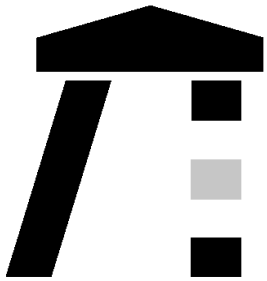
University of Kaiserslautern
Department of Mathematics
Laboratory of Technomathematics

Reinitialization for Level Set Methods

Diploma Thesis

by

Rainer Keck



Kaiserslautern, June 1998

To Renate and Gerhard

Table of Contents

List of Illustrations

Motivation and Outline

Level set methods are used to solve front propagating problems like crystal growth, multi-phase flow, and motion of soap bubbles where a moving boundary must be determined. The main characteristic of front propagating problems is that the location of the moving boundary does essentially influence the evolution of the process. Therefore, it is an important task to know the position of the boundary. The level set method is one approach for this task. In contrast to front tracking methods, which are based on a Lagrangian formulation, the level set method represents the moving boundary as the zero level set of a so-called level set function, which is given on an Eulerian co-ordinate system. Thus, the moving front is 'captured' implicitly by the level set function. This approach avoids some complex problems, such as numerical instabilities or complicated bookkeeping techniques, which occur usually using front tracking methods.
But it turns out that the level set function becomes 'irregular' during the computation. Therefore, it needs to be updated in order to stay 'regular'. This process is called *reinitialization* of the level set function and needs to be done after a small number of evolution time steps. Hence, reinitialization is applied many times during the entire computation.
Since the zero level set represents a moving boundary, which may be an interface of e.g. two different fluids, it must not move during *reinitialization*. In fact, in numerical computations it does. Hence, it introduces an error to the areas separated by the zero level set. Moreover, the error is not qualitatively arbitrary but tends always to *one* direction. In particular, the area enclosed by the zero level set shrinks, i.e. area loss is introduced by reinitialization, which may accumulate from application to application.

The goal of this thesis is to discover why area loss occurs during reinitialization in numerical computations in order to be able to apply numerical schemes that reduces area loss. Furthermore, different modifications of the reinitialization procedure, which have been proposed in literature and are aimed to reduce area loss, should be analysed and compared to each other. Moreover, new modifications could be found, which further improve the results obtained by the methods taken from literature. The analysis and comparisons should primarily concentrate on area loss, in addition to that, their effect on the level set's local curvature should also be taken into account because the level set method is often applied

to curvature-dependent problems.

The outline of this thesis is as follows:

In the first chapter an introduction to the level set method is given concerning mainly its historical development and its broad field of applications. In particular, the basic model for incompressible two-phase flow is explained.

The second chapter deals with different approaches for modelling the moving interface. The essential properties of a Lagrangian and an Eulerian formulation are compared, which motivate the use of the level set method.

In chapter three the level set method is described in detail and problems arising in practical computations are mentioned. Thus, the need for reinitialization is motivated and different approaches for reinitialization are discussed, which are based on the desired properties of the level set function.

The fourth chapter gives a brief overview of the main properties of Hamilton-Jacobi equations because Hamilton-Jacobi equations appear several times throughout the level set method.

Furthermore, chapter five is concerned with a discretization of the resulting partial differential equation (PDE), which is motivated by the properties of Hamilton-Jacobi equations mentioned in the previous chapter. In addition to that, a numerical method for the evaluation of curvature is explained.

While chapter six investigates the effect of area loss during numerical computations, different solution procedures aimed to avoid area loss are compared in chapter seven.

In chapter eight numerical results confirm the observations made in the previous chapters and single out the most convenient methods and schemes as far as area loss and computation of curvature are concerned.

Finally, chapter nine gives a short summary of the main results, which lead to the final conclusion concerning reinitialization of level set methods without area loss.

Chapter 1

Introduction

1.1 Level Set Methods

Originally, level set methods have been developed for front propagating problems by Osher and Sethian [17]. Mathematically, the propagating front plays the role of a free boundary in the sense of free boundary value problems. Thus, level set methods have been applied to many kind of problems involving moving boundaries such as crystal growth ([21] p.145), flame propagation [20], multi-phase flow [23], motion of soap bubbles [12], and motion of multiple junctions [15]. Additional difficulties are introduced because the speed of the moving boundary often depends on its local curvature [17]. Therefore, it is necessary to find an accurate approximation for the location and for the shape of the moving fronts. Besides applications such as mentioned above, of which the level set's interpretation is rather straight forward, level set methods are also involved in subjects of which its interpretation is not that obvious. For instance level sets are also used in image processing ([25] p.33).
In general, the basic idea of the level set method is to describe a $(n-1)$-dimensional hypersurface (representing the moving front) by the zero level set of a n-dimensional function, which is called the *level set function*. A more detailed discussion of the level set method and its major advantages as well as arising problems are respectively given in section 2.2 and chapter 3.
An important application for the level set method is two-phase flow. It is rather easy to investigate the arising problems having this kind of application in mind. Hence, the method and the arising problems will be introduced by a two-phase flow application.

1.2 Two-Phase Flow

Consider the motion of two immiscible, incompressible fluids in a domain Ω. The fluids are respectively described by their densities ρ_1 and ρ_2 and their dynamic viscosities μ_1 and μ_2. The interface Γ between the two fluids divides the domain Ω into two disjoint subdomains Ω_1 and Ω_2. Assume that Γ describes a closed curve

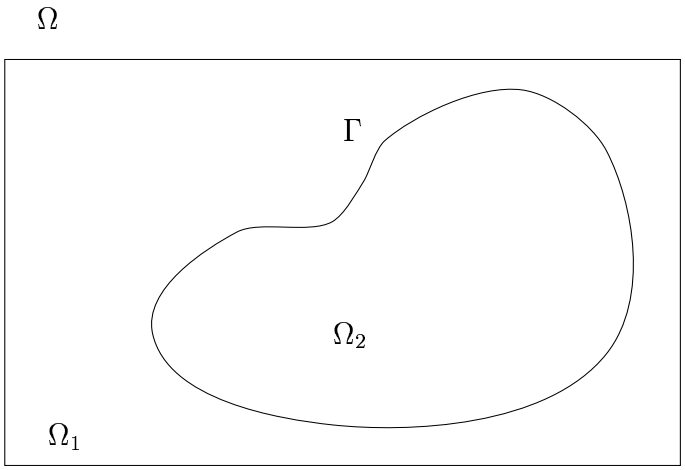


Figure 1.1: Computational domain Ω.

in Ω (see figure 1.1). According to that, the subdomain Ω_2 will be referred to as 'fluid bubble'. Since the fluids are assumed to be immiscible and incompressible, ρ and μ are functions on Ω, which are constant on each subdomain but with a jump at the interface Γ. The fluid bubble (i.e. Ω_2) moves with the flow according to its velocity field. Conversely, the bubble influences the velocity field by its shape.
For simplicity, assume that $\Omega \subset \mathbb{R}^2$. But in general, the methods discussed below shall also apply to the three-dimensional case.

1.3 The Mathematical Model

The motion of incompressible fluids is described by the incompressible Navier-Stokes equations (see [8] p.17):

$$\begin{aligned} \operatorname{div} u &= 0 \\ \rho(u_t + (u \cdot \nabla)u) &= -\nabla p + \rho g + \operatorname{div}(2\mu\mathcal{D}) \end{aligned}$$

where the following notations are used:

u : Velocity field of the flow
p : Pressure of the fluid
g : Gravitational constant
$\mathcal{D}$: Deformation tensor $= \frac{1}{2}(\mathrm{D}u + (\mathrm{D}u)^{\mathrm{T}})$

The meaning of the incompressibility condition (div $u = 0 \Leftrightarrow \nabla \cdot u = 0$) is expressed in [7] p.301 as follows:

> *In the case that* $\nabla \cdot u = 0$, [...], *which means that the area occupied by a given set of particles is constant in time. For this reason a velocity field* u *satisfying* $\nabla \cdot u = 0$ *is said to be incompressible.*

In particular, this means that the area (or volume in the three-dimensional case) of the fluid bubble Ω_2 remains *constant* throughout the calculation. However, in numerical computations this property is not fulfilled in general. Therefore, this thesis is mainly concerned with area conserving methods.

Since the fluids are immiscible, the densities and viscosities remain constant along trajectories of fluid particles:

$$\begin{aligned} \rho_t + \operatorname{div}(\rho u) &= 0 \\ \mu_t + \operatorname{div}(\mu u) &= 0 \end{aligned} \tag{1.1}$$

This means that the trajectories never cross the interface Γ. Since there are two different fluids separated by Γ into Ω_1 and Ω_2, there will respectively be a jump at the interface in ρ and μ. The effect of surface tension is to balance the jump in the normal stresses along the interface, which is proportional to the curvature κ and known as the Laplace-Young condition [9]:

$$[\tau_{ij} n_j]|_\Gamma = k\kappa n_i$$

where τ is the stress tensor, k is the surface tension coefficient, and n is the outer normal to the interface. This leads to a free boundary condition for the discontinuity in the normal stresses along the interface [3]. Therefore, the free boundary must be determined because it will change its position in time.
In order to compute the fluid's velocity u the location of the interface $\Gamma(t)$ and its local curvature κ must be determined (see equation (3.2) on page 18 for particular dependencies). Conversely, the underlying flow will change the position and shape of the interface, which again affects the fluid velocity. Thus, a description of the interface $\Gamma(t)$ for all $t > 0$ becomes necessary.

Chapter 2

Description of the Interface

It is very crucial how the interface is modelled mathematically because it will essentially determine the numerical methods that can be applied. In literature different approaches for the description of the interface can be found. All have their specific advantages and disadvantages. Hence, a short overview of the most important methods and their main properties is given below, which will motivate the use of the level set method.

2.1 Lagrangian Formulation - Front Tracking

The idea of a Lagrangian formulation is to describe the interface by a (closed) curve, which is given by a parameter representation. Hence, the physical coordinate system moves with the interface which is the reason for its name – Lagrangian formulation. The explicitly given curve is discretized by marker points that move through a fixed grid, which is used to compute the velocity field u. Since marker points are tracked on their way through the fixed grid, these methods are called *front tracking* methods. Boundary integral methods are of that kind, too [3]. This approach gives high-order accurate approximations to the moving interface without differentiating across the front [9]. Hence, it avoids introducing numerical diffusion, which smoothes out the front.
But this method has some serious drawbacks that lead to complicated numerical methods:

- Explicitly tracking the front is difficult for a complex geometry of the interface because it may be complicated to find a parameter representation.
- In addition to the grid for computing the velocity field u, numerical elements (marker points) for the discretization of the interface are necessary.
- It is difficult to continue tracking the front beyond some singularities in the interface (i.e. in the parametrization), which may occur in time [3].

- The numerical scheme becomes unstable since the grid points of the interface (marker points) may move so that the distance between neighbouring marker points decreases, i.e. that the mesh width Δx decreases. Conclusively, the time step Δt must be fit to the mesh width according to the CFL condition[1], which may become very small then [20]. Attacking this problem by a regridding technique will introduce diffusion, which will dominate the effects of curvature under consideration [17].

- The topological structure may change, e.g. when two bubbles merge. In that case corresponding marker points must be removed because the interface has vanished [2]. Further, if two bubbles are rather close together, the problem is to label the marker points to the correct bubble.

- These problems will be amplified if this method is implemented in three space dimensions [9].

2.2 Eulerian Formulation - Front Capturing

A method that avoids the main problems such as described above consists in describing the interface by means of a *fixed* co-ordinate system. Instead of tracking the interface through the computational domain explicitly, it will be implicitly captured on a fixed grid. Thus, methods based on an Eulerian formulation are called *front capturing* methods.

Volume of Fluid Methods

Volume of fluid methods differ significantly from the other methods discussed in this thesis because they do not describe the interface itself. They track the motion of the interior region by assigning to each cell on the computational grid a 'volume fraction' of the interior fluid instead. Hence, for each cell the size of the cell's part belonging to the bubble under consideration is known. Consequently, the volume of fluid method has two major advantages, which are named below [17]:

- In contrast to the parametrization no additional computational elements are needed. The grid, which is used to compute the velocity field u, may be used instead.

- Complicated topological boundaries are easily handled.

However, a serious drawback is the fact that it is very difficult to calculate the curvature of the interface from this kind of formulation. Therefore, it is difficult to apply this approach to curvature-dependent problems.

[1] See [14] p.100 for details concerning the CFL condition.

The Level Set Method

Another possibility to model the moving front, avoiding the problems of the Lagrangian formulation as well as taking curvature-dependencies into account, consists in describing the interface by the zero level set of a level set function, denoted as Φ. This means that the zero level set of the function is defined according to the position of the interface $\{x \in \Omega : \Phi(x) = 0\} := \Gamma$. At one side of the interface the level set function is defined larger than zero ($\Omega_1 =: \{x \in \Omega : \Phi(x) > 0\}$) and less than zero at the other side ($\Omega_2 =: \{x \in \Omega : \Phi(x) < 0\}$)[2]. Typically, the level set function is a Lipschitz-continuous function [23]. Motivated by the use of a level set as a description of the interface, this approach is known as the level set method.

On the one hand the level set method has some major advantages:

- In contrast to the Lagrangian formulation and coinciding with the volume of fluid method, no additional computational elements are used.
- Moreover, no explicit description (parametrization) of the interface is needed any more because it is implicitly captured on the Eulerian grid by the zero level set. This implies that the geometric characteristics of the interface are completely determined by the level set function.
- Therefore, complex interface structure and topological changes can be captured quite naturally because the interface is viewed as a level set [17]. Hence, it avoids the complex bookkeeping techniques, which plague front tracking methods.
- In addition to that, an extension to three space dimensions can be done rather easily.
- But the central advantage is that fixed-grid finite difference approximations may be used since no moving grid points are involved. Thus, no instability problems occur compared to front tracking methods.
- Finally, special properties of the interface such as local curvature or outer normal can be calculated easily from the level set function.

[2] In fact, the level set method is independent of the way of initialization the level set function. Alternatively, the role of the interior and exterior regions can be turned around, i.e. $\Omega_1 =: \{x \in \Omega : \Phi(x) < 0\}$ and $\Omega_2 =: \{x \in \Omega : \Phi(x) > 0\}$

On the other hand it is necessary to be aware of the disadvantages of the level set approach in order to find methods that reduce them as far as possible:

- High-order accuracy at the front - especially for incompressible two-phase flow with surface tension - is lost compared to front tracking methods [9].
- Furthermore, special care is needed in order to keep the thickness of the interface finite in time (see section 3.2 or [23]).
- The area of the fluid bubble must be conserved (see sections 1.3 and 3.3), which is not guaranteed in advance in numerical computations [23].
- Another important drawback is that computational expense is introduced additionally caused by the extension of one more space dimension. Recall that a $(n-1)$-dimensional curve (representing the interface) is described by the zero level set of a n-dimensional function (see section 3.6).

However, the disadvantages of the level set method does not seem to be as serious as the disadvantages introduced by the Lagrangian formulation, which can be avoided by the use of the level set method. In particular, the most important drawback of front tracking methods, which is finding the front, can be avoided. This is emphasized in [21] at p.68 by the following statement:

> *Nonetheless, finding the front is something that the level set scheme tries to avoid at all costs, since it introduces considerable complication to the technique.*

In addition to that, attempts to solve the problems introduced by an Eulerian formulation, at least partly, motivate to use the level set method instead of front tracking methods, which are based on a Lagrangian formulation.

Chapter 3

The Level Set Method

On the one hand the level set approach avoids some complex problems compared to front tracking methods as mentioned in the previous chapter. On the other hand the evolution of the level set function and in particular the evolution of the zero level set must be established according to the underlying flow. In addition to that, newly arising problems must be considered, which lead to the need of reinitialization of the level set function. Finally, different approaches for reinitialization, which are based on the desired properties of the level set function, are discussed in the proceeding chapter.

3.1 Level Set Formulation of the Problem

Using a level set approach as a description of the interface implies the existence of a mathematical relation between the evolution of the interface and the underlying flow field. The stipulation that the level sets always move according to the velocity field means that Φ does not change along any trajectory $x(t)$ (i.e. $\Phi(t, x(t)) = \text{constant}$). Thus, the following result is easily obtained by chain rule:

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{dt}}(\Phi(t, x(t))) &= 0 \\ \Rightarrow \Phi_t + \dot{x}(t) \cdot \nabla\Phi &= 0 \end{aligned} \tag{3.1}$$

This is a first-order partial differential equation of Hamilton-Jacobi type (see chapter 4). It moves the zero level set exactly as the interface moves with the flow, which is given by its velocity field $u \equiv \dot{x}(t)$. Therefore, equation (3.1) will be referred to as 'evolution equation for the interface'. Since the gradient $\nabla\Phi$ must be computed, a smooth function Φ is necessary. There is no problem to solve equation (3.1) numerically for Φ instead of using the *discontinuous* functions (1.1) for ρ and μ. This is true because $\rho(\Phi)$ can be expressed as $\rho(\Phi) = \rho_1 + (\rho_2 - \rho_1)H(\Phi)$ where H is the Heaviside function.

Modelling the effect of surface tension in terms of a singular source function modifies the Navier-Stokes equations as follows [23]:

$$\begin{aligned} \operatorname{div} u &= 0 \\ u_t + (u \cdot \nabla)u &= g + \tfrac{1}{\rho}(-\nabla p + \operatorname{div}(2\mu\mathcal{D}) + \sigma\kappa\delta(d)n) \end{aligned} \tag{3.2}$$

where σ denotes the surface tension, d the normal distance to the interface and δ the Dirac delta function. Hence, the correlation between the interface's position and shape on the one hand and the flow on the other hand, which was mentioned in the first chapter, becomes obvious.

