity, but any other kind of state information can be treated in a similar way. In the context of visual tracking this facilitates a natural framework to combine velocity and position estimation within a level set approach.

The normal dynamic geometric active contour relates to dynamic snakes [129] as geodesic or conformal active contours [75, 22] relate to the original snake formulation [74], advocating a different philosophy to dynamic curve evolution. Instead of discretizing evolution equations upfront (early lumping), partial differential equations describe the system as long as possible (late lumping [141]), resulting in a more natural and geometric formulation. Thus level set formulations become feasible. Due to the dynamic nature of the evolution equations, the application of the classical level set methodology is not straightforward. This thesis presented two different level set frameworks for the evolution of the normal dynamic geometric active contour: the partial and the full level set approaches. Both methods were implemented and their usefulness was demonstrated on a variety of real and artificial image sequences.

To increase the robustness an observer-like system was proposed using error injection to drive an evolving curve towards a desired state. Specifically, our proposed algorithm

searches for image features (i.e., likelihood maxima) along normal lines of an evolving contour, resulting in an algorithm that lies between purely edge-based and purely area-based approaches.

The partial level set approach was designed to propagate the geometrical curve shape in the image plane by means of a level set function. Any any additional information gets propagated by transport equations. Thus the classical level set approach [115] can be applied, but topological changes are restricted to topological changes set forth by the geometry of curves in the image plane. In particular, curve intersections cannot be handled in this framework.

The full level set approach is built upon level set methods capable of evolving implicit descriptions of objects of codimension larger than one. State and geometry information gets propagated in one coherent framework. There is no separation between the description of geometrical shape in the plane and additional information propagated with points on a curve. In particular, the full level set approach can handle (as demonstrated) intersections of curves in the image plane, thus potentially allowing for curves sliding over each other while being visually tracked.

For the full level set approach we employed a vector distance function based level set evolution, which is a useful tool to implement evolutions of objects of codimension greater than one. It allows for a narrow-banding scheme, reducing the computational complexity significantly. This is of great importance for evolutions in high dimensional spaces. To address implementation issues, we successfully proposed reducing the class of objects to be represented, to the class of closed curves. This greatly simplifies the required explicit reconstruction of the zero level set for reinitialization purposes, which is not straightforward in general, but then becomes tractable. The resulting evolution is a combination of an implicit scheme (the vector distance function based level set propagation) with occasional explicit reinitializations of the vector distance function. These explicit reinitializations are based on discrete geometry and the recent theory of computational homology. Inadvertently, this thesis proposed and proved a characterization of simple points in arbitrary (finite) dimensions based on the theory of cubical homology. While a completely implicit scheme

would be very desirable, also with regard to the representation and evolution of a less restrictive object class, the reconstruction of an explicit representation of the zero level set has the advantage of treating the fattening problem as a side-effect, for free. Topological mergers can then be handled (as demonstrated) satisfactorily.

The level set methods used for the partial and the full level set approaches are interesting in their own right and can be applied to a very general class of evolution equations. However, a better mathematical understanding of the vector distance function evolution scheme is highly desirable. Numerical methods tailored specifically for 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.

The proposed visual tracking approach has the potential to deal with partial occlusions. However, at the present stage it still lacks robustness. Generally, edge-based approaches trade off robustness for versatility. If strong, clear edge information exists they are a very useful class of algorithms; however, a straightforward edge-based approach will most likely suffer in performance for noisy or cluttered images. To increase robustness, shape information [82, 24, 23, 34, 32, 33, 112, 133] or area-based terms should be introduced into the formulation. As an alternative to including dynamic area terms the image sequence could be preprocessed (e.g., by some static segmentation step on every image frame).

Being able to transport additional state information is useful beyond the application of visual tracking. Any kind of information can potentially be propagated along with a curve. For example marker particles, enabling us to follow the movement of specific curve parts over time. This property results in increased flexibility and enables fundamentally different curve behaviors than in the static (i.e., non-dynamic) curve evolution case.

Furthermore, applications for dynamic active contours need not be restricted to tracking; e.g., applications in computer graphics are conceivable (where the curve movement would then be physically motivated), not necessarily involving an underlying real image (e.g., we could design artificial potential fields enforcing a desired type of movement). As geodesic

active contours extend to geodesic active surfaces, dynamic geodesic snakes can be extended to dynamic geodesic surfaces. Of interest are also area based dynamic evolutions and evolutions evolving tangential velocities which should be explored in future work.

APPENDIX A

DERIVATION OF THE EVOLUTION EQUATIONS

This chapter presents detailed derivations for the evolution equations associated with geometric dynamic curve evolution, area based dynamic curve evolution, and normal geometric dynamic curve evolution.

A.1 Geometric Dynamic Curve Evolution

We consider the evolution of closed curves of the form $\mathcal{C} : S^1 \times [0, \tau) \mapsto \mathbb{R}^2$ in the plane. Where $\mathcal{C} = \mathcal{C}(p, t)$ and $\mathcal{C}(0, t) = \mathcal{C}(1, t)$ [128, 76], with t being the time, and $p \in [0, 1]$ the curve's parameterization. The classical formulation for dynamic curve evolution as proposed by Terzopoulos and Szeliski [129] is derived by means of minimization of the *action integral*

$$\mathcal{L} = \int_{t=t_0}^{t_1} \int_{p=0}^{1} L(t, p, \mathcal{C}, \mathcal{C}_p, \mathcal{C}_{pp}, \mathcal{C}_t)\ dp\ dt, \tag{72}$$

where the subscripts denote partial derivatives with respect to the time t and the parameterization p. The Lagrangian $L = T - U$ is the difference between the kinetic and the potential energy. Minimizing equation (72) using the Lagrangian

$$L = \left(\frac{1}{2} \mu \|\mathcal{C}_t\|^2 - g \right) \|\mathcal{C}_p\|,$$

where $g > 0$ is a potential function (with the desired location of the curve forming a potential well) yields

$$\mu \mathcal{C}_{tt} + \frac{\partial}{\partial s} \left(\left(\frac{1}{2} \mu \|\mathcal{C}_t\|^2 - g \right) \mathcal{C}_s \right) + \mu \left(\mathcal{T} \cdot \mathcal{C}_{ts} \right) \mathcal{C}_t + \nabla g = 0. \tag{73}$$

which is intrinsic. Here $\mathcal{N}$ is the outward pointing unit normal, and $\mathcal{T} = \frac{\partial \mathcal{C}}{\partial s}$ the unit tangent vector to the curve. $\kappa = \mathcal{C}_{ss} \cdot \mathcal{N}$ denotes curvature and s is arclength [38, 104].

Assume the curve $\mathcal{C}$ gets perturbed by $\epsilon \mathcal{V}$ yielding the curve

$$\mathcal{C}^p = \mathcal{C} + \epsilon \mathcal{V}.$$

The action integral (72) then becomes

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

Computing the Gâteaux-Variation by taking the derivative with respect to ϵ for $\epsilon = 0$ yields:

$$\delta\mathcal{L}(\mathcal{C};\mathcal{V}) = \frac{\partial\mathcal{L}}{\partial\epsilon}|_{\epsilon=0} = \int_{t=t_0}^{t_1}\int_{p=0}^{1}\left(\mu\mathcal{C}_t\cdot\mathcal{V}_t - \nabla g\cdot\mathcal{V}\right)\|\mathcal{C}_p\| + \left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)\frac{1}{\|\mathcal{C}_p\|}\mathcal{C}_p\cdot\mathcal{V}_p\ dp\ dt.$$

Assuming μ to be constant integration by parts results in

$$\delta\mathcal{L}(\mathcal{C};\mathcal{V}) = \int_{t=t_0}^{t_1}\int_{p=0}^{1} -\frac{\partial}{\partial p}\left(\left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)\mathcal{T}\right)\cdot\mathcal{V}\frac{1}{\|\mathcal{C}_p\|}\|\mathcal{C}_p\| - \\ - \nabla g\cdot\mathcal{V}\|\mathcal{C}_p\| - \frac{\partial}{\partial t}\left(\|\mathcal{C}_p\|\mu\mathcal{C}_t\right)\cdot\mathcal{V}\frac{1}{\|\mathcal{C}_p\|}\|\mathcal{C}_p\|\ dp\ dt. \quad (74)$$

The boundary terms occurring from the integrations by parts drop out since the curves are closed. Then (since (74) has to be fulfilled for any $\mathcal{V}$)

$$\frac{\partial}{\partial s}\left(\left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)\mathcal{T}\right) + \nabla g + \frac{\partial}{\partial t}\left(\|\mathcal{C}_p\|\mu\mathcal{C}_t\right)\frac{1}{\|\mathcal{C}_p\|} = 0, \quad (75)$$

where we made use of the fact that

$$\frac{\partial}{\partial s} = \frac{1}{\|\mathcal{C}_p\|}\frac{\partial}{\partial p}.$$