Remark. Let F be the speed function of the interface in its outer normal direction. Thus,

$$F = \dot{x} \cdot n \tag{3.3}$$

where the outer normal is given by $n = \nabla\Phi/|\nabla\Phi|$. Substituting (3.3) into (3.1) yields

$$\Phi_t + F|\nabla\Phi| = 0$$

which is the level set equation introduced by Osher and Sethian in [17]. As already mentioned in chapter 1, some aspects of image processing are related to curve evolution. In particular, dilation and erosion belong to the class of the level set equations (see [25] p.35).

3.2 Properties of the Level Set Function

The first property is already mentioned above: Φ shall be a smooth function so that the gradient $\nabla\Phi$ can be computed. Moreover, curvature has to be calculated from the level set data, which implies the use of second-order derivatives (see section 3.7). Therefore, *'smoothness'* of the level set function is very crucial for the process.

Another important feature will be the saving of computational expenses if the level set function is computed only in a small neighbourhood around the zero level set. This is sufficient because only the position of the interface, which is given by the zero level set, and some of its properties such as local curvature, outer normal, and its gradient field need to be known. Therefore, this property will be called *'locality'* (see section 3.6).

Furthermore, it is important to keep the absolute value of the gradient of the level set function within reasonable bounds, i.e. the gradient should neither be too flat nor too steep. On the one hand computation of surface tension is difficult near a steep gradient [23]. On the other hand flat gradients may increase the width of

the interface because in numerical schemes grid points x_{ij} are assumed to belong to the interface if $\Phi(x_{ij}) \approx 0$. But this condition may be true for a rather thick layer of grid points around the zero level set if the gradients are too flat. This property will be referred to as *'bounded gradients'*.

Since the normal distance from the interface is used in the Navier-Stokes equations (3.2), it would be desirable that Φ is close to the *distance function* at least in a small neighbourhood of the interface, i.e. where $\delta(\Phi) \neq 0$.

Summary of the Properties

- The interface Γ is described as the zero level set $\{x \in \Omega : \Phi(x) = 0\}$ of a level set function.
- Let $\Phi(x) > 0$ if $x \in \Omega_1$ and let $\Phi(x) < 0$ if $x \in \Omega_2$ (or vice versa).
- Smoothness.
- Locality.
- Bounded gradients.
- Distance function.

3.3 Reinitialization

Even if Φ_0 is initialized as a distance function, the evolution equation (3.1) will destroy this property while moving it at the correct velocity u. Therefore, it may become 'irregular' in finite time.

- For example, steep gradients may occur when two bubbles merge, which affects the computation of surface tension, or flat gradients may occur, which lead to increasing width of the interface (see section 3.2).
- Moreover, the curvature of the level sets computed from an irregular level set function will be rather rough.
- Since it is desirable to compute the level set function only in a small neighbourhood around the zero level set (see section 3.2), there is a discontinuity along the boundary of the neighbourhood. The level set method will break down if the interface is close enough to this discontinuity (see section 3.6).
- Another important observation is that mass[1] of the fluid bubble is lost in time while applying equation (3.1) to the level set function. This obviously depends on its irregularity mentioned above [23].

[1] In fact, not mass is lost but the area of the fluid bubble shrinks (see section 6.2).

In order to avoid these problems, it is necessary to reconstruct the level set function so that it satisfies the properties mentioned above (section 3.2). The process that does the reconstruction is called *reinitialization* of the level set function. It turns out that reinitialization needs to be applied after a small number of evolution time steps.

3.4 Reconstruction of the Level Set Function

In order to find a convenient method to reinitialize the level set function, different approaches for reinitialization are compared. The desired properties of the level set function that will be fulfilled are mentioned, but the desired properties that will be violated are also taken into account.

Harmonic Approach

Since it is important to generate a *smooth* level set function, it is rather straight forward to solve the potential equation

$$\Delta\Phi(x) = 0, \qquad \forall x \in \Omega$$

with homogeneous Dirichlet boundary condition $\Phi(x) = 0$, $\forall x \in \Gamma$.

Although this approach creates a smooth function, it does not take into account locality property, which causes a lot of computational expenses. Moreover, it does not care about limits in the gradients. Therefore, it is quite inefficient to use this approach and may lead to rather thick interfaces due to flat gradients. In addition to that, an explicit description of the interface Γ is needed in order to impose boundary conditions at the interface, which contradicts the major advantage of the Eulerian formulation.

Eikonal Equation Approach

In [19] an iteration method based on the level set method was used to solve the following equation:

$$\begin{aligned} |\nabla\Phi| &= \lambda(x), \qquad \forall x \in \Omega_1 \quad \text{and} \quad \forall x \in \Omega_2 \\ \Phi &= 0, \qquad \forall x \in \Gamma \end{aligned}$$

Existence and uniqueness results for this equation are established in [19] and it is shown in [6] that monotone, stable, and consistent schemes converge to the viscosity solution[2]. This method was presented originally for shape from shading

[2]See chapter 4 for details concerning the notion of viscosity solutions.

applications, but if $\lambda \equiv 1$ it will be the Eikonal equation, which can be used for finding the distance function to the zero level set on one side on the interface. However, the distance function is needed on both sides of the interface. Conclusively, the Eikonal equation must be applied two times (first for Ω_1 and then for Ω_2).

Distance Function Approach

Consider the following partial differential equation, which appeared first in [23]:

$$\Phi_t = \text{sign}(\Phi_0)(1 - |\nabla\Phi|) \tag{3.4}$$

with initial condition $\Phi(x, 0) = \Phi_0(x)$ where Φ_0 defines the interface implicitly by $\Gamma =: \{x \in \Omega : \Phi_0(x) = 0\}$.

This is a non-linear hyperbolic PDE, which belongs to the class of Hamilton-Jacobi equations (see chapter 4 for details). The idea using this equation is that a steady-state solution of (3.4) will be a distance function for the zero level set (i.e. $|\nabla\Phi| = 1$; see [3]) with the same zero level set as the initial function Φ_0. Hence, the distance function may be computed for any given zero level set without changing its position. In contrast to the Eikonal equation one has to solve this PDE only once to obtain a solution on both sides of the interface and one need not to impose boundary conditions on Γ. This is exactly what is desirable for reinitialization: an equation that finds the distance function from the zero level set without moving the zero level set itself and without knowing explicitly the position of the zero level set.

Equation (3.4) can be rewritten as follows:

$$\Phi_t + w \cdot \nabla\Phi = \text{sign}(\Phi_0) \tag{3.5}$$

where

$$w = \text{sign}(\Phi_0)\frac{\nabla\Phi}{|\nabla\Phi|}$$

is a unit normal always pointing outward from the zero level set. Moreover, the characteristics are given by w since equation (3.5) has the form of a convection equation with velocity w. Hence, the characteristics are propagating away from the interface with speed one. But this means that corrections of the level set function (which are introduced at the zero level set) are propagating away from the interface in normal direction. If this equation is solved only for small times t, the distance function will be generated only in a small neighbourhood around the interface. More precisely, if the distance function needs to be computed within a neighbourhood around the interface of width $\alpha\Delta x$ where $\alpha > 0$, it is sufficient

to solve equation (3.4) up to time $t = 0..\alpha\Delta x$ [22]. Thus, the extension of one additional space dimension is compensated by computing the distance function only locally, i.e. around its zero level set.
Besides, a valid time step obeying the CFL condition is given by $\Delta t = \Delta x/2$ because the characteristics are propagating away from the interface with speed one.
The solution of (3.4) can be computed analytically by the method of characteristics, at least for small t and for smooth initial data Φ_0 (see [22] for details).

Remark. The method presented above leads to an iterative scheme for finding the distance function to the interface. Besides, direct methods exists (see [22] or [21] p.68) for finding the distance function, but the advantage of iterative methods is expressed in [22] as follows:

> *The advantage of an iterative scheme is the fact that if the interface moves a little, one can effectively use information from the previous value of distance to update the new values.*

In other words, the fact that Φ_0 is already a good guess for the distance function is effectively used by an iterative scheme to obtain convergence after a 'small' number of iterations.

3.5 Initialization

So far the evolution of the level set function and methods to maintain its desired properties have been discussed. But it is not clear how to generate a smooth initial function Φ_0 to be able to start the computation. Since the interface Γ may be a quite complicated curve in the computational domain $\Omega \subset \mathbb{R}^2$, this is not a trivial task.

Recall that it is possible to construct the distance function for any given zero level set by equation (3.4). Therefore, the problem is reduced to a construction of the zero level set itself. A straight forward approach is to initialize the computational domain Ω by a kind of 'characteristic function' Φ_0:

$$\Phi_0(x) := \begin{cases} 1; & x \in \Omega_1 \\ -1; & x \in \Omega_2 \end{cases}$$

Using Φ_0 defined above as initial data for equation (3.4) existence and uniqueness of the resulting solution cannot be expected because Φ_0 is not continuous (see [5]). Moreover, if a uniform mesh is considered for a discretization of Ω with initial data above an ill-posed problem will develop because the resulting zero level set may be anywhere between two neighbouring grid points that have been initialized by -1 and 1, respectively. In particular, no explicit zero level set is present at all, i.e. *grid*points x_{ij} satisfying $\Phi(x_{ij}) = 0$. To make sure that the resulting zero

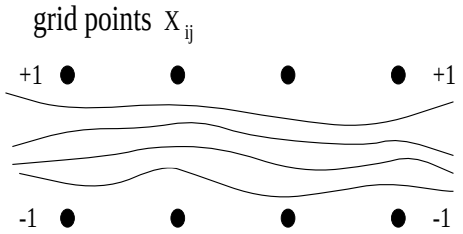


Figure 3.1: Possible positions of the zero level set after initialization.

level set lies somewhere between those neighbouring grid points that enclose the initial discontinuity, the grid points are manually forced to keep their sign (see section 7.1). However, numerical experiments show rather good results (figure 8.1). Since initialization is needed only once at the beginning of the computation, errors in accuracy will not too much influence the results.

3.6 Narrow Band Approach

It is sufficient to compute the level set function only in a small neighbourhood around its zero level set because only the zero level set is physically meaningful (representing the interface). Even if some properties of the interface need to be computed, it can be done from the level set function within the neighbourhood. Hence, locality is satisfied as it is put in section 3.2. The main reason to restrict the computations to such a narrow band around the zero level set is that much computational expenses can be saved. Thus, the additional afford introduced by the extension of one more space dimension by the level set method can be compensated at least partly. Let N be the number of grid points along one side of a quadratic computational grid and let k be the number of grid points within the narrow band. Then, computational expense[3] reduces from $O(N^2)$ to $O(kN)$ [1].

The narrow band approach works as follows: Only the values of the level set function at grid points within the band are updated by equation (3.1) to move the interface with the flow. Values of the boundary of the band are frozen. When the front moves near the boundary of the narrow band, this band must be redetermined according to the new position of the interface. In order to find the distance function within the new band reinitialization is performed as already indicated in section 3.3. Typically, the width of the band is approximately 12 to 24 grid points wide. For a more detailed explanation see [21] p.65 and [1].

Since the update during reinitialization of the level set function is performed from the zero level set and propagates in outer normal direction, it is possible to stop reinitialization if the corrections have reached a sufficiently large distance

[3]For two-dimensional computations.

from the interface. Furthermore, the distance function property of the level set function can be used to define the narrow band in a trivial way. Namely, the narrow band may be defined by the values of the level set function itself, i.e. grid points that have an absolute value less than some *bandwidth* > 0 are assumed to belong to the narrow band.

3.7 Calculation of Local Curvature

Since curvature-dependent evolution processes are considered, the ability to calculate the zero level set's local curvature is very important. This is achieved by the following equation:

$$\kappa = \nabla \cdot n \tag{3.6}$$

where κ denotes the level set's curvature and n the outer normal, which is given by $n = \frac{\nabla \Phi}{|\nabla \Phi|}$. A derivation of this equation can be found in [2]. The curvature can be expressed by Φ and its derivatives as follows [3]:

$$\kappa(\Phi) = \frac{\Phi_y^2 \Phi_{xx} - 2\Phi_x \Phi_y \Phi_{xy} + \Phi_x^2 \Phi_{yy}}{(\Phi_x^2 + \Phi_y^2)^{3/2}}$$

Since κ involves second-order derivatives, it is more sensitive to numerical errors in the level set function (see [11], [2] and figure 8.13). Therefore, it is very important to generate a very accurate and *smooth* level set function.

Chapter 4

Hamilton-Jacobi Equations

The level set method consists in solving Hamilton-Jacobi equations in two[1] essential parts. Therefore, it is necessary to discuss briefly their main properties. In particular, their close relation to hyperbolic conservation laws is considered.

4.1 Definition

Consider the so-called Cauchy problem for first-order partial differential equations of Hamilton-Jacobi type (see [5] and [6]), which are shortly called Hamilton-Jacobi equations (H-J equations).

$$\begin{aligned} u_t + H(\nabla u) &= 0, && \text{in } \mathbb{R}^n \times (0,\infty) \\ u(x,0) &= u_0(x), && \text{in } \mathbb{R}^n \end{aligned}$$

Basically, all H-J equations considered in this thesis are of that kind. Therefore, it seems to be convenient to summarize briefly its main properties and some basic numerical approaches to compute approximate solutions. A more detailed and general discussion including existence, uniqueness, and stability proofs for smooth Hamiltonian H and smooth initial data u_0 is given by Crandall, Lions, and Evans in [5], [6], and [4].

4.2 Mathematical Properties

It is well-known that H-J equations do not have classical solutions [11]. Therefore, only weak solutions can be expected. Unfortunately, weak solutions are not uniquely defined in general. Therefore, the notion of *viscosity solutions* is introduced in [5], which is a weak solution satisfying some entropy conditions (compare [14] p.36) aimed to single out the physically relevant solution among

[1] Evolution of the level set function according to the underlying flow field by equation (3.1) and reinitialization by equation (3.4).

all weak solutions. Thus, viscosity solutions are uniquely defined. Moreover, viscosity solutions are proven to be stable with respect to small perturbations in the computational data. The name, viscosity solution, refers to the 'vanishing viscosity' approach, where the limiting solution of the viscous equation[2] for the viscous term tending to zero[3] is determined ([14] p.26). Although the formal definition of a viscosity solution for H-J equations is different, it is proven by Crandall and Lions [5] to be this limit (see also [21] p.84).

Typically, solutions of H-J equations are continuous but with discontinuous derivatives even for smooth initial data u_0. This is similar to the analysis of hyperbolic conservation laws, where even discontinuities (shocks) may develop from a $\mathbb{C}^\infty$ function as an initial condition. In addition to that, an analytical relation exists between H-J equations and hyperbolic conservation laws. In case of one space dimension the H-J equation $(u_t + H(\nabla u) = 0)$ can be regarded as the 'once integrated' conservation law $(\tilde{u}_t + (H(\tilde{u}))_x = 0)$ with respect to x putting $\tilde{u} \equiv u_x$ if the solution u is twice continuously differentiable.

$$\begin{aligned} \tilde{u}_t + (H(\tilde{u}))_x &= 0, \quad \int \; . \; dx \\ \int u_{tx} \; dx + H(u_x) &= 0 \\ u_t + H(u_x) &= 0 \end{aligned}$$

Although such a relation does not exist in the multi-dimensional case, it emphasizes the close connection between these kind of partial differential equations.

4.3 Basic Ideas for Numerical Methods

The first approaches solving H-J equations are done in [4]. The authors show that monotone schemes converge to the viscosity solution. Unfortunately, monotone schemes are at most first-order accurate ([14] p.170). Furthermore, traditional high-order methods introduce oscillations in the presence of discontinuous derivatives [18]. However, based on the relation above between hyperbolic conservation laws and H-J equations, high-order numerical methods, which are originally designed for hyperbolic conservation laws, have been adapted to H-J equations. Two high-order methods are presented in the following chapter.

[2] Adding a viscous right-hand side (i.e. RHS= $\epsilon \triangle u, \epsilon > 0$) to the partial differential equation.
[3] Thus, $\epsilon \to 0$.

Chapter 5

Discretization

Based on the observations from the previous chapter a discretization technique of various order of accuracy, designed originally for hyperbolic conservation laws, is adapted to H-J equations in this chapter. In particular, the discretization of the basic initial value problem (3.4) for reinitialization is explained. Moreover, a numerical method for computing the level set's local curvature is mentioned.

5.1 Semi-Discrete Methods

The resulting partial differential equation (3.4) that will be used for reinitialization should be applicable for two and three space dimensional problems. Moreover, the solution should be obtained by a high-order method. Therefore, the discretization process is split into two parts. Firstly, only the space derivatives are discretized while the time dependency is left continuous. Thus, an ordinary differential equation (ODE) is obtained, which is called 'semi-discrete equation'. Finally, any high-order ODE solver may be applied to discretize the temporal derivative.
Since the space and time dependencies are decoupled, it is much easier to obtain high-order accuracy. Moreover, this approach, which is sometimes also called 'method of lines', will be useful if two or more space dimensions are under consideration (for more details see [14] p.193).

5.2 Discretization in Space

ENO Schemes

In order to avoid oscillations in the presence of non-smooth data using higher-order schemes, so-called essentially non-oscillatory (ENO) schemes have been constructed by Harten, Osher, and Shu. Originally, they have been designed for hyperbolic conservation laws. Motivated by the close relation (see chapter 4)

between H-J equations and hyperbolic conservation laws, Osher and Sethian [17] extended ENO schemes in order to apply them to H-J equations achieving very good numerical results. In [18], Osher and Shu provided a more general ENO scheme construction procedure, which is explained in this section by an application to equation (3.4) and (7.1)[1].