To simplify Equation (75) the following correspondences are useful:

$$\begin{aligned}
\mathcal{C}_{pt} &= \mathcal{C}_{tp},\\
\frac{1}{\|\mathcal{C}_p\|}\frac{\partial}{\partial t}\|\mathcal{C}_p\| &= \frac{1}{2\|\mathcal{C}_p\|^2}2\mathcal{C}_p\cdot\mathcal{C}_{pt} = \frac{\mathcal{C}_p}{\|\mathcal{C}_p\|^2}\cdot\mathcal{C}_{tp} = \mathcal{C}_s\cdot\mathcal{C}_{ts} = \mathcal{T}\cdot\mathcal{C}_{ts},\\
\frac{\partial}{\partial s}\mathcal{T} &= \kappa\mathcal{N},\\
\frac{\partial}{\partial s}\|\mathcal{C}_t\|^2 = &= \frac{1}{\|\mathcal{C}_p\|}\frac{\partial}{\partial p}\left(\|\mathcal{C}_t\|^2\right) = \frac{1}{\|\mathcal{C}_p\|}2\mathcal{C}_t\cdot\mathcal{C}_{tp} = 2\mathcal{C}_t\cdot\mathcal{C}_{ts},\\
\frac{\partial}{\partial s}g &= \nabla g\cdot\mathcal{T},\\
\frac{\partial}{\partial t}\mathcal{C}_t &= \mathcal{C}_{tt}.
\end{aligned}$$

Specifically it follows that

$$\begin{aligned}
\frac{\partial}{\partial s}\left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right) &= \mu\mathcal{C}_t\cdot\mathcal{C}_{ts} - \nabla g\cdot\mathcal{T},\\
\frac{1}{\|\mathcal{C}_p\|}\frac{\partial}{\partial t}\left(\|\mathcal{C}_p\|\mu\mathcal{C}_t\right) &= \mu\mathcal{C}_{tt} + \mu\left(\mathcal{T}\cdot\mathcal{C}_{ts}\right)\mathcal{C}_t.
\end{aligned}$$

Plugging everything in (75) yields

$$\mu\mathcal{C}_{tt} + \left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)\kappa\mathcal{N} + \frac{\partial}{\partial s}\left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)\mathcal{T} + \mu\left(\mathcal{T}\cdot\mathcal{C}_{ts}\right)\mathcal{C}_t + \nabla g = 0, \tag{76}$$

which is Equation (73). Expanding $\frac{\partial}{\partial s}\left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)$, Equation (76) can be written as

$$\mu\mathcal{C}_{tt} = -\left(\frac{1}{2}\mu\|\mathcal{C}_t\|^2 - g\right)\kappa\mathcal{N} - \left(\nabla g\cdot\mathcal{N}\right)\mathcal{N} - \mu\left(\mathcal{T}\cdot\mathcal{C}_{ts}\right)\mathcal{C}_t - \mu\left(\mathcal{C}_t\cdot\mathcal{C}_{ts}\right)\mathcal{T}. \tag{77}$$

A.2 Area Based Dynamic Curve Evolution

We need to find the first variation of

$$S = \int_R I\ dA.$$

From the divergence theorem follows that

$$\int_{\mathcal{C}} \boldsymbol{F}\cdot\boldsymbol{N}\ ds = \int_R \nabla\cdot\boldsymbol{F}\ dx\ dy = \int_R f\ dx\ dy,$$

where $\boldsymbol{N} = -\mathcal{N}$ is the unit outward normal, $\boldsymbol{F}$ is a vector field and R is the interior of the curve $\mathcal{C}$. Assume the energy to minimize is

$$E(\mathcal{C}) = \int_{\mathcal{C}} \boldsymbol{F}\cdot\boldsymbol{N}\ ds.$$

We know that

$$E(\mathcal{C}) = \int_0^1 \boldsymbol{F}\cdot\boldsymbol{N}\|\mathcal{C}_p\|\ dp = \int_0^1 \boldsymbol{F}\cdot J\mathcal{C}_p\ dp,$$

where we made use of the fact that

$$\boldsymbol{N} = J\mathcal{T} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\frac{\mathcal{C}_p}{\|\mathcal{C}_p\|}.$$

Perturbing the curve C by $\epsilon\mathcal{V}$ yields

$$E(\mathcal{C}+\epsilon\mathcal{V}) = \int_0^1 \boldsymbol{F}(\mathcal{C}+\epsilon\mathcal{V})\cdot J(\mathcal{C}_p+\epsilon\mathcal{V}_p)\ dp.$$

Then the first variation is

$$\delta E = \frac{\partial E}{\partial\epsilon}\Big|_{\epsilon=0}.$$

We obtain

$$\begin{aligned}\delta E &= \int_0^1 \nabla \boldsymbol{F} \mathcal{V} J \mathcal{C}_p + \boldsymbol{F} J \mathcal{V}_p \; dp \\ &= \int_0^1 \nabla \boldsymbol{F} \mathcal{V} J \mathcal{C}_p - \left(\frac{\partial}{\partial p} (\boldsymbol{F}^T J) \right) \mathcal{V} \; dp \\ &= \int_0^1 \mathcal{C}_p^T \left(J^T \nabla \boldsymbol{F} - (\nabla \boldsymbol{F})^T J \right) \mathcal{V} \; dp.\end{aligned}$$

Define

$$\phi J := J^T \nabla \boldsymbol{F} - (\nabla \boldsymbol{F})^T J, \tag{78}$$

then

$$\begin{aligned}\delta E &= \int_0^1 (\phi J \mathcal{T})^T \mathcal{V} \|\mathcal{C}_p\| \; dp \\ &= \int_{\mathcal{C}} < \phi \boldsymbol{N}, \mathcal{V} > \; ds.\end{aligned}$$

By setting

$$\boldsymbol{F} = \begin{pmatrix} F^x \\ F^y \end{pmatrix},$$

we obtain after substitution into Equation (78)

$$\phi J = -(F_x^x + F_y^y) J = -\nabla \cdot \boldsymbol{F} J,$$

and thus

$$\phi = -\nabla \cdot \boldsymbol{F}.$$

Then it follows that

$$\delta E = \int_{\mathcal{C}} < -f \boldsymbol{N}, \mathcal{V} > \; ds = \int_{\mathcal{C}} < f \mathcal{N}, \mathcal{V} > \; ds, \tag{79}$$

where $\mathcal{N}$ is as usual the unit inward normal.

Define

$$g_a(u, w) = -\frac{1}{2} (u - w)^2,$$

where $u = S_u / A_u$ and $w = S_w / A_w$, where

$$\begin{aligned} S_u &= \int_{R^u} I \; dA \qquad & A_u &= \int_{R^u} dA \\ S_w &= \int_{R^w} I \; dA \qquad & A_w &= \int_{R^w} dA, \end{aligned}$$

and R^u and R^w denote the domains inside and outside the curve respectively. The gradients (using Equation (79)) evaluate to

$$\nabla S_u = I\mathcal{N} \qquad \nabla A_u = \mathcal{N}$$
$$\nabla S_v = -I\mathcal{N} \qquad \nabla A_v = -\mathcal{N}.$$

With this follows

$$\nabla u = \frac{A_u \nabla S_u - S_u \nabla A_u}{A_u^2} = \frac{A_u I - S_u}{A_u^2}\mathcal{N} = \frac{I-u}{A_u}\mathcal{N},$$

and

$$\nabla w = -\frac{I-w}{A_w}\mathcal{N}.$$

Thus

$$\nabla g_a = (u-w)\left(\frac{I-u}{A_u} + \frac{I-w}{A_w}\right).$$

A.3 Normal Geometric Dynamic Curve Evolution

The action functional is

$$\begin{aligned} S &= \int\int \left(\frac{1}{2}\|\mathcal{C}_t\|^2 - g\right)\, ds\, dt \\ &= \int\int \left(\frac{1}{2}\|\mathcal{C}_t\|^2 - g\right)\|\mathcal{C}_p\|\, dp\, dt. \end{aligned}$$