The main idea for approximating the partial derivatives $\Phi_x(x_{ij})$ and $\Phi_y(x_{ij})$ (for simplicity in notation only one space dimension will be considered from now on) is to construct stencil interpolating polynomials $P^{\Phi,r}(x_i)$ of degree r as an approximation of the function Φ. Hence, the derivative $\frac{d}{dx}P^{\Phi,r}(x_i)$ of this polynomial is used as an approximation of the partial derivative of the level set function, $\Phi_x(x)$. In particular, a forward and a backward approximation is computed for each grid point x_i. For the forward approximation at x_i a stencil interpolating polynomial $P^{\Phi,r}_{i+1/2}(x)$ valid on the interval $[x_i, x_{i+1}]$ is computed and Φ_x is approximated by $\Phi^+_x(x_i) := \frac{d}{dx}P^{\Phi,r}_{i+1/2}(x_i)$. Analogously, $P^{\Phi,r}_{i-1/2}(x_i)$ is computed on $[x_{i-1}, x_i]$ for the backward approximation $\Phi^-_x(x_i) := \frac{d}{dx}P^{\Phi,r}_{i-1/2}(x_i)$.

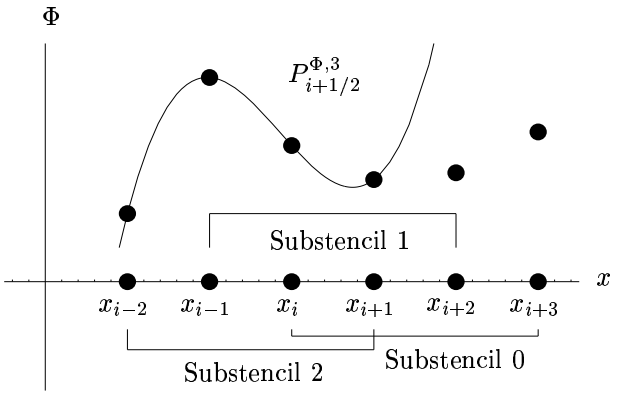


Figure 5.1: Stencil interpolating polynomial $P^{\Phi,3}_{i+1/2}$ used for the forward approximation of the partial derivative $\Phi^+_x(x)$ at $x = x_i$. Substencil 2 is selected by the stencil choosing process out of three possible substencils.

The freedom in choosing the additional stencil points for the forward and backward approximations (if the degree of the polynomial is bigger than one) is used to obtain information always from the locally smoothest region of the function. This is achieved by selecting this grid point as an additional stencil point of which the approximated derivative has the smallest absolute value. See [18] and [11] for further details.
The construction works as follows: Define undivided differences $\phi(j,k)$ for each space direction.

$$\begin{aligned} \phi(j,0) &:= \Phi(x_j) \\ \phi(j,k) &:= \phi(j+1,k-1) - \phi(j,k-1), \quad k = 1,\ldots,r \end{aligned}$$

[1]Details are given only for equation (3.4), but equation (7.1) works in a similar way.

Hence, $\phi(j,0)$ approximate the function Φ and $\phi(j,k)$ its kth derivative in $[x_j, x_{j+k}]$. The stencil choosing process for the forward approximation of the partial derivative $\Phi_x^+(x_j)$, according to the smoothness property mentioned above, works in the following way.

$$\begin{array}{lcll} i(j) & := & j, & \forall \text{ stencil points } j \\ i(j) & := & i(j)-1, & \text{if } \quad |\phi(i(j),k)| > |\phi(i(j)-1,k)| \end{array} \tag{5.1}$$

for $k = 2, \ldots, r$, where $i(j)$ is the leftmost point in the stencil for $P^{\Phi,r}_{j+1/2}(x)$. Corresponding to figure 5.1 substencil 0 is selected by default (since $i(j) := j$). If the absolute value of the second derivative[2] at grid point x_{j-1} is smaller than the one[3] at x_j (i.e. $|\phi(i(j),2)| > |\phi(i(j)-1,2)|$), then substencil 1 will be selected ($i(j) := i(j)-1$). This process continues until $k = r$.
Finally, the derivative of the stencil interpolating polynomial is given by

$$\Phi_x^+(x_i) = \frac{1}{\Delta x} \sum_{k=1}^{r} c(i(j)-j,k)\phi(i(j),k)$$

where

$$c(m,k) = \frac{1}{k!} \sum_{s=m}^{m+k-1} \prod_{\substack{l=m \\ l \neq s}}^{m+k-1} (-l)$$

Note that the matrix $(c)_{mk}$ is independent of Φ. Hence, it has to be computed only once and may be stored for later use. The computation of the backward approximation is done in a similar way.
In this thesis, ENO schemes of second, third, and fourth order are used as an approximation of the partial space derivatives. Only equal mesh widths $\Delta x = \Delta y$ are considered.

Weighted ENO Schemes

A further extension of ENO schemes are weighted ENO (WENO) schemes initiated by Liu et al. and improved by Jiang and Shu, who both approximated hyperbolic conservation laws, too. Jiang and Peng [11] applied them to H-J equations.
In contrast to the ENO scheme above, which selects *one* (the smoothest) substencil to construct the interpolating polynomial, does the WENO scheme take use of *all* possible substencils and weights them according to the local smoothness of the function. In other words, r different polynomials (for each substencil

[2] $\phi(j-1,2)$ which is valid on $[x_{i-1}, x_{i+1}]$.
[3] $\phi(j,2)$ which is valid on $[x_i, x_{i+2}]$.

one polynomial) are computed for each grid point and a convex combination or weighted average of all polynomials is used. Therefore, WENO schemes may be considered as 'central' schemes in regions where the solution is smooth. Thus, it is much more accurate as the original ENO scheme. As shown in [11]:

> [This WENO scheme] *which is the 5th order approximation to* $\Phi_x(x_i)$ [the partial derivative] *and is known to provide the smallest truncation error on such a sixpoint stencil.*

If there are singularities of the solution, the WENO scheme will adapt its basic ENO scheme to avoid oscillations. In order to achieve these features the weights are defined according to the following two principles (see [11] for details):

- If Φ is smooth on the whole stencil, information from *all* possible substencils will be used to approximate the derivative.
- If the stencil contains a singularity of Φ, the weights adaptively approach the ENO '1-0-weights' to avoid oscillations.

5.3 Approximation of the Hamiltonian

Consider the H-J equation:

$$\begin{aligned} \Phi_t + H(\Phi_x, \Phi_y) &= 0, && \text{in } \mathbb{R}^2 \times (0,\infty) \\ \Phi(x,y,0) &= \Phi_0(x,y), && \text{in } \mathbb{R}^2 \end{aligned}$$

The semi-discrete equation is obtained by approximating the Hamiltonian H by a Lipschitz-continuous monotone flux function $\hat{H}(\Phi_x^+, \Phi_x^-, \Phi_y^+, \Phi_y^-)$, which is consistent with H, i.e. $\hat{H}(u,u,v,v) = H(u,v)$. Monotonicity means that $\hat{H}$ is non-increasing in its first and third argument and non-decreasing in its other arguments [17]. A variety of monotone flux functions is given in [17]. As pointed out in [11] the Godunov flux $\hat{H}^G$ is rather simple for equation (3.4), namely:

$$\hat{H}^G := \begin{cases} s\sqrt{(\max\{(\Phi_x^+)^-, (\Phi_x^-)^+\})^2 + (\max\{(\Phi_y^+)^-, (\Phi_y^-)^+\})^2} - 1 & ;\Phi_0 \geq 0 \\ s\sqrt{(\max\{(\Phi_x^+)^+, (\Phi_x^-)^-\})^2 + (\max\{(\Phi_y^+)^+, (\Phi_y^-)^-\})^2} - 1 & ;\Phi_0 < 0 \end{cases} \tag{5.2}$$

where $s = \text{sign}(\Phi_0)$, $(a)^+ = \max\{a, 0\}$, and $(a)^- = -\min\{a, 0\}$.

However, in [22] another approach is applied to choose the appropriate approximation of the partial derivative. The authors used the upwind method because equation (3.4) can be written in the form of a convection equation with characteristics propagating outward from the zero level set (see section 3.4). LeVeque ([14] p.112) explained the method and its name in the following way:

> *This method is usually called [..] upwind method, since the one-sided stencil points in the 'upwind' or 'upstream' direction, the correct direction from which characteristic information propagates. If we think of the advection equation as modelling the advection of a concentration profile in a fluid stream, then this is literally the upwind direction.*

Roughly speaking, the upwind method 'differentiates' always towards the characteristics' direction. Hence, the upwind method uses backward approximations if the characteristics are propagating from below to the point under consideration, while it uses forward approximations if the characteristics are propagating from above. The method works as follows [22]:

$$\Phi_x \approx \begin{cases} \Phi_x^+ & ; \Phi_x^+ \text{sign}(\Phi_0) < 0 \quad \text{and} \quad (\Phi_x^- + \Phi_x^+)\text{sign}(\Phi_0) < 0 \\ \Phi_x^- & ; \Phi_x^- \text{sign}(\Phi_0) > 0 \quad \text{and} \quad (\Phi_x^- + \Phi_x^+)\text{sign}(\Phi_0) > 0 \\ 0 & ; \Phi_x^- \text{sign}(\Phi_0) < 0 \quad \text{and} \quad \Phi_x^+ \text{sign}(\Phi_0) > 0 \end{cases} \tag{5.3}$$

Φ_y is approximated in a similar way. Conclusively, the Hamiltonian is approximated by using the corresponding forward and backward approximations. Moreover, it is possible to show that $\hat{H}^G$ is equivalent to the upwind method (5.3):

Theorem. *Consider the initial value problem (3.4).*
Then the Godunov method (5.2) is equivalent to the upwind method (5.3).

Proof. For simplicity, consider only the partial derivatives with respect to x. Godunov's method (5.2) can be rewritten as follows:

$$\Phi_x \approx \begin{cases} \max\{-\min\{\Phi_x^+, 0\}, \max\{\Phi_x^-, 0\}\} & ; \Phi_0 \geq 0 \\ \max\{\max\{\Phi_x^+, 0\}, -\min\{\Phi_x^-, 0\}\} & ; \Phi_0 < 0 \end{cases}$$

1. Let the Godunov method choose $\Phi_x^+ \neq 0$.
   (a) Let $\Phi_0 > 0$: $\Rightarrow$ $\Phi_x^+ < 0$ and $(-\Phi_x^+ > \Phi_x^-$ or $\Phi_x^- < 0)$
   $\Rightarrow$ $\Phi_x^+ \text{sign}(\Phi_0) < 0$ and $\Phi_x^- + \Phi_x^+ < 0$
   Thus, the upwind method chooses also Φ_x^+. ✓
   (b) Let $\Phi_0 < 0$: $\Rightarrow$ $\Phi_x^+ > 0$ and $(-\Phi_x^- < \Phi_x^+$ or $\Phi_x^- > 0)$
   $\Rightarrow$ $\Phi_x^+ \text{sign}(\Phi_0) < 0$ and $\Phi_x^- + \Phi_x^+ > 0$
   Thus, the upwind method chooses also Φ_x^+. ✓
2. Let the Godunov scheme choose $\Phi_x^- \neq 0$:
   (a) Let $\Phi_0 > 0$: $\Rightarrow$ $\Phi_x^- > 0$ and $(-\Phi_x^+ < \Phi_x^-$ or $\Phi_x^+ > 0)$
   $\Rightarrow$ $\Phi_x^- \text{sign}(\Phi_0) > 0$ and $\Phi_x^- + \Phi_x^+ > 0$
   Thus, the upwind method chooses also Φ_x^-. ✓
   (b) Let $\Phi_0 < 0$: $\Rightarrow$ $\Phi_x^- < 0$ and $(-\Phi_x^- > \Phi_x^+$ or $\Phi_x^+ < 0)$
   $\Rightarrow$ $\Phi_x^- \text{sign}(\Phi_0) > 0$ and $\Phi_x^- + \Phi_x^+ < 0$
   Thus, the upwind method chooses also Φ_x^-. ✓

3. Let the Godunov scheme choose zero:

   (a) Let $\Phi_0 > 0$: $\Rightarrow$ $\Phi_x^+ > 0$ and $\Phi_x^- < 0$
   $\Rightarrow$ $\Phi_x^+ \text{sign}(\Phi_0) > 0$ and $\Phi_x^- \text{sign}(\Phi_0) < 0$
   Thus, the upwind method chooses also zero. ✓

   (b) Let $\Phi_0 < 0$: $\Rightarrow$ $\Phi_x^+ < 0$ and $\Phi_x^- > 0$
   $\Rightarrow$ $\Phi_x^+ \text{sign}(\Phi_0) > 0$ and $\Phi_x^- \text{sign}(\Phi_0) < 0$
   Thus, the upwind method chooses also zero. ✓

1. Conversely, let the upwind method choose $\Phi_x^+ \neq 0$.

   (a) Let $\Phi_0 > 0$: $\Rightarrow$ $\Phi_x^+ < 0$ and $\Phi_x^- + \Phi_x^+ < 0$
   But this implies that $\Phi_x^- < -\Phi_x^+$.
   Thus, Godunov's method chooses also Φ_x^+. ✓

   (b) Let $\Phi_0 < 0$: $\Rightarrow$ $\Phi_x^+ > 0$ and $\Phi_x^- + \Phi_x^+ > 0$
   But this implies that $-\Phi_x^- < \Phi_x^+$.
   Thus, Godunov's method chooses also Φ_x^+. ✓

2. Let the upwind method choose $\Phi_x^- \neq 0$.

   (a) Let $\Phi_0 > 0$: $\Rightarrow$ $\Phi_x^- > 0$ and $\Phi_x^- + \Phi_x^+ > 0$
   But this implies that $-\Phi_x^+ < \Phi_x^-$.
   Thus, Godunov's method chooses also Φ_x^-. ✓

   (b) Let $\Phi_0 < 0$: $\Rightarrow$ $\Phi_x^- < 0$ and $\Phi_x^- + \Phi_x^+ < 0$
   But this implies that $\Phi_x^+ < -\Phi_x^-$.
   Thus, Godunov's method chooses also Φ_x^-. ✓

3. Let the upwind method choose zero.

   (a) Let $\Phi_0 > 0$: $\Rightarrow$ $\Phi_x^- < 0$ and $\Phi_x^+ > 0$
   $\Rightarrow$ $\max\{\Phi_x^-, 0\} = 0$ and $\min\{\Phi_x^+, 0\} = 0$
   Thus, Godunov's method chooses also zero. ✓

   (b) Let $\Phi_0 < 0$: $\Rightarrow$ $\Phi_x^- > 0$ and $\Phi_x^+ < 0$
   $\Rightarrow$ $\max\{\Phi_x^+, 0\} = 0$ and $\min\{\Phi_x^-, 0\} = 0$
   Thus, Godunov's method chooses also zero. ✓

□

Remark. The same analysis applies for the partial derivative with respect to y. Moreover, the case where $\Phi_0 = 0$ is trivial because the Hamiltonian H will be zero for equation (3.4).

5.4 Discretization in Time

Runge-Kutta type procedures for the time-discretization are used to obtain Φ^{n+1} from Φ^n, which means proceeding one time step. Define

$$\begin{aligned} L_{ij}^{(0)} &:= -\Delta t \hat{H}(\Phi_x^+, \Phi_x^-, \Phi_y^+, \Phi_y^-) \\ \Phi_{ij}^{(0)} &:= \Phi_{ij}^n \end{aligned} \tag{5.4}$$

where $\hat{H} := \text{sign}(\Phi_0)(1 - \sqrt{(\Phi_x^{\pm})^2 + (\Phi_y^{\pm})^2})$ and compute

$$\Phi_{ij}^{(k)} = \sum_{l=0}^{k-1} (\alpha_{kl}\Phi_{ij}^{(l)} + \beta_{kl}L_{ij}^{(l)}), \quad k = 1, \ldots, r \tag{5.5}$$

Finally, obtain $\Phi_{ij}^{n+1} := \Phi_{ij}^{(r)}$ where r denotes the level of the Runge-Kutta method and α_{kl} and β_{kl} are defined as shown in figure 5.2 for different order of accuracy. Up to order 4 the level of the Runge-Kutta method may be equal to the desired order of approximation using the schemes below.

Order	α_{kl}	β_{kl}
2	1 $\frac{1}{2}\quad\frac{1}{2}$	1 $0\quad\frac{1}{2}$
3	1 $\frac{3}{4}\quad\frac{1}{4}$ $\frac{1}{3}\quad 0\quad\frac{2}{3}$	1 $0\quad\frac{1}{4}$ $0\quad 0\quad\frac{2}{3}$
4	1 $\frac{1}{2}\quad\frac{1}{2}$ $\frac{1}{9}\quad\frac{2}{9}\quad\frac{2}{3}$ $0\quad\frac{1}{3}\quad\frac{1}{3}\quad\frac{1}{3}$	$\frac{1}{2}$ $-\frac{1}{4}\quad\frac{1}{2}$ $-\frac{1}{9}\quad-\frac{1}{3}\quad 1$ $0\quad\frac{1}{6}\quad 0\quad\frac{1}{6}$

Figure 5.2: Three TVD Runge-Kutta schemes.

The following statement taken from [18] gives that the Runge-Kutta method (5.5) is total variation diminishing (TVD)[4]:

> *The method (5.5) can be proven TVD under the CFL condition* $\lambda = \Delta t/\Delta x \leq c\lambda_0$ *if the Euler forward version of (5.4) is TVD under the CFL condition* $\lambda = \Delta t/\Delta x \leq \lambda_0$.

Where $c = 1$ for the 2nd and 3rd order version and $c = 2/3$ for the 4th order version.

[4]See [14] p.165 for a more detailed investigation of TVD methods.

5.5 Discretization of Curvature

Central differences are used to compute local curvature since the level set function is assumed to be smooth at least in the neighbourhood of the zero level set. In literature ([3], [2]) two slightly different approximation methods can be found. But the results do not differ very much. The scheme found in [3] works as follows:

$$\kappa(\Phi(x_{ij})) = \frac{(D_y^0\Phi_{ij})^2 D_x^- D_x^+\Phi_{ij} - 2D_x^0\Phi_{ij}D_y^0\Phi_{ij}D_x^0D_y^0\Phi_{ij} + (D_x^0\Phi_{ij})^2 D_y^- D_y^+\Phi_{ij}}{((D_x^0\Phi_{ij})^2 + (D_y^0\Phi_{ij})^2)^{3/2}}$$

where the forward, backward, and centred differences are defined as follows:

$$\begin{aligned} D_x^+\Phi_{ij} &:= (\Phi(x_{i+1,j}) - \Phi(x_{i,j}))/h \\ D_x^-\Phi_{ij} &:= (\Phi(x_{i,j}) - \Phi(x_{i-1,j}))/h \\ D_x^0\Phi_{ij} &:= (\Phi(x_{i+1,j}) - \Phi(x_{i-1,j}))/(2h) \end{aligned}$$

5.6 Smoothing of Discontinuous Functions

Smoothed out approximations for the sign-, Heaviside-, and Dirac delta function are used in order to prevent the schemes from unwanted instabilities at the interface (see [3] and [22]):

$$\begin{aligned} H_\epsilon(x) &\equiv \begin{cases} 0 & ; x < -\epsilon \\ \frac{x+\epsilon}{2\epsilon} + \frac{\sin(\pi x/\epsilon)}{2\pi} & ; |x| \le \epsilon \\ 1 & ; x > \epsilon \end{cases} \\ \delta_\epsilon(x) &\equiv \begin{cases} \frac{1}{2\epsilon}(1 + \cos(\frac{\pi x}{\epsilon})) & ; |x| < \epsilon \\ 0 & ; \text{otherwise} \end{cases} \end{aligned}$$

$$\text{sign}_\epsilon(x) \equiv 2(H_\epsilon(x) - 1/2)$$

This definition implies a prescribed thickness of the interface of order ϵ and satisfies the conditions $\frac{d}{dx}H_\epsilon(x) = \delta_\epsilon(x)$ and $\int_{-\epsilon}^{\epsilon}\delta_\epsilon(x)dx = 1$. In fact, only the smoothed out sign function[5] is used during reinitialization. The other functions are given for completeness, too.

[5]The Dirac delta function is used only in the constraint of equation (7.2).

Chapter 6

Analysis of Area Loss during Reinitialization

A serious problem of the level set method occurring in numerical computations is area loss during reinitialization (see [23] or [3]). Therefore, the numerical scheme that is used for reinitialization is analysed more detailed in order to find reasons for area loss.

6.1 Preliminary Remarks

Since incompressible fluids are under consideration, the area (or volume in the 3d case) enclosed by the zero level set is equivalent to the fluid's mass. Therefore, it is justified to speak about mass conservation instead of area conservation and both expressions are equivalently used throughout this context, as it was already done in earlier publications (e.g. [23] or [3]).

Theoretically, equation (3.4) used for reinitialization conserves the area of the domain bounded by the curve defined implicitly by $\Phi_0(x, y) = 0$ because (3.4) does not change the position of the boundary (i.e. the zero level set, since $\text{sign}(0) = 0$). But unfortunately, in numerical computations this is not true any more because there are hardly any grid points x_{ij} satisfying $\Phi(x_{ij}) = 0$.
In fact, numerical experiments show that a considerable amount of mass is always *lost* during reinitialization, i.e. that the area enclosed by the zero level set shrinks (see [23], [3] and figure 8.4). Since reinitialization is applied periodically and the error tends always to one direction qualitatively (area loss), it does accumulate from application to application. Hence, it is very important to find numerical schemes that reduce area loss as much as possible or to find methods that preserve area conservation during reinitialization.

Note that area loss during reinitialization is another kind of area loss occurring in the level set method. It is important not to mix up area loss by reinitialization (i.e. introduced by equation (3.4)) and area loss mentioned in section 3.3 that

is introduced by the evolution of the fluid bubble in time using equation (3.1). Recall that the latter kind of area loss was one of the main reasons to reinitialize the level set function. Qualitatively, the problem is only transferred from equation (3.1) to equation (3.4). However, a quantitative improvement is achieved as far as area conservation is concerned (see [23]). Moreover, the other difficulties that require reinitialization can be handled. Thus, reinitialization quantitatively improves the numerical results.

6.2 Area Computation on a Discrete Grid

In order to be able to find methods that conserve area it is important to know how the area of the fluid bubble is computed on a discrete grid. Consider a quadratic grid cell c_{ij} of area $(\Delta x)^2$, which is assigned to each grid point x_{ij} so that x_{ij} is the grid point in the lower left corner of the cell c_{ij}. Assume that the interface divides the cell c_{ij} into two parts so that $\Phi(x_{ij}) < 0$ and the values of the other grid points are larger than zero, i.e. that x_{ij} belongs to the fluid bubble and the other grid points belong to the outer fluid as it is illustrated in figure 6.1.

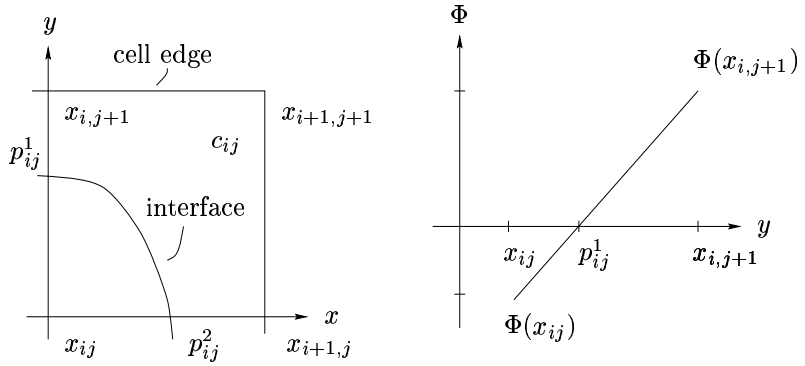


Figure 6.1: LEFT: Quadratic grid cell c_{ij}. RIGHT: Zero determination of the level set function at the left edge of the grid cell c_{ij} by linear interpolation.

All other possible situations are handled in a similar way as far as the position of the interface is concerned. These positions, denoted by p^1_{ij} and p^2_{ij}, are computed from (linear) interpolation of the corresponding grid points by a determination of its zero. The position of the interface at the cell itself may be computed by spline or polynomial interpolation of desired order. But the effect of area loss during *reinitialization* depends only on the computed position of the interface on cell *edges* p^k_{ij}, $k = 1, 2$ and is independent[1] of the reconstruction of the interface across the cell. Thus, area loss is reflected in the movement of the interface's position p^k_{ij}, $k = 1, 2$ at cell edges. This movement is introduced by the evolution

[1]Since the reconstruction of the interface across the cell depends only on the stencil points that are the zeros of the level set function determined at cell edges.

of $\Phi(x_{ij})$ and $\Phi(x_{i,j+1})$ during reinitialization, which may be so that the positions p_{ij}^k, $k = 1, 2$ change.

Remark. Alternatively, the volume fraction of the bubble at one cell may be computed by bilinear reconstruction of Φ at the cell, which is more sophisticated. However, the method that will develop by the analysis above yields also good results if it is applied to methods using bilinear reconstruction.

6.3 Geometric Dependencies of Area Loss

In order to identify the reasons for area loss it seems convenient to consider a one-dimensional problem first. But it turns out that no area loss occurs. Moreover, it is possible to show that area loss does not occur in the one-dimensional case even if the very basic first-order scheme is used.

Theorem. *Consider the first-order (i.e. forward and backward finite differences) upwind method (5.3) and the forward Euler discretization in time. Suppose that the space dimension is one, that the level set function is symmetric around its zero and monotonously increasing. Then no area loss occurs during reinitialization by equation (3.4) if the zero is computed by linear interpolation.*

Proof. It is sufficient to show that symmetry is preserved. Assume that the zero of the level set function lies between the grid points x_i and x_{i+1} (i.e. $\Phi(x_{i+1}) = -\Phi(x_i) > 0$). Define $\Delta x := x_{i+1} - x_i > 0$. Remember the definitions of the forward and backward finite differences:

$$\Phi_x^+(x_i) := \frac{\Phi(x_{i+1}) - \Phi(x_i)}{\Delta x} \geq 0 \qquad \Phi_x^-(x_{i+1}) := \frac{\Phi(x_{i+1}) - \Phi(x_i)}{\Delta x} \geq 0$$

The upwind method (5.3) chooses Φ_x^+ at x_i and Φ_x^- at x_{i+1} because it 'differentiates towards the zero level set'. Thus, the values changes during one time step (from Φ^n to Φ^{n+1}) as follows:

$$\begin{aligned}
\Phi^{n+1}(x_i) &= \Phi^n(x_i) - \Delta t(1 - |\Phi_x^+(x_i)|) \\
&= \Phi^n(x_i) - \lambda(\Delta x - (\Phi^n(x_{i+1}) - \Phi^n(x_i))) \\
&= (1 - \lambda)\Phi^n(x_i) - \lambda\Delta x + \lambda\Phi^n(x_{i+1})
\end{aligned}$$

$$\begin{aligned}
\Phi^{n+1}(x_{i+1}) &= \Phi^n(x_{i+1}) + \Delta t(1 - |\Phi_x^-(x_{i+1})|) \\
&= \Phi^n(x_{i+1}) + \lambda(\Delta x - (\Phi^n(x_{i+1}) - \Phi^n(x_i))) \\
&= (1 - \lambda)\Phi^n(x_{i+1}) + \lambda\Delta x + \lambda\Phi^n(x_i)
\end{aligned}$$

where $\lambda \equiv \frac{\Delta t}{\Delta x}$ denotes the grid ratio, which is assumed to be less than 0.5 according to the CFL condition. Since $|\Phi^n(x_i)| = |\Phi^n(x_{i+1})|$ is assumed, symmetry

is preserved. Moreover, since $\Phi^n(x_{i+1}) = -\Phi^n(x_i)$, it follows that $\Phi^{n+1}(x_i) = (1-2\lambda)\Phi^n(x_i) - \lambda\Delta x$, which implies that $\Phi^n(x_i) \to -\Delta x/2$ for $n \to \infty$. Analogously, it follows that $\Phi^n(x_{i+1}) \to \Delta x/2$ for $n \to \infty$. □

However, area loss is observed in two-dimensional computations. Therefore, the next step to be considered will be a two-dimensional example. In particular, a straight line representing the interface is reinitialized. Again area is conserved. Hence, area loss must depend on the bends of the interface. In order to investigate its influence consider an interface with one single bend. Numerical computations always show the same results: The corners are smoothed out so that the bends become smooth, convex curves of which the local curvature decreases during the iteration. Thus, the interface always moves into the convex part of the domain (see figure 8.2). Tests show that this movement is independent of initializing Φ to be larger or less than zero inside the bubble. The reason for that is given below.

6.4 Approximation of the Gradient

Even if the exact distance function (which can be determined analytically – for a circle at least) is reinitialized by equation (3.4) area is lost in time. Therefore, the effect of area loss must have to do with the accuracy of the approximation of the absolute value of the gradient $|\nabla\Phi|$.

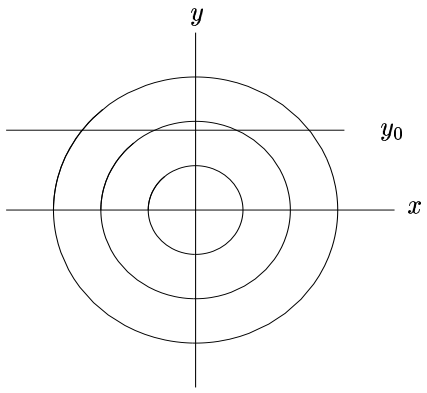


Figure 6.2: Contour plot of the level set function and labelled line $y \equiv y_0$ for the conic section.

In order to analyse this effect consider the distance function for a unit circle of radius one centred at $(0,0)$ with $\Phi < 0$ inside and $\Phi > 0$ outside the circle. The level set function reads then: $\Phi(x,y) = \sqrt{x^2+y^2} - 1$, which already is the desired steady-state solution of equation (3.4). Thus, the iteration should stop immediately. Now, fix $y \equiv y_0$ (see figure 6.2) and consider the conic section for a particular y_0, i.e. consider Φ as a function of x only, which reads then $\Phi(x;y_0) = \sqrt{x^2+y_0^2} - 1$ and is shown in figure 6.3 for different y_0.

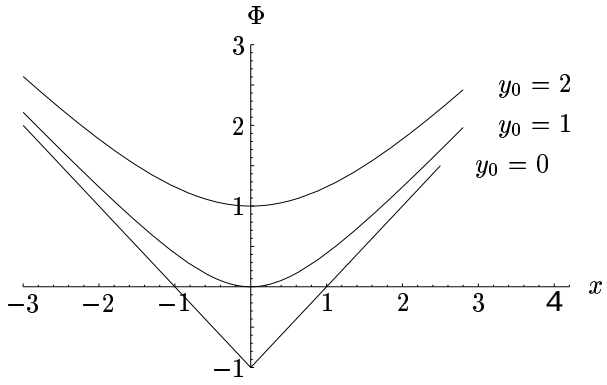


Figure 6.3: Three conic sections $\Phi(x; y_0)$ for $y_0 = 0, 1, 2$.

From figure 6.3 the following observations can be done concerning the conic sections of the level set function:

1. Large $|x|$ leads to small deviation from a straight line, i.e. it leads to small curvature[2] of the conic section. This observation holds for all y_0.

2. For $y_0 = 0$ the conic section consists of two straight lines.

3. Moreover, a minimum in the conic section at $x = 0$ is observed for all y_0.

Compare the standard forward ($\Phi_x^+(x; y_0)$) and backward ($\Phi_x^-(x; y_0)$) finite differences with the exact partial derivative Φ_x (see figure 6.4). The following observations can be done:

1. $|\Phi_x^+(x; y_0)| > |\Phi_x(x; y_0)| > |\Phi_x^-(x; y_0)|$ if $x > 0$ (and vice versa if $x < 0$)[3] , i.e. that the backward difference underestimates and the forward difference overestimates the absolute value of the partial derivative (if $x > 0$).

2. Quantitatively, the approximation error of the partial derivative is increasing with increasing curvature of the conic section because the curvature measures the deviation of a curve from a straight line, for which the finite difference is an exact estimate.

3. Furthermore, the accuracy of the approximation depends on y_0 because $y_0 = 0$ implies zero curvature ($\Phi_{xx} \equiv 0$, for $x \neq 0$ and see figure 6.3) and, thus, no approximation error of Φ_x is introduced by the finite differences.

[2]The curvature of the conic section is given by $\kappa = \frac{\Phi_{xx}}{(1+\Phi_x^2)^{3/2}}$ with $\Phi_{xx} = \frac{y_0^2}{(x^2+y_0^2)^{3/2}}$.

[3]In general, this case distinction respectively corresponds to a monotonously increasing and decreasing level set function.

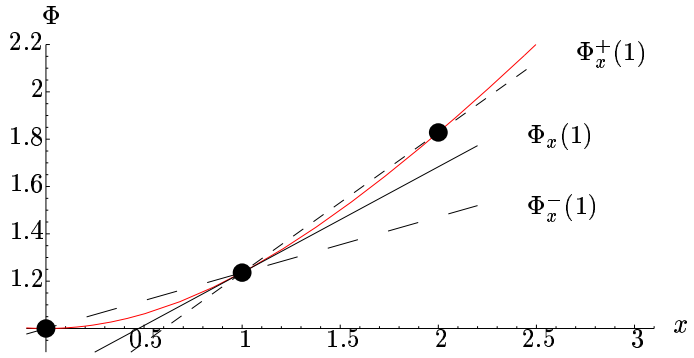


Figure 6.4: Comparison among forward and backward finite differences ($\Phi_x^+(x)$ and $\Phi_x^-(x)$) and the partial derivative ($\Phi_x(x)$) at $x = 1$.

The relationship between straight lines and curvature used in the second observation from figure 6.4 is expressed in [13] p.47 as follows[4]:

> *So mißt also* κ [die Krümmung] *gewissermaßen die Abweichung der Kurve vom geradlinigen Verlauf an der betreffenden Stelle. Die Bezeichnung „Krümmung“ ist demnach gerechtfertigt.*
>
> *Thus,* κ [curvature] *measures the deviation of the curve from its straight way at the corresponding place. Therefore, the notion 'curvature' is justified.*

The first and essential observation from figure 6.4 holds for all monotone, convex functions and will be proven below:

Theorem. *Let* $a < x_{i-1} < x_i < x_{i+1} < b$, $\Delta x := x_{i+1} - x_i = x_i - x_{i-1}$, *and let* $\Phi : [a, b] \to \mathbb{R}$ *be a convex function that is differentiable in* (a, b) *and strictly monotonously increasing. Then*

$$|\Phi^+(x_i)| > |\Phi'(x_i)| > |\Phi^-(x_i)|$$

where $\Phi^{+/-}$ *denote the standard forward and backward finite differences, respectively.* Φ' *denotes the first derivative of* Φ.

Proof. The definition of the standard backward finite difference gives

$$\Phi^-(x_i) := \frac{f(x_i) - f(x_{i-1})}{\Delta x} = \Phi'(\xi^-)$$

using the mean value theorem where $\xi^- \in (x_{i-1}, x_i)$. By the same analysis it follows that $\Phi^+(x_i) = \Phi'(\xi^+)$ where $\xi^+ \in (x_i, x_{i+1})$. Since Φ is convex, which implies that the derivative is monotonously increasing, and $\Phi' > 0$ (Φ is strictly monotonously increasing) the statement follows from the fact that $\xi^- < x_i < \xi^+$. □

[4]The translation is done by myself and shall at least semantically return the statement's meaning

If Φ' is assumed to be negative, i.e that Φ is strictly monotonously decreasing the result will be the following achieved by the same analysis:

$$|\Phi^+(x_i)| < |\Phi'(x_i)| < |\Phi^-(x_i)|$$

These are exactly the same results, which have been claimed above.