Computing the first variation results in

$$\begin{aligned} \delta S &= \int\int \left\{\mathcal{C}_t \cdot \delta\mathcal{C}_t\|\mathcal{C}_p\| - \nabla g \cdot \delta\mathcal{C}\|\mathcal{C}_p\| + \left(\frac{1}{2}\|\mathcal{C}_t\|^2 - g\right)\frac{\mathcal{C}_p \delta\mathcal{C}_p}{\|\mathcal{C}_p\|^2}\|\mathcal{C}_p\|\right\}\, dp\, dt \\ &= \int\int \left\{\mathcal{C}_t \cdot \delta\mathcal{C}_t\|\mathcal{C}_p\| - \nabla g \cdot \delta\mathcal{C}\|\mathcal{C}_p\| + \left(\frac{1}{2}\|\mathcal{C}_t\|^2 - g\right)\mathcal{T} \cdot \delta\mathcal{C}_s\|\mathcal{C}_p\|\right\}\, dp\, dt \\ &= \int\int \left\{\frac{-1}{\mathcal{C}_p}\left(\|\mathcal{C}_p\|\mathcal{C}_t\right)_t - \nabla g - \left(\left(\frac{1}{2}\|\mathcal{C}_t\|^2 - g\right)\mathcal{T}\right)_s\right\} \cdot \delta\mathcal{C}\, ds\, dt, \end{aligned}$$

using integration by parts. This gives the Euler-Lagrange equation:

$$\frac{1}{\mathcal{C}_p}\left(\|\mathcal{C}_p\|\mathcal{C}_t\right)_t + \nabla g + \left(\left(\frac{1}{2}\|\mathcal{C}_t\|^2 - g\right)\mathcal{T}\right)_s = 0.$$

Expanding the derivatives yields

$$\mathcal{C}_{tt} + \frac{1}{\|\mathcal{C}_p\|}\frac{\mathcal{C}_p \cdot \mathcal{C}_{tp}}{\|\mathcal{C}_p\|} + \nabla g + (\mathcal{C}_t \cdot \mathcal{C}_{ts})\mathcal{T} - (\mathcal{T}\nabla g)\mathcal{T} + \left(\frac{1}{2}\|\mathcal{C}_t\|^2 - g\right)\kappa\mathcal{N} = 0.$$

Using

$$\mathcal{C}_s = \mathcal{T}, \text{ and } \nabla g - (\mathcal{T}\cdot\nabla g)\mathcal{T} = (\mathcal{N}\cdot\nabla g)\mathcal{N}$$

one obtains

$$\mathcal{C}_{tt} + (\mathcal{T}\cdot\mathcal{C}_{ts})\mathcal{C}_t + (\mathcal{C}_t\cdot\mathcal{C}_{ts})\mathcal{T} + (\mathcal{N}\cdot\nabla g)\mathcal{N} + \left(\frac{1}{2}\|\mathcal{C}_t\|^2 - g\right)\kappa\mathcal{N} = 0.$$

One can always choose the parameterization of $\mathcal{C}$ so that

$$\mathcal{C}_t = \beta\mathcal{N}.$$

Then

$$\begin{aligned}\mathcal{C}_{ts} &= (\beta\mathcal{N})_s = \beta_s\mathcal{N} - \kappa\beta\mathcal{T}\\ \mathcal{C}_{tt} &= (\beta\mathcal{N})_t = \beta_t\mathcal{N} - \beta\beta_s\mathcal{T},\end{aligned}$$

(since $\mathcal{N}_t = -\beta_s\mathcal{T}$). Thus

$$\beta_t\mathcal{N} - \beta\beta_s\mathcal{T} + (-\kappa\beta)\beta\mathcal{N} + \beta\beta_s\mathcal{T} + (\mathcal{N}\nabla g)\mathcal{N} + \left(\frac{1}{2}\beta^2 - g\right)\kappa\mathcal{N} = 0,$$

and finally,

$$\beta_t - \left(g + \frac{1}{2}\beta^2\right)\kappa + \mathcal{N}\nabla g = 0.$$

APPENDIX B

NUMERICAL METHODS

This appendix briefly discusses some of the numerical methods used for the implementation of the normal geometric dynamic active contour using the partial and the full level set methods. The numerical methods for the classical level set method are well developed. For details we refer the interested reader to [115, 104] and for more numerical background information to the excellent books by LeVeque on numerical methods for hyperbolic problems [83, 84]. Since we are interested in the dynamics of our equations and not only the steady state solution (as is for example usually the case for level set based segmentation algorithms for static images) higher order numerical methods become potentially very useful. However, this is beyond the scope of this thesis. We restrict ourselves (also for the sake of presentational simplicity) to the most basic numerical algorithms and only care about consistency and stability at this point. Section B.1 introduces some notation and operators. Section B.2 discusses the numerics for the partial level set approach. As an interlude, Section B.4 addresses the extension of quantities and Section B.5 the redistancing of a level set function. Finally, Section B.6 gives some background on the numerics for the vector distance function based full level set approach.

B.1 Numerical Operators

We define the first order discrete differentiation operators as

$$\begin{aligned} D^-_{\boldsymbol{e}_i} F(\boldsymbol{x}) &:= \frac{F(\boldsymbol{x}) - F(\boldsymbol{x} - \boldsymbol{e}_i)}{\Delta \boldsymbol{e}_i}, \\ D^+_{\boldsymbol{e}_i} F(\boldsymbol{x}) &:= \frac{F(\boldsymbol{x} + \boldsymbol{e}_i) - F(\boldsymbol{x})}{\Delta \boldsymbol{e}_i}, \\ D^c_{\boldsymbol{e}_i} F(\boldsymbol{x}) &:= \frac{F(\boldsymbol{x} + \boldsymbol{e}_i) - F(\boldsymbol{x} - \boldsymbol{e}_i)}{2\Delta \boldsymbol{e}_i}, \end{aligned}$$

and the second order central derivative operators as

$$\begin{aligned} D^c_{\boldsymbol{e}_i\boldsymbol{e}_i}F(\boldsymbol{x}) &:= \frac{F(\boldsymbol{x}+\boldsymbol{e}_i)-2F(\boldsymbol{x})+F(\boldsymbol{x}-\boldsymbol{e}_i)}{(\Delta\boldsymbol{e}_i)^2}, \\ D^c_{\boldsymbol{e}_i\boldsymbol{e}_j}F(\boldsymbol{x}) &:= \frac{F(\boldsymbol{x}+\boldsymbol{e}_i+\boldsymbol{e}_j)+F(\boldsymbol{x}-\boldsymbol{e}_i-\boldsymbol{e}_j)}{4\Delta\boldsymbol{e}_i\Delta\boldsymbol{e}_j} - \\ &\quad -\frac{F(\boldsymbol{x}+\boldsymbol{e}_i-\boldsymbol{e}_j)+F(\boldsymbol{x}-\boldsymbol{e}_i+\boldsymbol{e}_j)}{4\Delta\boldsymbol{e}_i\Delta\boldsymbol{e}_j} \quad \text{for } i\neq j, \end{aligned}$$

where $\cdot^-$, $\cdot^+$, $\cdot^c$ denote the backward, forward and central differences respectively. The direction of differentiation is given by $\boldsymbol{e}_i$ (which correspond to the coordinate directions of the discrete grid), the discretization step size in direction $\boldsymbol{e}_i$ is $\Delta\boldsymbol{e}_i$, and $F:\mathbb{R}^d\mapsto\mathbb{R}$.

B.2 Numerics for the Partial Level Set Approach

The velocity evolves as

$$\beta_t = \nabla\cdot\left(\frac{\nabla\Phi}{\|\nabla\Phi\|}\left(\frac{g}{\mu}+\frac{1}{2}\beta^2\right)\right),$$

which simplifies, since we extend β normal to the curve,to

$$\beta_t = \nabla\cdot\left(\frac{\nabla\Phi}{\|\nabla\Phi\|}\frac{g}{\mu}\right)+\frac{1}{2}\beta^2\kappa, \tag{80}$$

where

$$\kappa = \frac{\Phi_x^2\Phi_{yy}-2\Phi_x\Phi_y\Phi_{xy}+\Phi_y^2\Phi_{xx}}{\Phi_x^2+\Phi_y^2}$$

is the curvature. Based on this velocity the level sets of the curve move as

$$\Phi_t-\beta\|\nabla\Phi\|=0. \tag{81}$$

Equations (80) and (81) need to get solved simultaneously. We propose the following algorithm:

(1) Compute one time step of Equation (80).

(2) Extend β away from the zero level set of Φ.

(3) Propagate the level set function Φ according to Equation (81) based on the velocities computed in step 2.

(4) Go to step 1.

Section B.4 discusses the extension of β. The solution method breaks the dependency between Equations (80) and (81) by computing a solution to β keeping Φ fixed and vice versa. Equation (80) is then an ordinary differential equation in β.