The following conclusions can be drawn by the observations above:

The accuracy of the finite differences is improved with increasing $|x|$. In particular forward differences are more accurate than backward differences if $x > 0$ and vice versa otherwise.

Fix $x = x_0$ and consider $\Phi(y; x_0)$ as a function of y only. The same analysis leads to analogous results for the approximation of the partial derivative with respect to y. Therefore, if the error that is introduced by differentiation with respect to y is considered additionally, the result will qualitatively remain unchanged, but it will be amplified in magnitude.

Since the upwind method (see section 5.3) 'differentiates' always towards the zero level set, the essential result of this section is that this approximation always *overestimates* the absolute value of the gradient *inside* and *underestimates* it *outside* the unit circle. Thus, a systematic error is introduced during reinitialization.

According to the second observation from figure 6.3, Φ_x will be computed exactly if $y_0 = 0$. Moreover, if y is equal to zero, no forward or backward approximation of Φ_y will be used, but the partial derivative with respect to y will be approximated by zero according to the upwind method. This will also be an exact solution, which can be seen by the third observation of figure 6.3. Thus, *no* error will be introduced at all if the gradient is approximated at $y = 0$. Exchanging the role of x and y leads to the same result at $x = 0$.

Remark. Although the observations that have been done above are based on a finite difference approximation of first order, they are qualitatively equivalent for ENO and WENO approximations in numerical computations. However, this remains to be proven in general. Nevertheless, some heuristics for this relation are given below for the second-order ENO interpolation.

Transference to ENO schemes

Assume that the level set function is already close to the distance function and consider the conic section at $y_0 = 0.5$ of a unit circle around $(0, 0)$. Then, the conic section reads: $\Phi(x, y_0) = \sqrt{x^2 + y_0^2} - 1$
In figure 6.5 the partial derivatives with respect to x of various order are plotted together with the zero of the conic section (which is labelled by a black circle). Obviously, the absolute value of the partial derivative (of order two or higher) is decreasing in the neighbourhood of the zero (black circle). Thus, the stencil

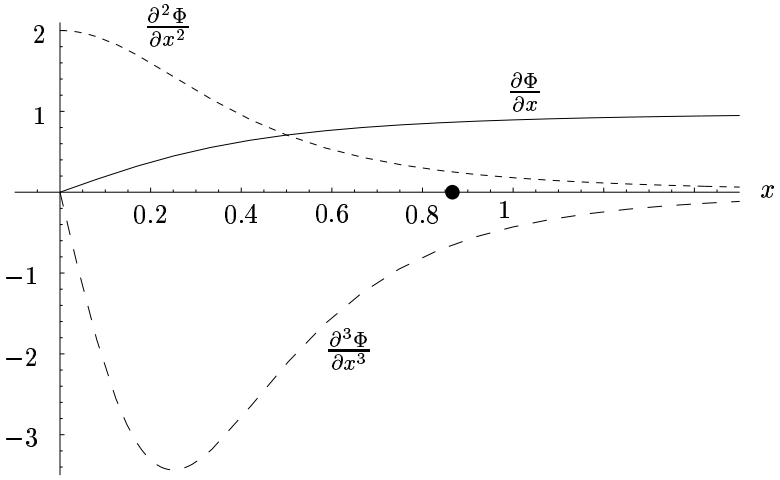


Figure 6.5: Derivatives of the conic section of various order and the zero of the conic section (labelled by a black circle).

choosing process (see equation 5.1 p. 29) will select the *rightmost* substencil to construct the stencil interpolating polynomial. Hence, the polynomials used for the forward and backward approximation[5] differ only in one stencil point. In particular, the leftmost stencil point in the forward and backward approximation at x_i will respectively be x_i and x_{i-1}. For simplicity, consider a polynomial of degree two and a constant grid width Δx. The polynomials read then:

$$\begin{aligned} P_{i+1/2}(x) &= f[x_i] + f[x_i, x_{i+1}](x - x_i) + f[x_i, x_{i+1}, x_{i+2}](x - x_i)(x - x_{i+1}) \\ P_{i-1/2}(x) &= f[x_{i-1}] + f[x_{i-1}, x_i](x - x_{i-1}) + f[x_{i-1}, x_i, x_{i+1}](x - x_{i-1})(x - x_i) \end{aligned}$$

where $f[x_{i-1}, \ldots, x_{i+k}]$ denote the usual Newton's kth divided difference. Conclusively, the approximations for the partial derivatives read:

$$\begin{aligned} (P_{i+1/2})'(x_i) &= f[x_i, x_{i+1}] + f[x_i, x_{i+1}, x_{i+2}](x_i - x_{i+1}) \\ (P_{i-1/2})'(x_i) &= f[x_{i-1}, x_i] + f[x_{i-1}, x_i, x_{i+1}](x_i - x_{i-1}) \end{aligned} \tag{6.1}$$

As already mentioned in chapter 5 the kth divided difference approximates the kth derivative. Therefore, it may be possible to conclude from figure 6.5 the following relations:

$$\begin{aligned} a^+ := f[x_i, x_{i+1}] &> a^- := f[x_{i-1}, x_i] > 0 \\ 0 < b^+ := f[x_i, x_{i+1}, x_{i+2}] &< b^- := f[x_{i-1}, x_i, x_{i+1}] \end{aligned} \tag{6.2}$$

Substituting these conclusions into equation (6.1) leads to the following result:

$$\begin{aligned} (P_{i+1/2})'(x_i) &= a^+ - \Delta x b^+ \\ (P_{i-1/2})'(x_i) &= a^- + \Delta x b^- \end{aligned} \tag{6.3}$$

[5]Coinciding with chapter 5 the polynomials for the forward and backward approximation at x_i are respectively denoted by $P_{i+1/2}$ and $P_{i-1/2}$.

Recall that the following relation holds by definition of the divided differences using the definitions above:

$$b^- = \frac{a^+ - a^-}{2\Delta x} \tag{6.4}$$

Consider the following calculations using equation (6.4) in the last step:

$$\begin{aligned} & \tfrac{1}{2}(a^+ + a^-) &=& \tfrac{1}{2}(a^+ + a^-) \\ \Leftrightarrow\ & a^+ - \tfrac{1}{2}(a^+ - a^-) &=& a^- + \tfrac{1}{2}(a^+ - a^-) \\ \Leftrightarrow\ & a^+ - \Delta x b^- &=& a^- + \Delta x b^- \end{aligned}$$

Substituting the relation between b^+ and b^- from equation (6.2) into the equation above yields:

$$\begin{aligned} & a^+ - \Delta x b^+ &>& a^+ - \Delta x b^- &=& a^- + \Delta x b^- \\ \Leftrightarrow\ & |(P_{i+1/2})'(x_i)| &>& |(P_{i-1/2})'(x_i)| \end{aligned}$$

which is similar to the result obtained for the first-order finite difference approximation in the previous section. Moreover, it is possible to conclude from equation (6.3) that the gap in the absolute value between the forward and backward approximations using the first-order scheme is larger than the gap using the second-order ENO scheme. Thus, the systematic error remains qualitatively equivalent but it is quantitatively damped.

6.5 Area Loss by Error in the Gradient

Consider the influence of the systematic error above on the evolution of the zero level set by the following analysis. Without loss of generality, assume that $x > 0$ and define:

$$\begin{aligned} |\Phi_x^+(x; y_0)| - |\Phi_x| &=: \epsilon_x^+ > 0 \\ |\Phi_x| - |\Phi_x^-(x; y_0)| &=: \epsilon_x^- > 0 \end{aligned}$$

According to the observation above ϵ must be defined vice versa for $x < 0$ in order to get $\epsilon > 0$, i.e. $\epsilon_x^+ := |\Phi_x| - |\Phi_x^+(x; y_0)|$.

Consider two grid points that enclose the interface, i.e. consider $\Phi(x_i) < 0$ and $\Phi(x_{i+1}) > 0$. Assume that Φ is initialized as the correct signed distance from the interface (figure 6.6), i.e. that $|\nabla\Phi| = 1$. Consider the error that is introduced by the finite difference approximation of the partial derivative with respect to x and compute the first time step of reinitialization using the first-order scheme in space and time mentioned above:

$$\Phi^{(n+1)}(x_i) = \Phi^{(n)}(x_i) + \Delta t\ \operatorname{sign}(\Phi^{(0)}(x_i))(1 - (|\Phi_x^{+/-}(x_i)|))$$

$$\begin{aligned} \Phi^{(1)}(x_i) &= \Phi^{(0)}(x_i) - \Delta t(1 - (|\Phi_x^+|)) &= -\tilde{d} + \Delta t \epsilon_x^+ \\ \Phi^{(1)}(x_{i+1}) &= \Phi^{(0)}(x_{i+1}) + \Delta t(1 - (|\Phi_x^-|)) &= d + \Delta t \epsilon_x^- \end{aligned}$$

where d and $\tilde{d}$ respectively denote the distance of x_i and x_{i+1} to the interface. Since $\Phi_x^+(x_i) = \Phi_x^-(x_{i+1})$ it follows that $\epsilon_x^+ = \epsilon_x^- =: \epsilon$. Hence, from figure 6.6 follows how the shift 'upwards' of the level set function leads to a movement of the interface (i.e. the zero) *towards* the centre[6] of the circle.

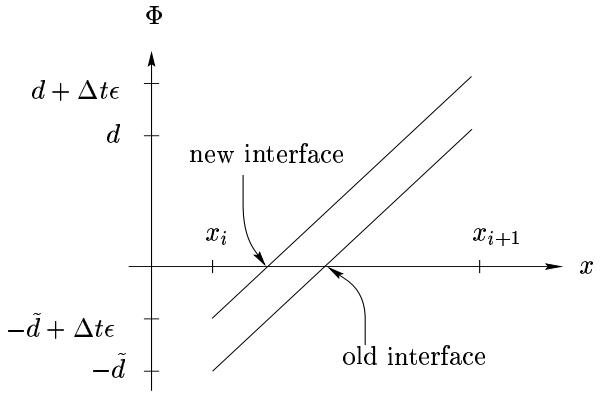


Figure 6.6: Area loss introduced by the systematic error in the approximation of the partial derivative.

This means that it moves into the 'convex part' of the domain as it is put above. Quantitatively, area loss increases with increasing approximation error ϵ of the gradient. But this error is not uniformly distributed. Recall that the error will be zero along the lines $x = 0$ and $y = 0$. Conclusively, area loss should not occur along those lines (see figure 8.3, page 55). Thus, there should be regions of the circle where area loss is worse than elsewhere.

Since every boundary can be approximated locally by a circle, these results apply for arbitrary fronts. Moreover, area is always *lost* during reinitialization because every closed curve is more convex than concave.

Concerning the observations about area conservation in case of a straight line representing the interface in two space dimensions[7], the following argumentation applies: The distance function consists of two half planes, which are not curved, i.e. that the curvature is zero. Thus, no approximation error of the gradient will be introduced. This may explain that no area loss occurs in such a case.

As far as initialization of the level set function is concerned, the same results are obtained in numerical experiments, independently of initializing $\Phi > 0$ or $\Phi < 0$ inside the circle. Therefore, assume the level set function to be initialized

[6]Since the level set function is assumed to be initialized less than zero inside the circle (see page 38).

[7]As well as for the one-dimensional case (both mentioned in section 6.3).

conversely, i.e. larger than zero inside and less than zero outside the circle. Since $\text{sign}(\Phi_0)$ is conversely defined inside and outside the bubble, the shift of the level set function will be turned around (moving 'downwards'). However, the effect of area loss will be maintained because the level set function is turned around, too.

All together, the essential result of this section is that area is always *lost* if the domain is more convex than concave, independently of the way of initialization.

Remark. A straight forward approach to attack this problem works as follows: Since forward approximations always overestimate[8] the absolute value of the partial derivative while backward approximations always underestimate it, it seems convenient to use central schemes instead. Indeed, central schemes (WENO schemes) work much more accurate than the purely upwinded forward/ backward schemes (see section 8.2).

6.6 Area Gain by Large Corrections

The observations above may lead to the conclusion that total area can be conserved if the partial derivatives can be computed exactly. Since the distance function for a unit circle centred at $(0,0)$ is known analytically ($\Phi(x,y) = \sqrt{(x^2+y^2)} - 1$), it is possible to proceed one time step using the exact partial derivatives[9]. But the numerical computation shows that area is not conserved, either. Assume that the level set function is initialized by the doubled distance from the interface. Thus, the absolute value of the gradient is equal to two. Following the calculations above we get the following results in the first time step:

$$\begin{aligned} \Phi^{(1)}(x_i) &= \Phi^{(0)}(x_i) - \Delta t(1-2) &&= -\tilde{d} + \Delta t \\ \Phi^{(1)}(x_{i+1}) &= \Phi^{(0)}(x_{i+1}) + \Delta t(1-2) &&= d - \Delta t \end{aligned}$$

Again, d and $\tilde{d}$ denote the distance to the interface of the corresponding grid points.
The large deviation of the gradient compared to the desired gradient introduces a large correction[10] of the level set function, which may[11] lead to a movement of the interface, that will increase the area. This situation is illustrated in figure 6.7. Obviously, these two kinds of errors superpose each other. Therefore, 'area gain' is sometimes observed at the beginning of the computation if the gradients are steep (or flat) enough to dominate the effects of the approximation error of the

[8] If Φ is monotonously increasing.

[9] The partial derivative with respect to x of the distance function for the unit circle around (0,0) reads: $\Phi_x = \frac{x}{\sqrt{x^2+y^2}}$

[10] Therefore, the approximation error ϵ of the gradient may be neglected.

[11] Obviously, area gain and area loss depend on whether $d > \tilde{d}$ or vice versa.

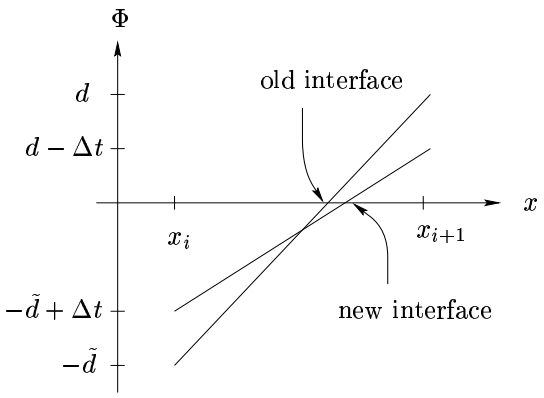


Figure 6.7: Area gain introduced by large corrections of the level set function.

partial derivatives. Moreover, if the corrections are sufficiently large, it is possible that the value of the level set function will change its sign at grid points adjacent to the interface. Consequently, this will lead to a decreasing or increasing area enclosed by the zero level set.

However, the influence of large corrections decreases as the steady-state is approached (because the gradients will tend to one). Therefore, it is clear that area is always lost in the end because the error introduced by the approximation of the partial derivatives still affects the computations.

Chapter 7

Methods Preserving Area Conservation

While reasons and effects influencing area conservation have been established in the previous chapter, different methods that are designed to preserve area conservation are now compared. Moreover, a new method designed to reduce are loss is proposed, which is based on the observations in section 6.2.

7.1 Reset Procedure

One possibility that is commonly used to prevent area loss is to fix lowest bounds for $|\Phi(x)|$ in advance, i.e. that no grid point must change its sign. This will be referred to as *reset* procedure because it checks whether a grid point has changed its sign. In that case the value of the grid point is reset to the correct sign (e.g. if $\Phi(x_{ij}) = 0.5$ has changed its sign from minus to plus, it will be put to $\Phi(x_{ij}) := -\epsilon$, $\epsilon > 0$ and small) manually. Therefore, the loss of mass is limited to a maximum amount, which depends on the mesh width Δx. But this is not accurate enough. Alternatively, the value can be reset to half the value of its previous value, which seems to be slightly more sophisticated because it takes its previous value into account instead of putting it to a constant regardless of its original value. However, the method is still very rough.

7.2 Area Conserving PDE Approach

Another way to overcome this problem is to solve the following perturbed Hamilton-Jacobi equation to steady-state, which was introduced by Hou in [3]:

$$\Phi_t = -(v_0 - v(t))(-P + \kappa)|\nabla\Phi| \tag{7.1}$$

where $v(t)$ denotes the volume of the bubble at time $t > 0$ and v_0 at time $t = 0$, κ denotes the curvature of the level sets, and P is a stability constant. The idea

of using this equation is that a steady-state solution will have the property that $v_0 = v(t)$ or that the curvature is equal to P. But at least the latter property will be difficult to obtain for geometries that do not have constant curvature. This procedure would not only prevent area loss during reinitialization but it would also correct area loss introduced by the evolution of the level set function according to the underlying flow using equation (3.1).

As far as solvability, motivation, and well-posedness of this equation are concerned, Chang and Hou noticed the following in [3]:

> *Motivated by the observation that numerical diffusion introduces a normal motion proportional to the interface's local curvature (see e.g. [15]), he* [Hou] *introduced a re-initialization procedure to remedy this effect* [area loss]. *[...] The above perturbed Hamilton-Jacobi equation may look ill-posed by itself. But since we solve the perturbed Hamilton-Jacobi equation with the governing level set equation and the solution procedure of the governing level set equation introduces numerical viscosity, the combined fractional step method can be shown to be stable.*

The discretization of (7.1) is done similarly to the discretization of equation (3.4), but with a more general flux function[1]. For details about these general flux functions see [18].

7.3 Constrained Distance Function Approach

Recently Sussman [22] proposed a constraint for equation (3.4) used for reinitialization in order to conserve the volume enclosed by the zero level set during the computation. This is achieved by requiring that

$$\frac{\partial}{\partial t}\int_{\tilde{\Omega}} H(\Phi) = 0$$

where H denotes the Heaviside function and $\tilde{\Omega}$ is any fixed domain. Equation (3.4) is modified as follows:

$$\Phi_t = \operatorname{sign}(\Phi_0)(1 - |\nabla\Phi|) + \lambda f(\Phi) \tag{7.2}$$

where f is any function and λ is a function of t only determined by requiring that

$$\frac{\partial}{\partial t}\int_{\tilde{\Omega}} H(\Phi) = \int_{\tilde{\Omega}} H'(\Phi)\Phi_t = \int_{\tilde{\Omega}} H'(\Phi)(\mathrm{L}(\Phi_0, \Phi) + \lambda f(\Phi)) = 0$$

where L is defined to be $\mathrm{L}(\Phi_0, \Phi) := \operatorname{sign}(\Phi_0)(1-|\nabla\Phi|)$. Therefore, λ is calculated to be

$$\lambda = \frac{-\int_{\tilde{\Omega}} H'(\Phi)\mathrm{L}(\Phi_0, \Phi)}{\int_{\tilde{\Omega}} H'(\Phi) f(\Phi)}$$

[1] Recall that the upwind method presented in section 5.3 fits only to equation (3.4).

where f is chosen as follows:

$$f(\Phi) \equiv H'(\Phi)|\nabla\Phi|$$

which insures that the correction is applied only at the interface without disturbing the distance function property distant from the interface. Note that if (7.2) is solved exactly, then λ will be zero [22]:

> *This is because* $\mathrm{L}(\Phi_0, \Phi)$ *as it appears in (7.2) will be zero in regions where* $H'(\Phi)$ *is not zero (the zero level set of* Φ *).*

Following the discretization in [22], the constraint is applied only once per time-step. Thus, for high-order Runge-Kutta methods $\tilde{\Phi}^{n+1}$ is obtained as a solution of the unconstrained equation (3.4) from Φ^n. Compute Φ^{n+1} from $\tilde{\Phi}^{n+1}$ by

$$\Phi^{n+1} = \tilde{\Phi}^{n+1} + \Delta t \;\; \lambda_{ij} \;\; H'_{\Delta x}(\Phi_0) \;\; |\nabla\Phi_0|$$

where λ is discretized as follows

$$\lambda_{ij} = \frac{-\int_{\tilde{\Omega}_{ij}} H'_{\Delta x}(\Phi_0)\frac{\tilde{\Phi}^{n+1}-\Phi^n}{\Delta t}}{\int_{\tilde{\Omega}_{ij}} (H'_{\Delta x}(\Phi))^2|\nabla\Phi_0|}$$

The numerical integration over the domain

$$\tilde{\Omega}_{ij} = \{(x,y) : x_{i-1/2} < x < x_{i+1/2} \;\text{ and }\; y_{j-1/2} < y < y_{j+1/2}\}$$

is computed using a nine point stencil

$$\int_{\tilde{\Omega}_{ij}} g \approx \frac{h^2}{24}(16g_{ij} + \sum_{\substack{m=-1 \\ m\neq 0}}^{1} \sum_{\substack{n=-1 \\ n\neq 0}}^{1} g_{i+m,j+n})$$

7.4 Extrapolation Method for the Interface

A straight forward way to preserve area conservation during reinitialization by applying equation (3.4) is motivated by the analysis of area loss in section 6.2. In order to avoid unwanted movements of the zeros (i.e. of the zero level set), linear extrapolation of the adjacent grid points is used to keep the zeros fixed.

The algorithm works as follows and is illustrated in figure 7.1:

- Firstly, reinitialization is applied only on grid points that are negative (i.e. that are inside the fluid bubble). Thus, $\Phi^{k+1}(x_i)$ is obtained from $\Phi^k(x_i)$, for all $\Phi^k(x_i) < 0$.
- Secondly, the corresponding grid point outside that is adjacent to the interface ($\Phi^{k+1}(x_{i+1})$ in figure 7.1) is extrapolated so that the position of the interface does not change.
- Finally, the *remaining* outer grid points are updated according to the resulting gradient field of the level set function.

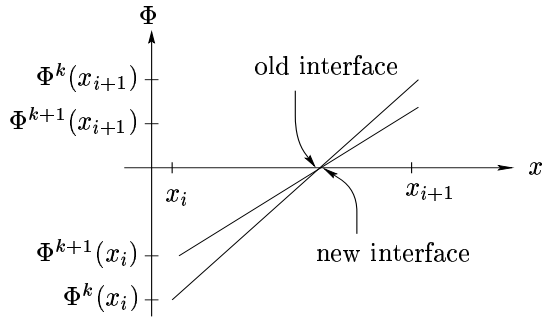


Figure 7.1: Extrapolated value $\Phi^{k+1}(x_{i+1})$.

However, some difficulties occur for curved interfaces. If the interface is not parallel to the grid, some exterior grid points adjacent to the interface may be determined by *two* corresponding interior grid points. In figure 7.2 this situation is labelled by 'filled circular' grid points, which have extrapolated values that are determined by two 'filled quadratic' grid points. Hence, this may lead to two *different* extrapolated values for the same grid point. In that case, the zeros will be moved so that the *volume* of the corresponding grid cells c_{ij} and $c_{i-1,j-1}$ remains constant[2]. But in general the volume of the *neighbouring* grid cell $c_{i-1,j}$ cannot be conserved. Thus, total conservation of the area enclosed by the zero level set is not possible. But the error will be small and by turning around the

[2]The reconstruction of the interface across the grid cell is done by piecewise linear interpolation of the corresponding zeros of the level set function at the edges of the grid cell.

role of interior and exterior grid points[3] in the next time step the grid cells that have not been conserved yet ($c_{i-1,j}$) will be conserved in the following time step and vice versa.

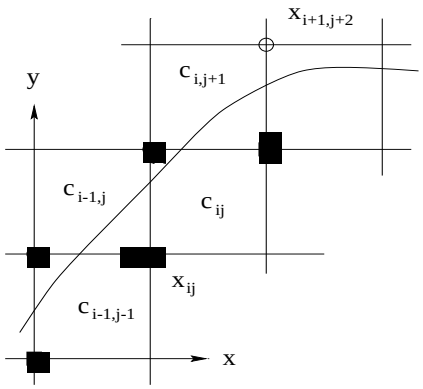


Figure 7.2: Computational grid including grid cells where area is conserved and where area is not conserved during one iteration step. Grid points that are twice determined by two quadratic grid points are labelled by a filled circle. The unfilled circle labels a grid point that is adjacent to a twice determined grid point but is not twice determined itself. Extrapolation is not applied at such a grid point.

In addition to that, a grid point that is adjacent to such a twice determined extrapolated grid point and is *not* twice determined itself will not be extrapolated because numerical experiments show better results as far as smoothness of curvature is concerned. In figure 7.2 such a grid point ($x_{i+1,j+2}$) is labelled by an 'unfilled' circle. This effect may be explained by the freedom, which is given to the interface at grid cells where the curvature may change (grid cell $c_{i,j+1}$ in figure 7.2).
This approach uses three essential properties of equation (3.4), which is used for reinitialization:

1. The corrections of the level set function will decrease as the solution approaches the steady-state. Hence, the corrections introduced by the extrapolation procedure will become smaller, too.

2. The corrections by the extrapolation method of the level set function are introduced at the zero level set. However, the corrections by the iteration of equation (3.4) are also introduced at the zero level set.

3. The characteristics of equation (3.4) will transport the 'noise' introduced by the extrapolation method away from the interface (see section 3.4).

[3] Apply reinitialization to all *exterior* grid points and extrapolate the values at interior grid points adjacent to the interface.

These facts together give hope to obtain an *area conserving* and *smooth* steady-state solution, which is very important for computation of the gradient and even more for computation of curvature. Although total conservation of area is not always possible in general, the error that remains will be very small. Moreover, the error becomes even smaller during the computation (see figure 8.11 and 8.12) by the fact mentioned above. Crucial to this process is to determine the zeros of the level set function always from the initial data Φ_0 and not from the solution of the previous time step of reinitialization, Φ^{n-1}, because the area of the *initial* data should be conserved. In addition to that, it is of prime importance to prevent a grid point from changing its sign, as described in section 7.1, because the interpolation method will break down in such a case. Therefore, reset procedure is applied additionally. In particular, the corresponding level set value will not be put to a fixed value ϵ of the desired sign, but it will be put to half the value of its previous value.

Chapter 8

Numerical Results

The schemes and methods discussed in the previous chapters have been implemented and their effects on reinitialization are compared to each other. The comparisons are done with respect to area loss and smoothness of the computed solution, which is visualized by the smoothness of the level set's local curvature. In order to get a first impression of a level set function, figure 8.1 shows the level set function for a circle, which is generated by equation (3.4) with a first-order numerical scheme. At first sight it looks rather smooth and accurate. On second thoughts the results of the proceeding chapter will show that this solution is not regular enough.

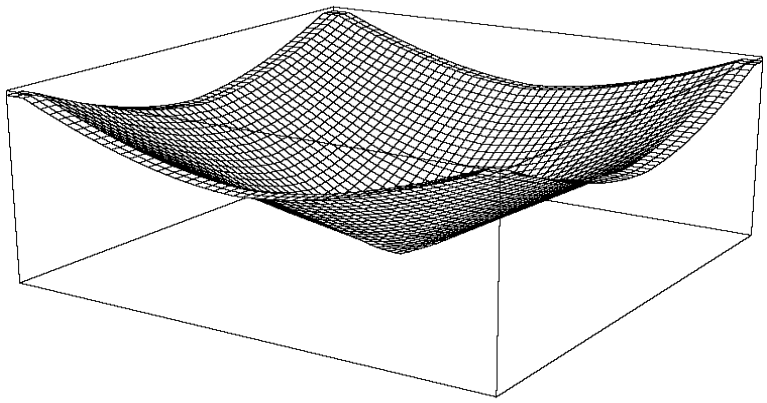


Figure 8.1: Level set function after reinitialization by equation (3.4).

8.1 Analysis of Area Loss

The basic equation used for reinitialization is equation (3.4):

$$\begin{aligned} \Phi_t &= \text{sign}(\Phi_0)(1 - |\nabla\Phi|) \\ \Phi(x, y, 0) &= \Phi_0(x, y) \end{aligned}$$

Therefore, numerical experiments are done by an application to the initial value problem above. Later on, when different methods that are designed to avoid area loss will be compared, equation (3.4) will be referred to as 'conventional' equation for reinitialization in order to avoid misapprehensions.

The main observations done in chapter 6 will be confirmed in this section. To start with, the influence of the bends of the interface on area loss is considered. Therefore, the development of an interface with one single bend is shown in figure 8.2. The movement into the 'convex part' of the domain, which has already been mentioned in section 6.3 at p. 37, is clearly visible. This movement may also be considered as smoothing out of the bend. Moreover, it is obvious that the deviation from the initial interface is introduced at the bend and propagates further. This shows that area loss depends on the bends of the interface not on its straight parts.

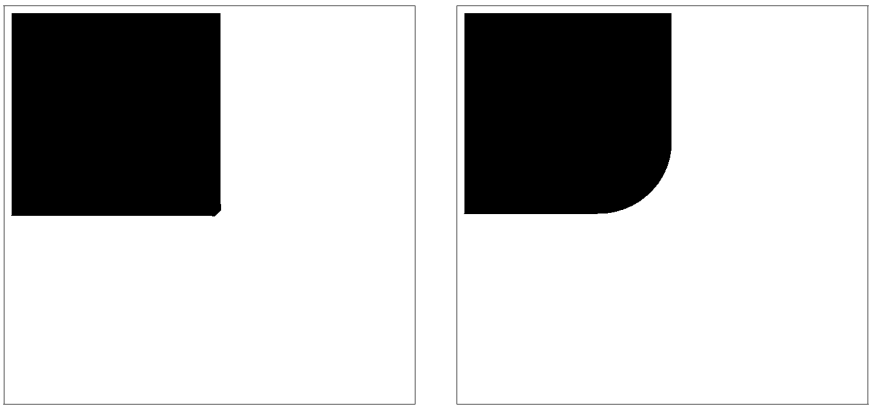


Figure 8.2: Influence of a single bend on the interface during reinitialization. (a) LEFT: Area enclosed by the zero level set before reinitialization. (b) RIGHT: Area after reinitialization.

The next point to be considered is a quantitative aspect of area loss. Recall that *no* error is introduced in the approximation of the gradient of the level set function along the lines $x = 0$ and $y = 0$ (see section 6.4, p. 41 and section 6.5, p. 44). Area loss is clearly getting worse 'between' the axis of the computational grid while it is almost not visible along the axis (see figure 8.3).

Conclusively, the curvature of the zero level set increases at locations where it does not move while it decreases elsewhere. This may explain the 'oscillations' in the curvature, which are observed using equation (3.4) for reinitialization (see figure 8.5). In order to emphasize this effect, no volume conserving method (reset procedure) is applied. Moreover, the first-order scheme is used because the effect would hardly be visible otherwise.

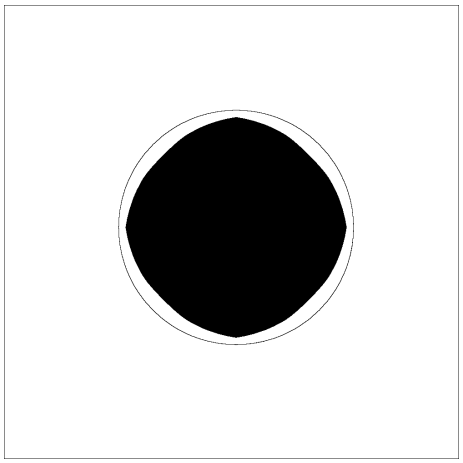


Figure 8.3: Local differences in area conservation. Contour plot of the zero level set before reinitialization (outer circle) and contour plot of the level set function's threshold after reinitialization (black area).

8.2 Different Numerical Schemes

The basic tests for a comparison of different schemes and methods are performed on a 50×50 grid, which discretizes a physical domain $(0, 20) \times (0, 20)$. Hence the grid width is chosen to be $\Delta x := 0.4$. Although the CFL condition (see section 3.4) allows larger time steps, $\Delta t := 0.04$ is used. This is done because the influence of area conserving methods, in particular the reset procedure, shall be kept low as long as only different numerical schemes are compared. Later $\lambda \equiv \Delta t/\Delta x := 0.5$ will be used because it reduces the number of time steps, which is very important in practical applications due to computational expense. The level set function is initialized ($\Phi_0(x, y)$) with the double distance from the zero level set, which is defined to be the circle around $(10, 10)$ of radius 5.1. 300 iterations of equation (3.4) are applied for reinitialization. As already indicated above, reset procedure is applied by default unless stated otherwise.

In order to find the most convenient *numerical scheme*, which will be used for further tests, the results obtained by different schemes are compared with respect to area conservation and smoothness of curvature.

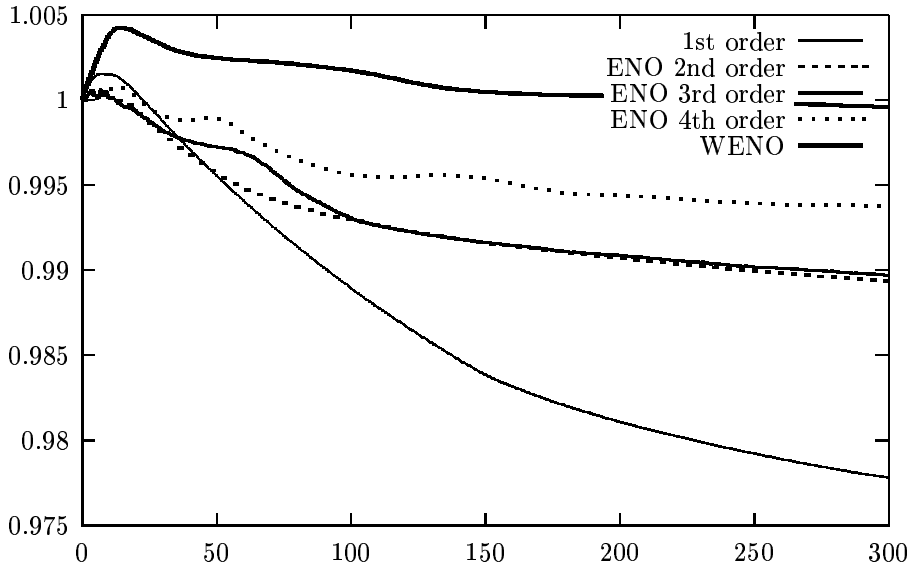


Figure 8.4: Evolution of area enclosed by the zero level set during reinitialization. Comparison of numerical schemes of different order of accuracy.

In figure 8.4 the area of the fluid bubble, which is enclosed by the zero level set, is plotted over the number of iterations. The area of the fluid bubble is given in fractions of the initial data's area. In general, ENO schemes are working better with increasing order of accuracy as far as area conservation and accuracy is concerned (see figures 8.4 and 8.5).
Since WENO schemes are of higher order, they work even better in smooth

regions. Moreover, area gain at the beginning of the iteration is well visible for the first-order scheme as well as for the WENO scheme. The oscillations that occur within the first ten iterations of the other schemes are introduced by the reset procedure. An improvement with increasing order of the numerical scheme is clearly observed.