Let's define

$$c := \nabla \cdot \left(\frac{\nabla \Phi}{\|\nabla \Phi\|} \frac{g}{\mu} \right), \qquad d := \frac{1}{2}\kappa.$$

Then Equation (80) can be rewritten as

$$\beta_t = d\beta^2 + c. \tag{82}$$

Equation (82) can be solved analytically. Separation of variables yields (for $dc \neq 0$)

$$\int_{\beta_0}^{\beta} \frac{1}{d\overline{\beta}^2 + c} \, d\overline{\beta} = \int_{t_0}^{t} d\overline{t}.$$

The solution is then

$$\beta(t) = \begin{cases} \beta_0 & \text{for} \quad cd = 0 \vee \beta^2 = -\frac{c}{d}, \\ \dfrac{\beta_0}{1 + \beta_0 d(t_0 - t)} & \text{for} \quad c = 0, \\ \beta_0 + c(t - t_0) & \text{for} \quad d = 0, \\ \sqrt{\dfrac{c}{d}} \tan\left(\arctan \dfrac{\beta_0}{\sqrt{\frac{c}{d}}} + d(t - t_0) \right) & \text{for} \quad cd > 0, \\ -\sqrt{-\dfrac{c}{d}} \dfrac{\sqrt{-\frac{c}{d}} - \beta_0 + e^{2d(t_0 - t)} \left(\sqrt{-\frac{c}{d}} + \beta_0 \right)}{\sqrt{-\frac{c}{d}} - \beta_0 - e^{2d(t_0 - t)} \left(\sqrt{-\frac{c}{d}} + \beta_0 \right)} & \text{for} \quad cd < 0 \wedge |\beta| < \sqrt{-\frac{c}{d}}, \\ -\sqrt{-\dfrac{c}{d}} \dfrac{\beta_0 - \sqrt{-\frac{c}{d}} + e^{2d(t_0 - t)} \left(\sqrt{-\frac{c}{d}} + \beta_0 \right)}{\beta_0 - \sqrt{-\frac{c}{d}} - e^{2d(t_0 - t)} \left(\sqrt{-\frac{c}{d}} + \beta_0 \right)} & \text{for} \quad cd < 0 \wedge |\beta| > \sqrt{-\frac{c}{d}}. \end{cases}$$

For $c = 0$, $\beta(t)$ blows up in finite time,

$$t = t_0 + \frac{1}{\beta_0 d} = t_0 + \frac{2}{\beta_0 \kappa},$$

if $\beta_0 < 0 \wedge d < 0 \vee \beta_0 > 0 \wedge d > 0$. In practical applications this will only be the case in pathological situations (e.g., if $g \equiv 0$), since

$$c = 0 \iff \kappa g + \frac{\nabla \Phi}{\|\nabla \Phi\|} \cdot \nabla g = 0.$$

Of concern are also the cases for $cd < 0$. If $cd < 0 \wedge |\beta| < \sqrt{-\frac{c}{d}}$:

$$\begin{aligned}
d\beta^2 + c &< 0 \quad \text{for} \quad \beta < 0 \wedge d > 0 \quad \text{evolves to } \beta = -\sqrt{-\frac{c}{d}}, \\
d\beta^2 + c &> 0 \quad \text{for} \quad \beta < 0 \wedge d < 0 \quad \text{evolves to } \beta = \sqrt{-\frac{c}{d}} \\
d\beta^2 + c &< 0 \quad \text{for} \quad \beta > 0 \wedge d > 0 \quad \text{evolves to } \beta = -\sqrt{-\frac{c}{d}}, \\
d\beta^2 + c &> 0 \quad \text{for} \quad \beta > 0 \wedge d < 0 \quad \text{evolves to } \beta = \sqrt{-\frac{c}{d}}.
\end{aligned}$$

This is well defined. If $cd < 0 \wedge |\beta| > \sqrt{-\frac{c}{d}}$:

$$\begin{aligned}
d\beta^2 + c &< 0 \quad \text{for} \quad \beta > 0 \wedge d < 0 \quad \text{evolves to } \beta = \sqrt{-\frac{c}{d}}, \\
d\beta^2 + c &> 0 \quad \text{for} \quad \beta > 0 \wedge d > 0 \quad \text{evolves to } \beta \to \infty, \\
d\beta^2 + c &< 0 \quad \text{for} \quad \beta < 0 \wedge d < 0 \quad \text{evolves to } \beta \to -\infty, \\
d\beta^2 + c &> 0 \quad \text{for} \quad \beta < 0 \wedge d > 0 \quad \text{evolves to } \beta = -\sqrt{-\frac{c}{d}}.
\end{aligned}$$

Here, the normal velocity blows up if $\beta > 0 \wedge d > 0 \vee \beta < 0 \wedge d < 0$. It is important to keep in mind that overall an equation system gets evolved and not only the evolution equation for β, but then

$$cd = \frac{1}{2}\kappa^2 g + \frac{1}{2}\kappa \frac{\nabla\Phi}{\|\nabla\Phi\|} \cdot \nabla g,$$

will ultimately change sign if g is designed properly. These two cases should thus be of no relevance in real life applications. However, it is easy to design cases where this behavior can be observed (e.g., a circle collapsing under its own curvature influence or a circle blowing up to an infinite radius). The straightforward approach to solve for β for one time step is to use an ordinary differential equation solver, e.g., an Euler forward approach in the simplest possible case. Alternatively, the analytical solution (as shown above) of Equation (82) gives the solution for β for a given time step. However, since the solution to Equation (82) may exhibit finite-time blowup behavior it is advisable to use a numerical scheme like Euler forward for its solution. This will lead to conservative estimates of β in pathological cases. Section B.3 gives one possible solution method for Equation (81) given an updated value for β.

B.3 Numerical Implementation of a Generic Level Set Evolution Equation

Consider the evolution of closed curves of the form $\mathcal{C} : S^1 \times [0, \tau) \mapsto \mathbb{R}^2$ in the plane. Where $\mathcal{C} = \mathcal{C}(p, t)$ and $\mathcal{C}(0, t) = \mathcal{C}(1, t)$ [128, 76], with t being the time, and $p \in [0, 1]$ the curve's parameterization. The most general curve evolution equation is

$$\mathcal{C}_t = \boldsymbol{x}_t, \tag{83}$$

where $\boldsymbol{x}_t$ is the velocity associated with any point on the curve. The associated level set equation is

$$\Phi_t + \boldsymbol{x}_t \cdot \nabla\Phi = 0.$$

The velocity vector $\boldsymbol{x}_t$ decomposes into

$$\boldsymbol{x}_t = \boldsymbol{V} - \nabla g + a\frac{\nabla\Phi}{\|\nabla\Phi\|} - b\kappa\frac{\nabla\Phi}{\|\nabla\Phi\|},$$

where g is a smooth function on Ω (the domain of interest), a and b are (possibly non-smooth) functions on Ω, κ denotes curvature and $\boldsymbol{V} = [u, v]^T$ is an external vector field. The term

$$T_1 = -b\kappa\frac{\nabla\Phi}{\|\nabla\Phi\|}$$

can be approximated by central differences. The same is true for ∇g, since g is smooth. This amounts to

$$\begin{aligned}
\kappa &= \frac{(D_x^c\Phi)^2 D_{yy}^c\Phi - 2D_x^c\Phi D_y^c\Phi D_{xy}^c\Phi + (D_y^c\Phi)^2 D_{xx}^c\Phi}{(D_x^c\Phi)^2 + (D_y^c\Phi)^2}, \text{ where} \\
\Phi_x &= D_x^c\Phi, \\
\Phi_y &= D_y^c\Phi, \text{ and} \\
g_x &= D_x^c g, \\
g_y &= D_y^c g.
\end{aligned}$$

Having discretized ∇g, define

$$\overline{\boldsymbol{v}} := \boldsymbol{V} - \nabla g = \begin{pmatrix} \overline{v}_1 \\ \overline{v}_2 \end{pmatrix}.$$

It remains then to discretize

$$T_2 = \overline{\boldsymbol{v}} + a\frac{\nabla\Phi}{\|\nabla\Phi\|},$$

which is more complicated to handle than T_1, since it is not immediately clear how to discretize the derivatives of Φ in this case. The solution to the linear advection equation

$$u_t + au_x = 0,$$

is its initial data propagating to the right (the left) for $a > 0$ (for $a < 0$) [83]. T_2 is then the speed vector of a multi-dimensional advection (transport) equation. One of the simplest ways for this solution of problem is Godunov's method (in our case with dimensional splitting; see [83, 84] for details). The idea of Godunov's method is to propagate based on cell averages and then to reaverage the result. This results automatically in an upwind structure of the resulting numerical scheme. Figure 37 shows this for a simple one dimensional example. Assuming $a > 0$ the solution gets propagated by $\Delta s = a\Delta t$ during the time increment Δt, but then reaveraging the cell i at time instant n yields

$$\Delta x u_i^{n+1} = u_i^n(\Delta x - \Delta s) + \Delta s u_{i-1}^n,$$

which can be rewritten as

$$u_i^{n+1} = u_i^n - \Delta t a\frac{u_i^n - u_{i-1}^n}{\Delta x}.$$

This corresponds to a forward Euler scheme with upwind differencing for the spatial derivative. The literature on numerical methods for these types of equations is extensive. See [137] for a higher order extension of the Godunov scheme and [81] for central schemes for example. Since T_2 depends on the derivatives of Φ, the numerical scheme needs to be adapted slightly. In particular, it needs to be able to handle shocks and rarefactions. The following presentation follows [104]. Assuming Φ is (at least approximately) a signed distance function

$$T_2 = \begin{pmatrix} H_1 \\ H_2 \end{pmatrix} = \overline{\boldsymbol{v}} + a\nabla\Phi = \begin{pmatrix} \overline{v}_1 + a\Phi_x \\ \overline{v}_2 + a\Phi_y \end{pmatrix}.$$