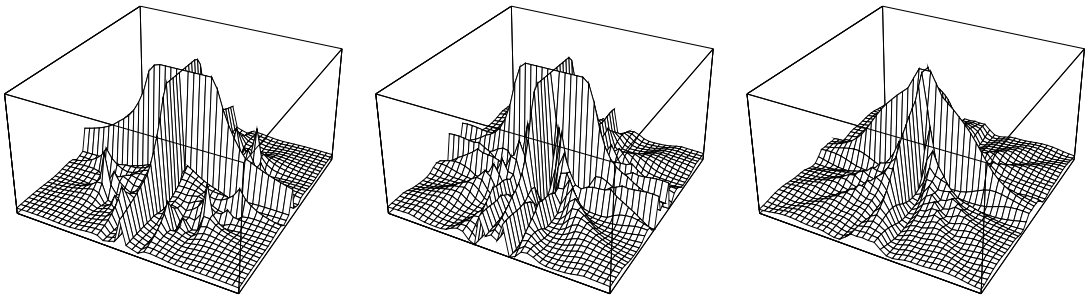


Figure 8.5: Curvature of the level sets after reinitialization. LEFT: Reinitialization by the first-order scheme. CENTRE: ENO scheme of order three. RIGHT: WENO scheme.

The results observed above concerning area loss coincide with the results concerning computation of curvature. An improvement in smoothness is observed with increasing order of the numerical method. In figure 8.5 the curvature of the level sets computed from the data obtained by ENO schemes of first- and third-order as well as by the WENO scheme is shown. Even if the WENO scheme is the most accurate one and the other schemes create even worse results, the curvature looks not very satisfactory, although the level set data itself shown in figure 8.1 looks rather smooth. Therefore, it would be desirable to find methods that smooth the level set function so that a significant improvement is achieved as far as smoothness of curvature is concerned.
Note that the main effect,which is a significantly higher curvature along the axes crossing the centre of the initial circle, coincides very well with the observations in figure 8.3, which shows the quantitative distribution of area loss. Moreover, this main effect is qualitatively equivalent in all pictures of figure 8.5.

The main qualitative difference between ENO and WENO schemes can be seen by considering the computation of curvature based on the smoothed level set function that has been disturbed initially[1]. Corresponding to [11], from figure 8.6 can be recognized that reinitialization smoothes the level set function so that the resulting curvature is much improved. But there is a distinguishable difference between ENO and WENO schemes, which Peng already expressed in [11] as follows:

[1]The perturbation of the initial level set function is done as described in [11]: $\Phi_0(x,y) := d + \epsilon/(16\pi)\sin(4\pi d/\pi \sin 5\theta)$ if $|d| \leq \epsilon$ and $\Phi_0(x,y) := d$ otherwise, where d denotes the distance from the zero level set, $\theta = tan^{-1}(y/x)$, and $\epsilon = 10\Delta x$.

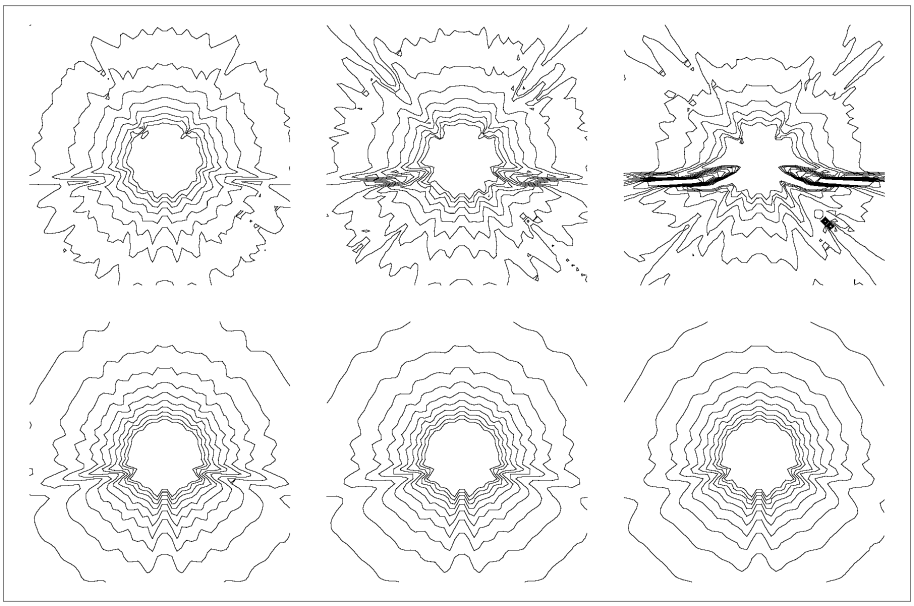


Figure 8.6: Contour plot of the level set's curvature. (a) TOP LEFT: ENO scheme (of order three) after 64 iterations. (b) TOP CENTRE: ENO scheme after 128 iterations. (c) TOP RIGHT: ENO scheme after 256 iterations. (d) BOTTOM LEFT: WENO scheme after 64 iterations. (e) BOTTOM CENTRE: WENO scheme after 128 iterations. (f) BOTTOM RIGHT: WENO scheme after 256 iterations.

> *The curvature computed from the reinitialized data of* Φ *by the WENO scheme is less noisy than that by the ENO scheme. Moreover, the WENO scheme does not deteriorate the level curves* [of curvature]*, which happens to the ENO scheme, as we continue the iteration further. We believe this is due to the fact that WENO smoothly weights the candidates stencil in contrast to the ENO scheme, which jumps from one stencil to another abruptly even in the smooth part of the solution.*

All together, it seems convenient to use the WENO scheme, which is up to fifth-order accurate in smooth regions, together with a TVD Runge-Kutta time discretization of order three for the further computation. In addition to the numerical results above, this choice is justified by the mathematical analysis of area loss in section 6.5 where central schemes turned out to be more convenient for reinitialization.

8.3 Methods Preserving Area Conservation

From figure 8.4 it follows that a considerable amount of area vanishes with respect to the area of the initial data Φ_0 even if the WENO scheme is used. Therefore, it is desirable to find out how the methods designed to avoid area loss (see chapter 7) improve this result.

Reset Procedure

In order to emphasize the effect of the reset procedure, which forces the grid points to keep their sign, the following comparison is done by the *first*-order scheme.
From figure 8.7 can be seen that area loss will appear to be fairly linear with respect to the number of iterations if the reset procedure is not applied. Obviously, area loss is limited to a maximum amount by the reset procedure. This is true because the interface may move closer to adjacent grid points but it cannot move further since the grid points are forced to keep their sign. But this is only a quite rough method and may introduce disturbances in the data around the zero level set, which will be reflected in the computation of the interface's curvature as shown in the first picture of figure 8.5.

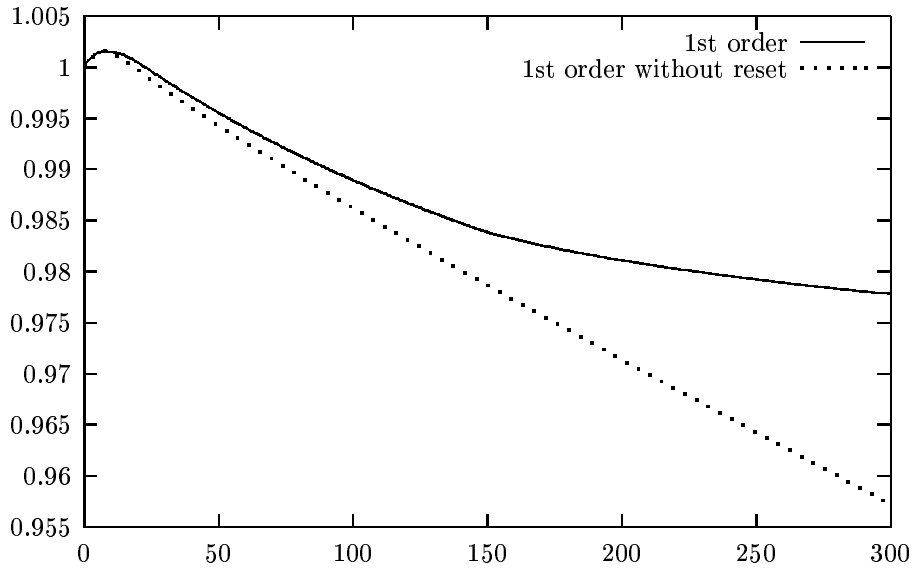


Figure 8.7: Area loss conventional reinitialization using equation (3.4) with and without reset procedure for the first-order scheme.

These additional oscillations in curvature around the zero level set that are visible compared to the other schemes are probably introduced by reset procedure, which applies almost after each iteration step using the first-order scheme.

Since the WENO scheme turned out to be the most convenient scheme for reinitialization, the following methods designed to avoid area loss are compared with the results produced by the WENO scheme (including reset procedure). Moreover, the grid ratio $\lambda \equiv \Delta t/\Delta x$ is put to 0.5, which implies a time step of $\Delta t = 0.2$, in order to reduce the number of iterations. Conclusively, the process terminates after 100 iterations.

Volume Conserving PDE Approach

Applying equation (7.1) after conventional reinitialization (i.e. 80 iterations using equation (3.4) and 20 additional iterations using equation (7.1)) corrects the level set function in order to avoid any area loss.

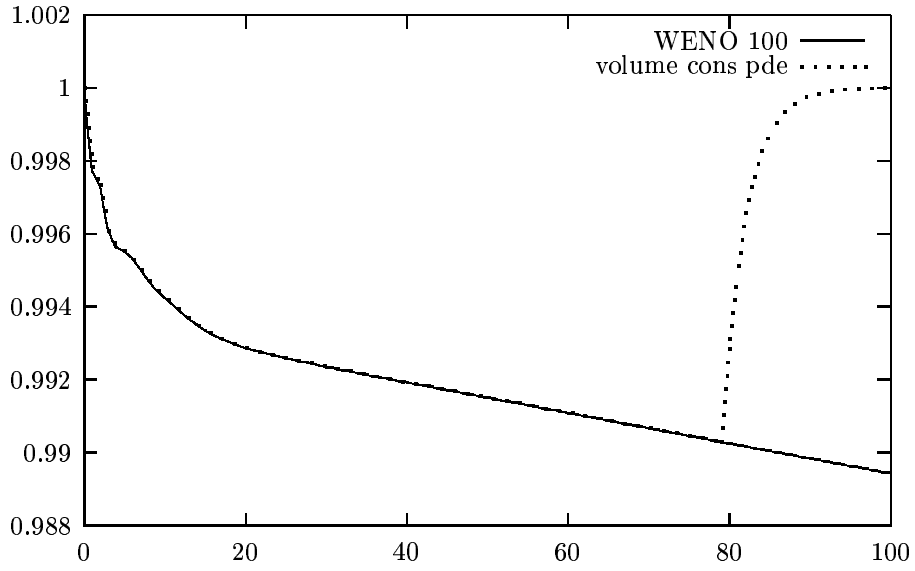


Figure 8.8: Area loss by reinitialization using equation (3.4) with and without applying equation (7.1).

However, one has to take into account that this approach causes flat gradients in the level set function as shown in figure 8.9. This plot shows the absolute value of the gradient field over the number of iterations. In contrast to the basic tests, the initial condition is chosen to be the theoretical distance function (i.e. $|\nabla \Phi_0| = 1$). Firstly, equation (3.4) is applied for 200 iterations. Secondly, 45 iterations of equation (7.1) are additionally applied in order to emphasize the evolution of flat gradients by equation (7.1).
Therefore, a 'mixture' of equation (7.1) and (3.4) would be necessary to preserve area conservation without getting flat gradients. Another drawback of this approach is the fact that the stability constant P is very sensitive to changes in the discretization, e.g. it does depend on the mesh width Δx. Moreover, it is very

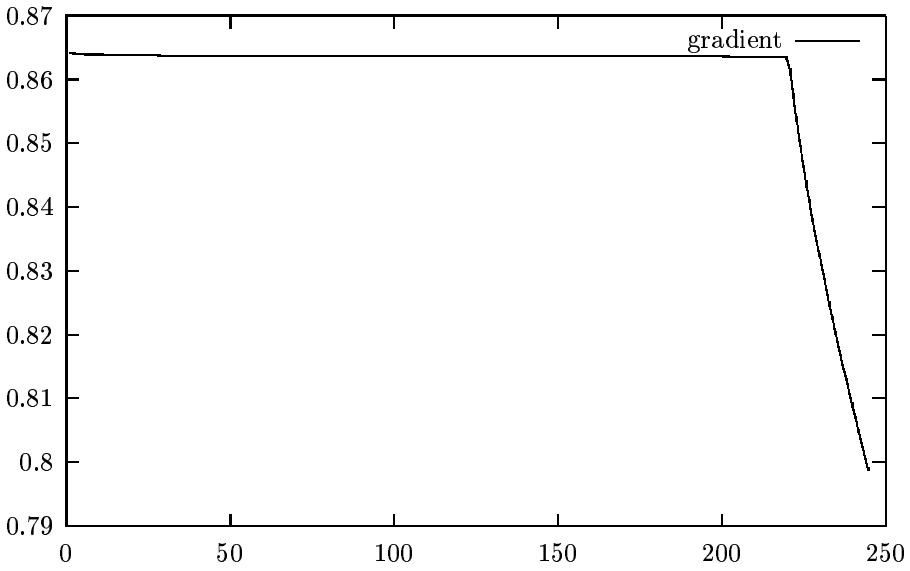


Figure 8.9: Average gradient of the distance function produced by 200 iterations of equation (3.4) and 45 additional iterations of equation (7.1).

difficult to determine this constant for complex geometries. In addition to that, area loss is compensated globally, i.e. it is not at all clear that the interface is corrected at those places where area has been lost before. This is the main reason that it seems *not* convenient to use equation (7.1) for reinitialization.

Constrained Distance Function Approach

The next method mentioned in chapter 7 is the constraint, which applies along the zero level set during the iteration of the conventional equation (3.4).

$$\Phi_t = \text{sign}(\Phi_0)(1 - |\nabla\Phi|) + \lambda f(\Phi)$$

The result looks rather encouraging. Figure 8.10 shows that using the constrained equation (7.2) for reinitialization does conserve area better than without the modification. A clear improvement is obtained compared to the conventional reinitialization using equation (3.4). Moreover, the constraint works locally, i.e. the correction applies at those cells where area has been lost before. This is a big advantage compared to the volume conserving PDE approach considered above. However, some area loss still remains.

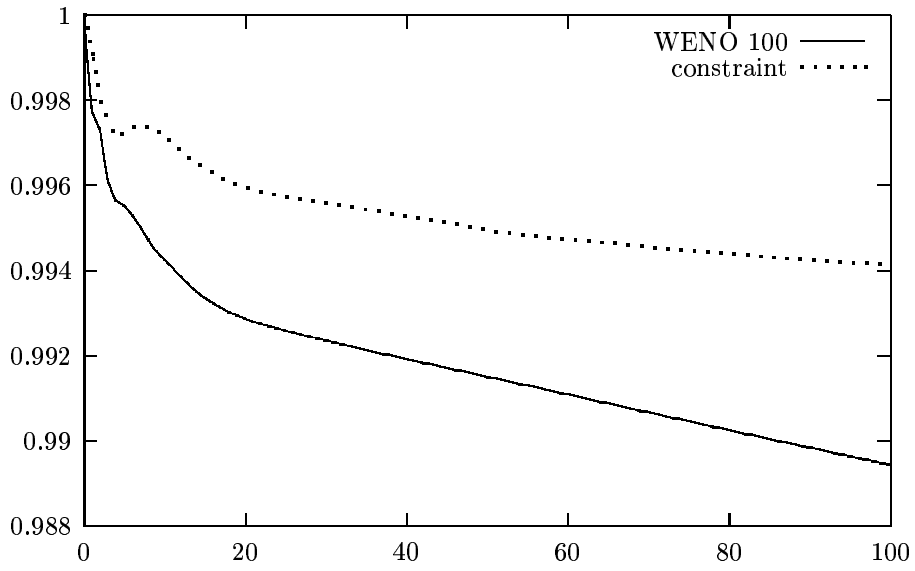


Figure 8.10: Area loss by conventional reinitialization using equation (3.4) and by the constrained equation (7.2).

Extrapolation Method for the Interface

Finally, the extrapolation method for the interface even improves the result obtained by the constraint. From figure 8.11 it is obvious that area is almost totally conserved. At the beginning of the iteration some oscillations occur that are introduced by the fact that area cannot be conserved in general (see section 7.1). But since the corrections become smaller and smaller, the oscillations decrease as well.

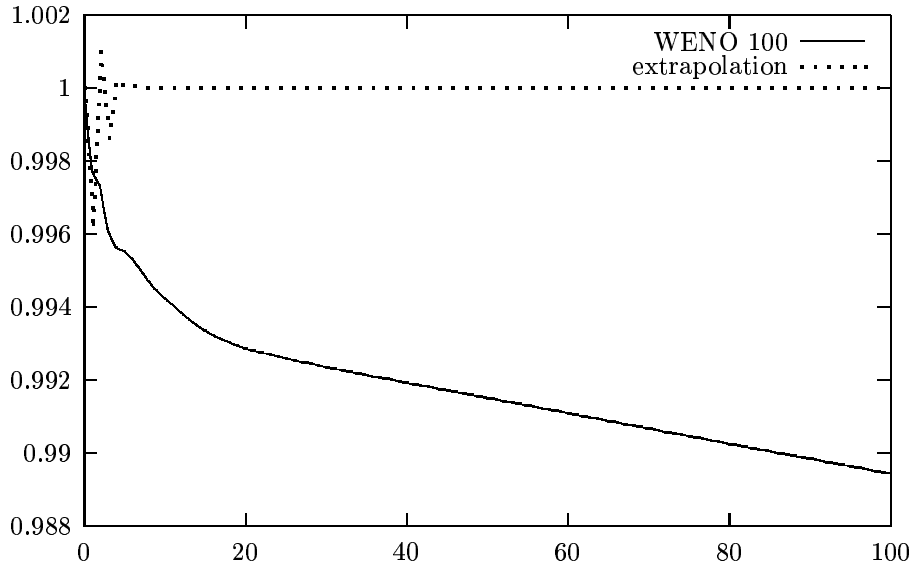


Figure 8.11: Area loss by conventional reinitialization using equation (3.4) and the new extrapolation method for the interface.