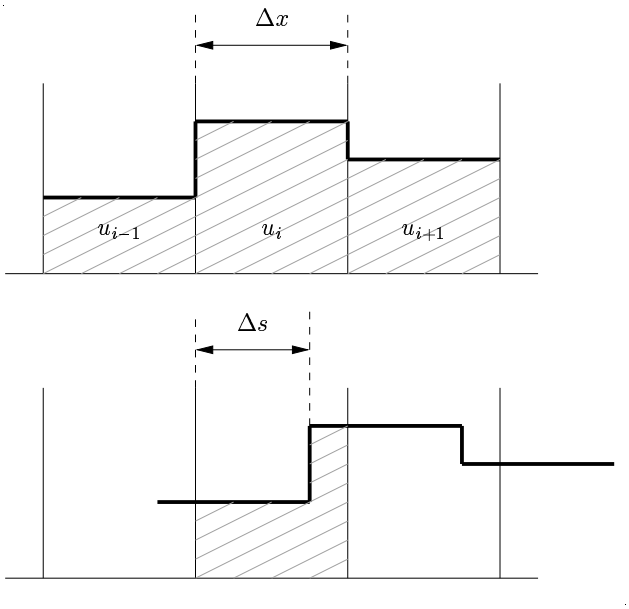


Figure 37: The piecewise constant Godunov scheme applied to a transport equation.

Then, the x and y components of T_2 are decoupled. This allows for the independent discretization of the x and y components of T_2[1]. Discretizing the x component as

$$
\begin{aligned}
H_1 > 0 \text{ for } D_x^+\Phi \text{ and } D_x^-\Phi &\implies \text{use } \Phi_x = D_x^-\Phi \\
H_1 < 0 \text{ for } D_x^+\Phi \text{ and } D_x^-\Phi &\implies \text{use } \Phi_x = D_x^+\Phi \\
H_1 < 0 \text{ for } D_x^-\Phi \text{ and } H_1 > 0 \text{ for} D_x^+\Phi &\implies \text{rarefaction, use } \Phi_x = -\frac{\overline{v}_1}{a} \\
H_1 > 0 \text{ for } D_x^-\Phi \text{ and } H_1 < 0 \text{ for} D_x^+\Phi &\implies \text{shock} \\
\Phi_x = \begin{cases} D_x^-\Phi & \text{if } |H_1(D_x^-\Phi)| > |H_1(D_x^+\Phi)| \\ D_x^+\Phi & \text{otherwise} \end{cases} &,
\end{aligned}
$$

[1]Maintaining the coupling would complicate the numerical scheme significantly.

and the y component as

$$
\begin{aligned}
H_2 > 0 \text{ for } D_y^+\Phi \text{ and } D_y^-\Phi &\implies \text{use } \Phi_y = D_y^-\Phi \\
H_2 < 0 \text{ for } D_y^+\Phi \text{ and } D_y^-\Phi &\implies \text{use } \Phi_y = D_y^+\Phi \\
H_2 < 0 \text{ for } D_y^-\Phi \text{ and } H_2 > 0 \text{ for } D_y^+\Phi &\implies \text{rarefaction, use } \Phi_y = -\frac{\overline{v}_2}{a} \\
H_2 > 0 \text{ for } D_y^-\Phi \text{ and } H_2 < 0 \text{ for } D_y^+\Phi &\implies \text{shock}
\end{aligned}
$$

$$
\Phi_x = \begin{cases} D_y^-\Phi & \text{if } |H_2(D_y^-\Phi)| > |H_2(D_y^+\Phi)| \\ D_y^+\Phi & \text{otherwise} \end{cases},
$$

describes the numerical scheme. The time derivative is approximated by forward Euler. Stability of the numerical scheme needs to be guaranteed by a sensible time step governed by the Courant-Friedrichs-Lewy (CFL) condition. The computed velocities now need to be extended to all of Ω. The solution method for Equation (81) follows immediately as a special case for

$$
\boldsymbol{V} = \nabla g = \mathbf{0},\ a = -\beta, b = 0.
$$

The overall scheme is in this case (using the Euler forward method for time discretization)

$$
\Phi^{n+1}[i,j] = \Phi^n[i,j] - \Delta t \left(\max(-\beta[i,j],0)\nabla^+ + \min(-\beta[i,j],0)\nabla^-\right)
$$

where

$$
\begin{aligned}
\nabla^+ &= \left(\max(\max(D_x^-\Phi,0)^2, min(D_x^+\Phi,0)^2) + \max(\max(D_y^-\Phi,0)^2, \min(D_y^+\Phi,0)^2)\right)^{\frac{1}{2}} \\
\nabla^- &= \left(\max(\min(D_x^-\Phi,0)^2, \max(D_x^+\Phi,0)^2) + \max(\min(D_y^-\Phi,0)^2, \max(D_y^+\Phi,0)^2)\right)^{\frac{1}{2}}.
\end{aligned}
$$

B.4 Extending Quantities

The proposed algorithm of Section B.2 requires the extension of β away from the zero level set of Φ. Two possible approaches are extension methods based on the fast marching algorithm [115] and extension methods based on the iterative solution of a partial differential equation [104]. The former is essentially an iteration-free method to obtain the solution of the partial differential equation solved iteratively in the iterative approaches. Even though an iteration-free (by construction) methodology, like the fast-marching approach, is very

efficient, this section will exclusively focus on iterative methods. These have the advantage of extending easily to higher-dimensional problems and they facilitate the use of higher order numerical schemes, to obtain high order accurate solutions.

The solution of the partial differential equation

$$s_t + \text{sign}(\Phi)\frac{\nabla\Phi}{\|\nabla\Phi\|} \cdot \nabla s = 0$$

extends the scalar quantity s away from the zero level set of Φ [104]. The discretization follows from the scheme presented in Section B.3. In particular,

$$\boldsymbol{V} = \text{sign}(\Phi)\frac{\nabla\Phi}{\|\nabla\Phi\|},\ \nabla g = \mathbf{0}, a = b = 0,$$

where the derivatives of Φ are computed using central differences. The extension of β is then straightforward.

B.5 Redistancing

As discussed in Section 5.2 the redistancing of vector distance functions is nontrivial. This section describes the redistancing of signed distance functions only. However, this is identical to redistancing an unsigned distance function (and thus also a vector distance function) if the redistancing is not performed in the vicinity of the zero level set.

The solution of the partial differential equation [125]

$$\Phi_t + S(\Phi_0)\frac{\nabla\Phi}{\|\nabla\Phi\|} \cdot \nabla\Phi = S(\Phi_0) \tag{84}$$

redistances the signed distance function Φ, where Φ_0 is the initial condition of Φ and

$$S(\Phi_0(\boldsymbol{x})) = \frac{\Phi_0(\boldsymbol{x})}{\sqrt{\Phi_0(\boldsymbol{x})^2 + \epsilon^2}},$$

with $\epsilon \in \mathbb{R}$ small. The discretization method of Section B.3 applies with

$$\boldsymbol{V} = \nabla g = \mathbf{0},\ a = S(\Phi_0),\ b = 0$$

to the left hand side of Equation (84).

B.6 Numerics for the Vector Distance Function Based Full Level Set Approach

The overall scheme for the full level set approach is described in Sections 5.2 and 5.4. The quantities needed to compute the current velocities (see Section 5.4.1) are all discretized using central differences. The redistancing scheme of Section B.5 facilitates the redistancing away from the zero level set of $\boldsymbol{u}$. The same is true for the velocity extensions, where the velocities are all extended component by component in accordance with the scheme presented in Section B.4. What remains is the discretization of

$$\boldsymbol{u}_t - (D\boldsymbol{u})^T \boldsymbol{b} = \boldsymbol{0}.$$

Since $D\boldsymbol{u}$ is symmetric, this is equivalent to the component-wise form

$$u_t^i - \boldsymbol{b} \cdot \nabla u^i = 0,\ 1 \leq i \leq n,\ i \in \mathbb{N},$$

but with

$$\boldsymbol{V} = -\boldsymbol{b},\ \nabla g = \boldsymbol{0},\ a = b = 0,$$

the scheme of Section B.3 applies.

REFERENCES

[1] Achenbach, J. D., *Wave Propagation in Elastic Solids*. North-Holland, 1975.

[2] Adalsteinsson, D. and Sethian, J. A., "A fast level set method for propagating interfaces," *Journal of Computational Physics*, vol. 118, pp. 269–277, 1995.

[3] Ahmed, N. U., *Linear and Nonlinear Filtering for Scientists and Engineers*. World Scientific, 1998.

[4] Allen, B. D., Bishop, G., and Welch, G., "Tracking: Beyond 15 minutes of thought: Siggraph 2001 course 11." Course Notes, Annual Conference on Computer Graphics and Interactive Techniques (Siggraph 2001), 2001.

[5] Alspach, D. L. and Sorenson, H. W., "Nonlinear Bayesian estimation using Gaussian sum approximations," *IEEE Transactions on Automatic Control*, vol. 17, no. 4, pp. 439–448, 1972.

[6] Ambrosio, L. and Soner, M., "Level set approach to mean curvature flow in arbitrary codimension," *Journal of Differential Geometry*, vol. 43, pp. 693–737, 1996.

[7] Arulampalam, M., Maskell, S., Gordon, N., and Clapp, T., "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–189, 2002.

[8] Ayache, N., Cohen, I., and Herlin, I., *Active Vision*, ch. Medical image tracking, pp. 3–20. MIT Press, 1992.

[9] Bellettini, G., Novaga, M., and Paolini, M., "An example of three dimensional fattening for linked space curves evolving by curvature," *Communications in Partial Differential Equations*, vol. 23, pp. 1475–1492, 1998.

[10] Bergman, N., *Recursive Bayesian Estimation: Navigation and Tracking Applications*. PhD thesis, Linköping University, 1999.

[11] Bertalmio, M., Sapiro, G., and Randall, G., "Region tracking on level-sets methods," *IEEE Transactions on Medical Imaging*, vol. 18, no. 5, pp. 448–451, 1999.

[12] Bertrand, G., "A Boolean characterization of three-dimensional simple points," *Pattern Recognition Letters*, vol. 17, pp. 115–124, 1996.

[13] Bethuel, F. and Ghidaglia, J.-M., *Geometry in Partial Differential Equations*, ch. 1, pp. 1–17. World Scientific Publishing Co., 1994.

[14] Beymer, D., McLauchlan, P., Coifman, B., and Malik, J., "A real-time computer vision system for measuring traffic parameters," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 495–501, IEEE, 1997.

[15] Bhanu, B., Dudgeon, D. E., Zelnio, E. G., Rosenfeld, A., Casasent, D., and Reed, I. S., "Introduction to the special issue on automatic target detection and recognition," *IEEE Transactions on Image Processing*, vol. 6, no. 1, pp. 1–6, 1997.

[16] Blake, A., Curwen, R., and Zisserman, A., "A framework for spatio-temporal control in the tracking of visual contours," *International Journal of Computer Vision*, vol. 11, no. 2, pp. 127–145, 1993.

[17] Blake, A. and Isard, M., *Active Contours*. Springer Verlag, 1998.

[18] Blake, A., Isard, M., and Reynard, D., "Learning to track the visual motion of contours," *Artificial Intelligence*, vol. 78, pp. 101–134, 1995.

[19] Blake, A. and Yuille, A., eds., *Active Vision*. MIT Press, 1992.

[20] Burchard, P., Cheng, L., Merriman, B., and Osher, S., "Motion of curves in three spatial dimensions using a level set approach," *Journal of Computational Physics*, vol. 170, pp. 720–741, 2001.

[21] Caselles, V., Catte, F., Coll, T., and Dibos, F., "A geometric model for active contours in image processing," *Numerische Mathematik*, vol. 66, pp. 1–31, 1993.

[22] Caselles, V., Kimmel, R., and Sapiro, G., "Geodesic active contours," *International Journal of Computer Vision*, vol. 22, no. 1, pp. 61–79, 1997.

[23] Chen, Y., Tagare, H. D., Thiruvenkadam, S., Huang, F., Wilson, D., Gopinath, K. S., Briggs, R. W., and Geiser, E. A., "Using prior shapes in geometric active contours in a variational framework," *International Journal of Computer Vision*, vol. 50, no. 3, pp. 315–328, 2002.

[24] Chen, Y., Thiruvenkadam, S., Tagare, H. D., Huang, F., Wilson, D., and Geiser, E. A., "On the incorporation of shape priors into geometric active contours," in *Proceedings of the Conference on Variational and Level Set Methods in Computer Vision*, pp. 145–152, IEEE, 2001.

[25] Cootes, T. F., Edwards, G. J., and Taylor, C. J., "Active appearance models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 681–685, 2001.

[26] Cootes, T. F. and Taylor, C. J., "Statistical models of appearance for medical image analysis and computer vision," in *Proceedings of the Conference on Medical Imaging*, vol. 4322, pp. 236–248, SPIE, 2001.

[27] Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J., "Active shape models – their training and application," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 38–59, 1995.

[28] Corke, P., *Visual Servoing*, vol. 7 of *Robotics and Automated Systems*, ch. Visual control of robotic manipulators – a review, pp. 1–31. World Scientific, 1993.

[29] Courant, R. and Hilbert, D., *Methods of Mathematical Physics*, vol. 2. Wiley, 1989.

[30] Cox, I. J., "A review of statistical data association techniques for motion correspondence," *International Journal of Computer Vision*, vol. 10, no. 1, pp. 53–66, 1993.

[31] Crandall, M. G., Ishii, H., and Lions, P.-L., "User's guide to viscosity solutions of second order partial differential equations," *Bulletin of the American Mathematical Society*, vol. 27, no. 1, pp. 1–67, 1992.

[32] Cremers, D. and Schnörr, C., "Statistical shape knowledge in variational motion segmentation," *Image and Vision Computing*, vol. 21, pp. 77–86, 2003.

[33] Cremers, D. and Soatto, S., "A pseudo-distance for shape priors in level set segmentation," in *Proceedings Workshop on Variational, Geometric and Level Set Methods in Computer Vision*, pp. 169–176, IEEE, 2003.

[34] Cremers, D., Tischhäuser, F., Weickert, J., and Schnörr, C., "Diffusion snakes: Introducing statistical shape knowledge into the Mumford-Shah functional," *International Journal of Computer Vision*, vol. 50, no. 3, pp. 295–313, 2002.

[35] Curtain, R. F. and Zwart, H. J., *An Introduction to Infinite-Dimensional Linear Systems Theory*. No. 21 in Texts in Applied Mathematics, Springer Verlag, 1995.

[36] Dickmanns, E. D., "The 4D-approach to dynamic machine vision," in *Proceedings of the 33rd conference on decision and control*, pp. 3770–3775, IEEE, 1994.

[37] Dierckx, P., *Curve and Surface Fitting with Splines*. Monographs on Numerical Analysis, Oxford Science Publications, 1993.

[38] do Carmo, M. P., *Differential Geometry of Curves and Surfaces*. Prentice Hall, 1976.

[39] Doucet, A., Godsill, S., and Andrieu, C., "On sequential monte carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, pp. 197–208, 2000.

[40] Doyle, J., Francis, B., and Tannenbaum, A., *Feedback Control Theory*. Macmillan, 1992.

[41] Dupont, T. F. and 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*, vol. 190, pp. 311–324, 2003.

[42] Dupont, T. F. and Liu, Y., "Back and forth error compensation and correction methods for semi-Lagrangian schemes with applications to interface computation using level set method," Tech. Rep. CDSNS2004-399, Georgia Institute of Technology, 2004.

[43] Evans, L. C. and Spruck, J., "Motion of level sets by mean curvature I," *Journal of Differential Geometry*, vol. 33, pp. 635–681, 1991.

[44] Evans, L. C. and Spruck, J., "Motion of level sets by mean curvature II," *Transactions of the American Mathematical Society*, vol. 330, pp. 321–332, 1992.

[45] Evans, L. C. and Spruck, J., "Motion of level sets by mean curvature III," *Journal of Geometric Analysis*, no. 2, pp. 121–150, 1992.

[46] Evans, L. C. and Spruck, J., "Motion of level sets by mean curvature IV," *Journal of Geometric Analysis*, no. 5, pp. 77–114, 1995.

[47] Evans, L., *Partial Differential Equations*, vol. 19 of *Graduate Studies in Mathematics*. American Mathematical Society, 1998.

[48] Franke, U., Gavrila, D., Görzig, S., Lindner, F., Paetzold, F., and Wöhler, C., "Autonomous driving goes downtown," *IEEE Intelligent Systems*, vol. 13, no. 6, pp. 40–48, 1999.

[49] Frankiln, G. F., Powell, J. D., and Emami-Naeini, A., *Feedback Control of Dynamic Systems*. Addison-Wesley, 3rd ed., 1994.

[50] Freedman, D. and Brandstein, M. S., "Contour tracking in clutter: A subset approach," *International Journal of Computer Vision*, vol. 38, no. 2, pp. 173–186, 2000.

[51] Fusiello, A., Trucco, E., Tommasini, T., and Roberto, V., "Improving feature tracking with robust statistics," *Pattern Analysis and Applications*, vol. 2, pp. 312–320, 1999.

[52] Gau, C. J. and Kong, T. Y., "4D minimal non-simple sets," vol. 2301 of *Lecture Notes in Computer Science*, pp. 81–91, Springer Verlag, 2002.

[53] Gelb, A., ed., *Applied Optimal Estimation*. MIT Press, 15th ed., 1999.

[54] Geman, S. and Geman, D., "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 6, no. 6, pp. 721–741, 1984.

[55] Gomes, J. and Faugeras, O., "Representing and evolving smooth manifolds of arbitrary dimension embedded in R^n as the intersection of n hypersurfaces: The vector distance functions," Tech. Rep. 4012, INRIA, 2000.

[56] Gomes, J. and Faugeras, O., "Shape representation as the intersection of $n - k$ hypersurfaces," Tech. Rep. 4011, INRIA, 2000.

[57] Gomes, J., Faugeras, O., and Kerckhove, M., "Using the vector distance functions to evolve manifolds of arbitrary codimension," in *Scale-Space and Morphology in Computer Vision*, vol. 2106 of *Lecture Notes in Computer Science*, pp. 1–13, 2001.

[58] Gould, P. L., *Introduction to Linear Elasticity*. Springer-Verlag, 1994.

[59] Grenander, U. and Miller, M. I., "Representations of knowledge in complex systems," *Journal of the Royal Statistical Society, Series B*, vol. 56, no. 4, pp. 549–603, 1994.

[60] Haker, S., Sapiro, G., and Tannenbaum, A., "Knowledge-based segmentation of SAR data with learned priors," *IEEE Transactions on Image Processing*, vol. 9, no. 2, pp. 299–301, 2000.

[61] Hotraphinyo, L. F. and Riviere, C. N., "Precision measurement for microsurgical instrument evaluation," in *Proceedings of the 23rd Annual EMBS International Conference*, pp. 3454–3457, 2001.

[62] Hutchinson, S., Hager, G. D., and Corke, P. I., "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996.

[63] Intille, S. S., Davis, J. W., and Bobick, A. F., "Real-time closed-world tracking," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 697–703, IEEE, 1997.

[64] Isard, M. and Blake, A., "Condensation – conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[65] Jazwinski, A. H., *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

[66] Julier, S. and Bishop, G., "Tracking: how hard can it be?," *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 22–23, 2002.

[67] Kaczynski, T., Mischaikow, K., and Mrozek, M., "Computing homology," tech. rep., Georgia Institute of Technology, 2001.

[68] Kaczynski, T., Mischaikow, K., and Mrozek, M., *Computational Homology*, vol. 157 of *Applied Mathematical Sciences*. Springer Verlag, 2004.

[69] Kalies, W. D., Mischaikow, K., and Watson, G., "Cubical approximation and computation of homology," in *Conley Index Theory*, pp. 115–131, Banach Center Publications, 1999.

[70] Kallianpur, G., *Stochastic Filtering Theory*. Springer Verlag, 1980.

[71] Kansy, K., Berlage, T., Schmitgen, G., and Wiškirchen, P., "Real-time integration of synthetic computer graphics into live video scenes," in *Proceedings of the Conference on the Interface of Real and Virtual Worlds*, pp. 93–101, 1995.

[72] Karatzas, I., "A tutorial introduction to stochastic analysis and its applications." Lecture notes., 1988.

[73] Karatzas, I. and Shreve, S. E., *Brownian Motion and Stochastic Calculus*. Springer Verlag, 1991.

[74] Kass, M., Witkin, A., and Terzopoulos, D., "Snakes: Active contour models," *International Journal of Computer Vision*, pp. 321–331, 1988.

[75] Kichenassamy, S., Kumar, A., Olver, P., Tannenbaum, A., and Yezzi, A., "Conformal curvature flows: From phase transitions to active vision," *Archive for Rational Mechanics and Analysis*, vol. 134, no. 3, pp. 275–301, 1996.

[76] Kimia, B., Tannenbaum, A., and Zucker, S., "Shapes, shocks, and deformations, I: the components of shape and the reaction-diffusion space," *International Journal of Computer Vision*, vol. 15, pp. 189–224, 1995.

[77] Kimia, B. B., Tannenbaum, A., and Zucker, S. W., "On the evolution of curves via a function of curvature. I. the classical case," *Journal of Mathematical Analysis and Applications*, vol. 163, no. 2, pp. 438–458, 1992.

[78] Klette, G., "Simple points in 2D and 3D binary images," vol. 2756 of *Lecture Notes in Computer Science*, pp. 57–64, Springer Verlag, 2003.

[79] Kong, T. Y., "Topology-preserving deletion of 1's from 2-, 3- and 4-dimensional binary images," in *Discrete Geometry for Computer Imagery*, vol. 1347 of *Lecture Notes in Computer Science*, pp. 3–18, Springer, 1997.

[80] Kriegman, D. J., Hager, G. D., and Morse, A. S., eds., *The confluence of vision and control*, vol. 237 of *Lecture Notes in Control and Information Sciences*. Springer Verlag, 1998.

[81] Kurganov, A. and Tadmor, E., "New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations," *Journal of Computational Physics*, vol. 160, pp. 241–282, 2000.

[82] Leventon, M. E., Grimson, W. E. L., and Faugeras, O., "Statistical shape influence in geodesic active contours," in *Proceeding of the Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 1316–1323, 2000.

[83] LeVeque, R. J., *Numerical Methods for Conservation Laws*. Birkhäuser, 1992.

[84] Leveque, R. J., *Finite Volume Methods for Hyperbolic Problems*. Cambridge Texts in Applied Mathematics, 2002.

[85] Liu, J. S. and Chen, R., "Sequential monte carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, no. 443, pp. 1032–1044, 1998.

[86] Lorigo, L. M., Faugeras, O., Grimson, W. E. L., Keriven, R., Kikinis, R., and Westin, C.-F., "Co-dimension 2 geodesic active contours for MRA segmentation," in *Proceedings of the International Conference on Information Processing in Medical Imaging*, pp. 126–139, 1999.

[87] Lototsky, S., Mikulevicius, R., and Rozovskii, B. L., "Nonlinear filtering revisited: A spectral approach," *SIAM Journal on Control and Optimization*, vol. 35, no. 2, pp. 435–461, 1997.

[88] Lototsky, S. V., *Problems in Statistic of Stochastic Differential Equations*. PhD thesis, University of Southern California, 1996.

[89] Luenberger, D. G., "An introduction to observers," *IEEE Transactions on Automatic Control*, vol. 16, no. 6, pp. 596–602, 1971.

[90] Malik, J. and Russell, S., "A machine vision based surveillance system for California roads," Tech. Rep. UCB-ITS-PRR-95-6, University of California, Berkeley, 1995.

[91] Malladi, R., Sethian, J. A., and Vemuri, B. C., "Shape modeling with front propagation: A level set approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 2, pp. 158–175, 1995.

[92] Massey, W. S., *A Basic Course in Algebraic Topology*. Graduate Texts in Mathematics, Springer-Verlag, 1991.

[93] McLauchlan, P. F. and Malik, J., "Vision for longitudinal vehicle control," in *Proceedings of the Conference on Intelligent Transportation Systems*, pp. 918–923, IEEE, 1997.

[94] Menold, P. H., Findeisen, R., and Allgöwer, F., "Finite time convergent observers for nonlinear systems," in *Proccedings of the Conference on Decision and Control*, vol. 6, pp. 5673–5678, IEEE, 2003.

[95] Miller, R. N., Carter, E. F., and Blue, S. T., "Data assimilation into nonlinear stochastic models," *Tellus A*, vol. 51, pp. 167–194, 1999.

[96] Min, C., "Local level set method in high dimension and codimension," *Journal of computational physics*, vol. 200, pp. 368–382, 2004.

[97] Mitiche, A. and Bouthemy, P., "Computation and analysis of image motion: A synopsis of current problems and methods," *International Journal of Computer Vision*, vol. 19, no. 1, pp. 29–55, 1996.

[98] Mitter, S. K., "Filtering and stochastic control: A historical perspective," *IEEE Control Systems Magazine*, vol. 16, no. 3, pp. 67–76, 1996.

[99] Moghaddam, B. and Pentland, A., "Probabilistic visual learning for object detection," in *Proceedings of the Conference on Computer Vision*, pp. 786–793, IEEE, 1995.

[100] Niethammer, M. and Tannenbaum, A., "Dynamic level sets for visual tracking," in *Proceedings of the Conference on Decision and Control*, IEEE, 2003.

[101] Niethammer, M. and Tannenbaum, A., "Dynamic geodesic snakes for visual tracking," in *Proceedings of the Conference on Computer Vision and Pattern Recognition.*, IEEE, 2004. accepted.

[102] Oka, K., Sato, Y., and Koike, H., "Real-time fingertip tracking and gesture recognition," *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 64–71, 2002.

[103] Osher, S., Cheng, L., Kang, M., Shim, H., and Tsai, Y., "Geometric optics in a phase space based level set and Eulerian framework," *Journal of Computational Physics*, vol. 179, no. 2, pp. 622–648, 2002.

[104] Osher, S. and Fedkiw, R., *Level Set Methods and Dynamic Implicit Surfaces*. Springer Verlag, 2003.

[105] Osher, S. and Fedkiw, R. P., "Level set methods: An overview and some recent results," *Journal of Computational Physics*, vol. 169, pp. 463–502, 2001.

[106] Paragios, N. and Deriche, R., "Geodesic active regions: A new framework to deal with frame partition problems in computer vision," *Journal of Visual Communication and Image Representation*, vol. 13, pp. 249–268, 2002.

[107] Peterfreund, N., "Robust tracking of position and velocity with Kalman snakes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 6, pp. 564–569, 1999.

[108] Pilarczyk, P., "Computer assisted method for proving existence of periodic orbits," *Topological Methods in Nonlinear Analysis*, vol. 13, no. 2, pp. 365–377, 1999.

[109] Pottman, H. and Hofer, M., *Visualization and Mathematics III*, ch. Geometry of the squared distance function to curves and surfaces, pp. 223–244. Springer, 2003.

[110] Remagnino, P., Jones, G. A., Paragios, N., and Regazzoni, C. S., eds., *Video-Based Surveillance Systems*. Kluwer Academic Publishers, 2001.

[111] Risken, H., *The Fokker-Planck Equation*. Springer Verlag, 3rd ed., 1996.

[112] Rousson, M. and Paragios, N., "Shape priors for level set representations," in *Proceedings of the European Conference on Computer Vision*, vol. 2, pp. 78–92, 2002.

[113] Sapiro, G., *Geometric Partial Differential Equations and Image Analysis*. Cambridge University Press, 2001.

[114] Sclaroff, S., "Active blobs: Region-based, deformable appearance models," *Computer Vision and Image Understanding*, vol. 89, no. 2-3, pp. 197–225, 2003.

[115] Sethian, J. A., *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 2nd ed., 1999.

[116] Shah, J., "A common framework for curve evolution, segmentation, and anisotropic diffusion," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 136–142, IEEE, 1996.

[117] Sharp, C. S., Shakernia, O., and Sastry, S. S., "A vision system for landing an unmanned aerial vehicle," in *Proceedings of the International Conference on Robotics and Automation*, pp. 1720–1727, IEEE, 2001.

[118] Shi, J. and Tomasi, C., "Good features to track," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 593–600, IEEE, 1994.

[119] Sinopoli, B., Micheli, M., Donato, G., and Koo, T. J., "Vision based navigation for an unmanned aerial vehicle," in *Proceedings of the International Conference on Robotics and Automation*, pp. 1757–1764, IEEE, 2001.

[120] Slepcev, D., "On level-set approach to motion of manifolds of arbitrary codimension," *Interfaces and Free Boundaries*, vol. 5, pp. 417–458, 2003.

[121] Sokolnikoff, I. S., *Mathematical Theory of Elasticity*. McGraw-Hill, 1956.

[122] Sonka, M., Hlavac, V., and Boyle, R., *Image Processing: Analysis and Machine Vision*. Brooks Cole, 1998.

[123] Sonka, M., Lelieveldt, B. P. F., Bosch, J. G., der Geest, R. J. V., and Reiber, J. H. C., "Active appearance motion model segmentation," in *Proceedings of the Workshop on Digital and Computational Video*, pp. 64–68, IEEE, 2001.

[124] Staib, L. H., "Prior shape models for boundary finding," in *Proceedings of the International Symposium on Biomedical Imaging*, pp. 30–33, IEEE, 2002.

[125] Sussman, M., Smereka, P., and Osher, S., "A level set approach for computing solutions to incompressible two-phase flow," *Journal of Computational Physics*, vol. 114, pp. 146–159, 1994.

[126] Swain, M. J. and Stricker, M. A., "Promising directions in active vision," *International Journal of Computer Vision*, vol. 12, no. 2, pp. 109–126, 1993.

[127] Tanawongsuwan, R. and Bobick, A., "Gait recognition from time-normalized joint-angle trajectories in the walking plane," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 726–731, IEEE, 2001.

[128] Tannenbaum, A., "Three snippets of curve evolution theory in computer vision," *Mathematical and Computer Modelling Journal*, vol. 24, pp. 103–119, 1996.

[129] Terzopoulos, D. and Szeliski, R., *Active Vision*, ch. Tracking with Kalman Snakes, pp. 3–20. MIT Press, 1992.

[130] Thürmer, G., "Closed curves in n-dimensional discrete space," *Graphical Models*, vol. 65, pp. 43–60, 2003.

[131] Tornambè, A., "High-gain observers for non-linear systems," *International Journal of Systems Science*, vol. 23, no. 9, pp. 1475–1489, 1992.

[132] Troutman, J. L., *Variational Calculus and Optimal Control*. Springer Verlag, 2nd ed., 1996.

[133] Tsai, A., Yezzi, A., III, W. W., Tempany, C., Tucker, D., Fan, A., Grimson, W. E., and Willsky, A., "Model-based curve evolution technique for image segmentation," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 463–468, IEEE, 2001.

[134] Ulmer, B., "VITA II – active collision avoidance in real traffic," in *Proceedings of the Intelligent Vehicles Symposium*, pp. 1–6, IEEE, 1994.

[135] Unal, G., Yezzi, A., and Krim, H., "Fast incorporation of optical flow into active polygons," preprint, submitted to *IEEE Transactions on Image Processing*, 2003.

[136] Unser, M., "Splines: a perfect fit for signal and image processing," *IEEE Signal Processing Magazine*, vol. 16, no. 6, pp. 22–38, 1999.

[137] van Leer, B., "Towards the ultimate conservative difference scheme," *Journal of Computational Physics*, vol. 135, pp. 229–248, 1997.

[138] Wang, Y. and Staib, L. H., "Boundary finding with prior shape and smoothness models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 7, pp. 738–743, 2000.

[139] Welch, G. and Foxlin, E., "Motion tracking: no silver bullet, but a respectable arsenal," *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 24–38, 2002.

[140] Wong, E., *Stochastic Processes in Information and Dynamical Systems*. McGraw-Hill, 1971.

[141] Wouwer, A. V. and Zeitz, M., *Control Systems, Robotics and Automation, Theme in Encyclopedia of Life Support Systems*, ch. State estimation in distributed parameter systems. EOLSS Publishers, 2001.

[142] Xu, C. and Prince, J. L., "Snakes, shapes, and gradient vector flow," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 359–369, 1998.

[143] Xu, C., Yezzi, A., and Prince, J. L., "On the relationship between parametric and geometric active contours," in *Proceedings of the Thirty-Fourth Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 483–489, 2000.

[144] Yezzi, A., Tsai, A., and Willsky, A., "Medical image segmentation via coupled curve evolution equations with global constraints," in *Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, pp. 12–19, 2000.

[145] Yezzi, A., Tsai, A., and Willsky, A., "A fully global approach to image segmentation via coupled curve evolution equations," *Journal of Visual Communication and Image Representation*, vol. 13, pp. 195–216, 2002.

[146] Yuille, A. and Hallian, P., *Active Vision*, ch. Deformable Templates, pp. 21–38. MIT Press, 1992.

VITA

Marc Niethammer was born in Waiblingen, Germany on 25 May 1975. He received his Diplom-Ingenieur in Engineering Cybernetics from the Universität Stuttgart, Germany in 2000. In 1999 and 2002, he earned a Masters of Science in Engineering Science and Mechanics and a Masters of Science in Applied Mathematics, both from the Georgia Institute of Technology. In 2004, he graduated from the Georgia Institute of Technology with a Doctor of Philosophy in Electrical and Computer Engineering.