Only a very small error remains, which cannot be reduced further (see figure 8.12). Therefore, small perturbations are introduced because the role of exterior and interior grid points changes all the time. Hence, the fluid cells that are respectively conserved or not change as well. This leads to a permanent correction of the interface's position, which prevents that a steady-state solution can be obtained. Stopping this 'changing process' when the oscillations are sufficiently small (e.g. after 60 iterations) will avoid such a permanent perturbation so that a steady-state solution can be obtained.
Note that in figure 8.12 the same data file is plotted as in figure 8.11. Only the plot range in 'y'-direction (i.e. the area fraction) is much smaller in the former plot. This simply means that figure 8.12 is a zoom of figure 8.11. The plot range of the area fraction in figure 8.12 goes from 0.999998 to 1.000002.

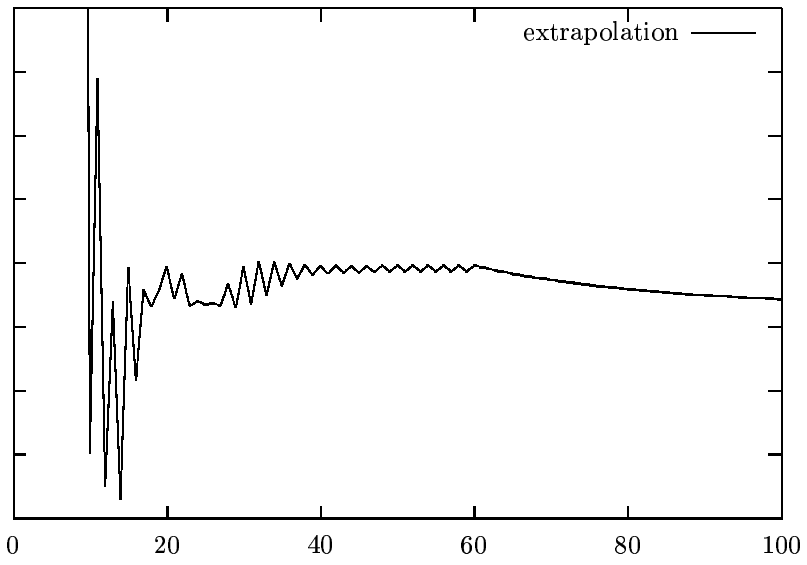


Figure 8.12: Remaining error in the area enclosed by the zero level set using the extrapolation method for the interface. The plot range goes from 0.999998 to 1.000002.

8.4 Curvature of Area Conserving Methods

While the effect of methods designed to avoid area loss on area conservation has been considered above, their effect on smoothness of the obtained solution is taken into account in this section. As already mentioned earlier, the smoothness of the solutions are visualized by the computed curvature because there are no distinguishable differences visible in the level set function itself. Moreover, the curvature is used in the level set formulation of the Navier-Stokes equations (3.2), which implies the need of sufficiently smooth curvature in order to get reasonable results.

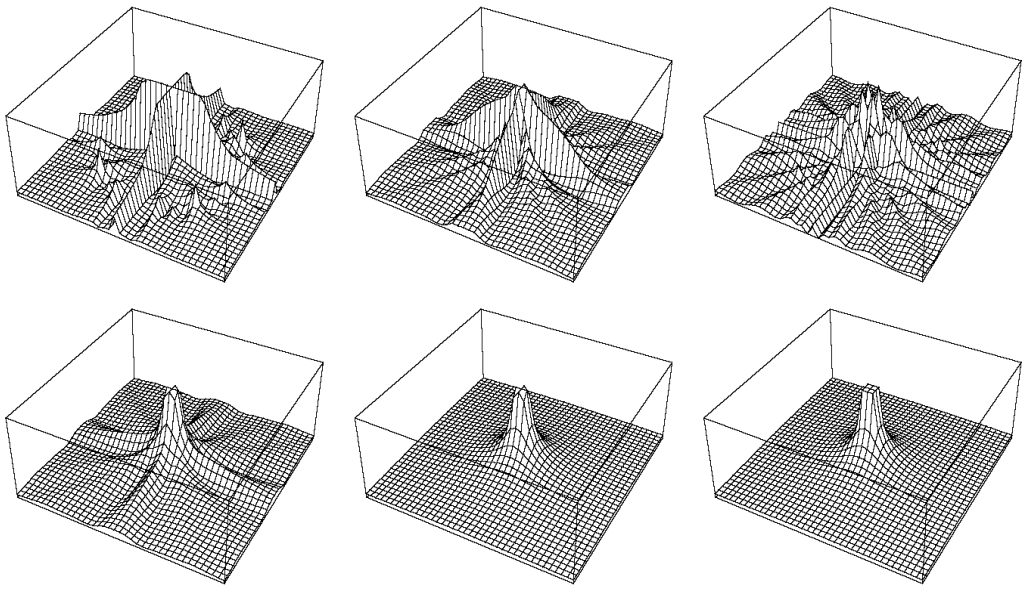


Figure 8.13: Curvature of the level sets after reinitialization. (a) Top Left: First order scheme applied to the conventional equation (3.4). (b) Top Centre: WENO scheme applied to the conventional equation (3.4). (c) Top Right: WENO scheme applied to the constrained equation (7.2). (d) Bottom Left: Additional smoothing of (b) by equation (7.1). (e) Bottom Centre: Extrapolation method for the interface. (f) Bottom Right: Curvature of the level sets computed from the theoretical distance function.

Distance Function Approach

Using the conventional equation (3.4) to generate the distance function creates a rather smooth level set function Φ (see figure 8.1). But as far as computation of curvature is concerned it is not smooth enough, i.e. that the computed curvature κ is still too rough (figure 8.13 (a), (b) and 8.5). The main effect, which can be seen in figure 8.5, is that curvature is significantly higher along the axis of

the co-ordinate system ($x = 0$ and $y = 0$). As already mentioned above, this observation coincides with the observations in section 6.5, which was that no area loss occurs along $x = 0$ and $y = 0$ and which leads to high curvature along these axes. A numerical confirmation of this result is given by figure 8.3.

Volume Conserving PDE Approach

As can be seen from figure 8.13, the volume conserving PDE approach (d) does not only reduce area loss totally but it even improves smoothness of curvature very much. Even if area is totally conserved, this observation can be used to improve further smoothness of curvature by manually reducing area. Therefore, equation (7.1) can be applied as long as necessary (to obtain smooth curvature). This approach is used in figure 8.14 to improve curvature obtained by ENO and WENO schemes from initially disturbed level set function as described in [11] and already mentioned above. The improvement is well visible in figure 8.14. The result of figure 8.14 is obtained by additional iterations of the results shown in figure 8.6 (for 64 iterations) using equation (7.1).
However, the disadvantages of this approach mentioned above still apply.

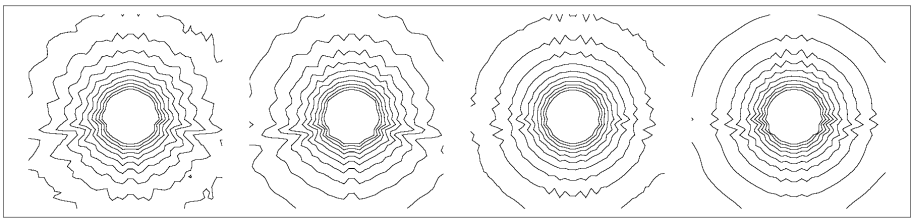


Figure 8.14: Curvature of the level sets (see figure 8.6) smoothed additionally by equation (7.1). From left to right: (a) First Picture: ENO scheme after 64 iterations of equation (3.4). (b) Second Picture: WENO scheme after 64 iterations of equation (3.4). (c) Third Picture: (a) with additional smoothing by equation (7.1). (d) Last Picture: (b) with additional smoothing by equation (7.1).

Constrained Distance Function Approach

Unfortunately, the constraint that is applied at the interface leads to worse results as far as accuracy of the interface's curvature is concerned. The result is shown in figure 8.13 (c) and is rather disappointing. Obviously, the perturbations in curvature come from the corrections of the interface introduced by the constraint. Therefore, the constrained distance function approach seems not to be a very convenient method as long as curvature-dependent problems are under consideration.

Extrapolation Method for the Interface

In addition to the good results as far as area conservation is concerned, it turns out that the level set function will be further smoothed so that the computed curvature is significantly improved as well. Figure 8.13 (e) shows this improvement. No significant difference is visible any more between curvature obtained by the extrapolation method and curvature obtained by the theoretical distance function (f).
Unfortunately, the extrapolation method seems to falsify curvature slightly as illustrated in figure 8.15 where the curvature of an ellipse is considered.

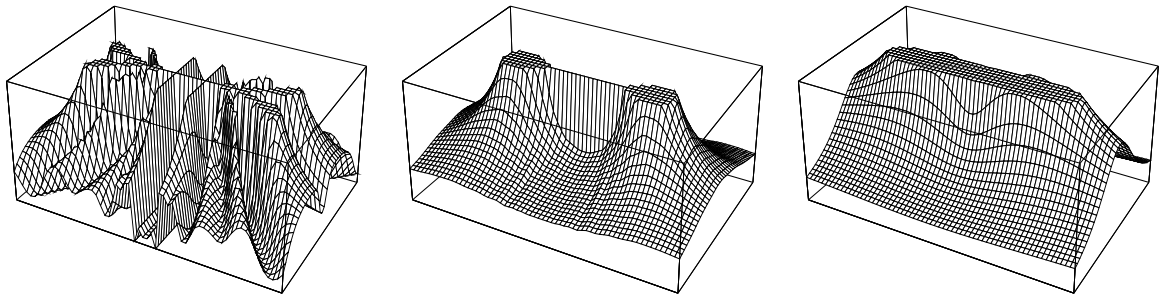


Figure 8.15: The level set's curvature of an ellipse. From left to right: (a) LEFT: After conventional reinitialization by equation (3.4). (b) CENTRE: After reinitialization by the extrapolation method for the interface. (c) RIGHT: Curvature computed before reinitialization.

However, the results appear to be still useful because the oscillations that are observed using the conventional iteration by purely applying equation (3.4) (a) are almost totally reduced. This is important because such oscillations affects curvature-dependent computations.

Remark. The computations have been performed on a PC under Linux Red Hat 4.2. The code has been implemented mainly by myself in ANSI C using the GNU C compiler (gcc). Only the implementation of the basic WENO interpolation procedure has been given to me by my supervisor Dr. Zemitis, who works at the 'Institut für Techno- und Wirtschaftmathematik (ITWM)' in Kaiserslautern. The numerical results have been visualized using the commercial software package Mathematica® for Students, version 3.0 by WOLFRAM Research and GNUPLOT by the Free Software Foundation under GNU General Public License.

Chapter 9

Summary and Conclusion

In this thesis an introduction to the level set method has been given, which has been motivated by a discussion on different approaches for solving problems with moving boundaries. Namely the Lagrangian formulation, the volume of fluid method, and the level set method (both based on an Eulerian formulation) have been compared. While the volume of fluid technique is difficult to apply on curvature-dependent problems, the Lagrangian formulation has been used successfully on various applications. However, using the level set method instead avoids serious drawbacks of the Lagrangian formulation. In particular, no explicit description of the interface is necessary, complex interface geometries are handled naturally, fixed-grid finite difference approximations may be used, and an extension to three space-dimensions can be done rather easily (see chapter 2).

The arising problems of the level set method, which lead to the need of reinitialization of the level set function, have been discussed. Based on a discussion of the desired properties of the level set function, different approaches for reinitialization have been proposed and the most convenient one has been singled out (see chapter 3).

The main problem of reinitialization, which is area loss, has been analysed in detail. The result of this analysis is that a systematic error is introduced when the partial space derivatives are approximated. The systematic error is caused by the upwind method, which chooses either forward *or* backward approximations according to the characteristics of the equation (see section 6.4). Consequently, the absolute value of the gradient is overestimated inside and underestimated outside the fluid bubble, which introduces a movement of the interface towards its interior region. In contrast to the basic ENO approximation, the WENO scheme avoids this error partly because it may be considered as a 'central' scheme in smooth regions. Therefore, one may explain the observation that WENO schemes produce less area loss than the corresponding ENO scheme. Furthermore, the systematic error is independent of the level set function's initialization, i.e. either $\Phi < 0$ inside and $\Phi > 0$ outside the fluid bubble or vice versa (see section 6.2).

Above all, the analysis of the systematic error gives an explanation for area *loss*, which was mentioned in almost all previous publications about the level set method. In addition to that general result, an explanation for area gain has been given, which is sometimes observed at the beginning of the iteration (see section 6.6). Moreover, a rough qualitative distribution of area loss on the computational domain has been presented, which may explain the main effect of the oscillations in curvature observed in numerical computations (see section 6.4 and 8.1).

Several numerical methods for solving the resulting partial differential equations have been compared. In fact, the WENO scheme gives the best results as far as area conservation and computation of curvature is concerned (see section 8.2), which coincides with the analysis of the systematic error (see section 6.2).

Since the most serious problem concerning the level set method seems to be area loss during reinitialization, several different approaches to solve this problem were proposed in literature. A comparison among those has been performed in this thesis. Primarily, their effect on area loss has been considered (see section 8.3), but in addition to that the influence on the level set's local curvature has also been taken into account (see section 8.4). In particular, the reset procedure has turned out not to be accurate enough and to introduce additional numerical 'noise', which affects computation of curvature. The volume conserving PDE approach does not only correct area loss but it smoothes very much the computed curvature. However, it is difficult to determine its stability constant P, and what is even more serious, it is not clear that the position of the interface is corrected at those places where area has been vanished during the computation. The constrained distance function approach further improves area loss, but it also introduces numerical 'noise' to the computed curvature.

Motivated by the analysis of area loss a modification of the conventional reinitialization procedure has been presented and implemented (see section 7.4). It is designed to avoid any area loss during reinitialization and the results are indeed rather good. Moreover, a significant improvement concerning the computation of curvature is achieved. Although there is still a deviation from the exact curvature visible, the oscillations that occur typically using the conventional reinitialization procedure are almost totally reduced.

In conclusion, I recommend the use of the extrapolation method using high-order WENO schemes for an approximation of the space derivatives and high-order TVD Runge-Kutta methods for the approximation of the temporal derivative. Thus, area loss during reinitialization can be avoided and smooth curvature can be computed from the level set data.

Appendix A

Names and Symbols

Γ:	Interface separating two fluids.
δ:	Dirac delta function.
$\Delta t, \Delta x, \Delta y$:	Mesh width in t-,x- and y-direction, respectively.
κ:	Curvature of the level sets.
λ:	Mesh ratio or grid ratio $\lambda \equiv \frac{\Delta t}{\Delta x}$.
μ:	Dynamic viscosity function $\mu : \ \Omega \to \mathbb{R}$.
μ_1, μ_2:	Dynamic viscosity of the outer and the interior fluid (fluid bubble), respectively.
ρ:	Density function $\rho : \ \Omega \to \mathbb{R}$.
ρ_1, ρ_2:	Density of the outer and interior fluid, respectively.
σ:	Surface tension.
τ:	Stress tensor.
ϕ:	Undivided differences for the ENO polynomial construction.
Φ:	Level set function.
Φ':	First derivative of Φ (assuming $\Phi : \ \mathbb{R} \to \mathbb{R}$).
Φ_t, Φ_x, Φ_y:	Partial derivatives of Φ with respect to t, x and y, respectively.
Φ_x^+, Φ_x^-:	Forward and backward approximations of the partial derivative (with respect to x), respectively.
Ω:	Computational domain.
Ω_1, Ω_2:	Outer part and interior part of the computational domain, respectively.
∇:	Nabla operator.
Δ:	Laplace operator.
$\lvert \ . \ \rvert$:	Euclidean norm of a vector.
$x \cdot y$:	Scalar product of the vectors x and y.
c_{ij}:	Fluid cell at the grid point x_{ij}.
Du:	Jacobi matrix of the vector field u.
$(Du)^T$:	Transposed matrix of the Jacobi matrix of the vector field u.

$\mathcal{D}$**:** Deformation tensor.
F**:** Speed function for the interface in outer normal direction.
g**:** Gravitational constant.
$i(j)$**:** Leftmost point in the stencil for the interpolating polynomial at x_j.
k**:** Surface tension coefficient.
n**:** Outer unit normal vector of the level sets.
p**:** Pressure of the fluid.
p_{ij}^1, p_{ij}^2**:** Position of the interface at the edges of the fluid cell c_{ij}, i.e. zeros of the level set function at the edges of the fluid cell.
RHS: Right-hand side of an equation.
u**:** Velocity field of the fluid.
t, x and y**:** Time-/ and space variables in x-/ and y-direction, respectively.
$\dot{x}$**:** Partial derivative of x (trajectory) with respect to time t.
x_{ij}**:** Grid point at $(x,y)=(x_0 + i\Delta x, y_0 + j\Delta y)$ where (x_0,y_0) is the leftmost lowest corner of the (rectangular) computational domain.

Bibliography

[1] Adalsteinsson, Sethian: *A fast level set method for propagating interfaces,* Journal of Computational Physics 118, 269-277 (1995)

[2] Brackbill, Kothe, Zemach: *A continuum method for modeling surface tension,* Journal of Computational Physics 100, 335-354 (1992)

[3] Chang, Hou, Merriman, Osher: *A level set formulation of Eulerian interface capturing methods for incompressible fluid flows,* Journal of Computational Physics 124, 449-464 (1996)

[4] Crandall, Evans, Lions: *Some properties of viscosity solutions of Hamilton-Jacobi equations,* Transactions of the American Mathematical Society, 282, (2), 487-502 (1984)

[5] Crandall, Lions: *Viscosity solutions of Hamilton-Jacobi equations,* Transactions of the American Mathematical Society, 277, (1), 1-42 (1983)

[6] Crandall, Lions: *Two approximations of solutions of Hamilton-Jacobi equations,* Mathematics of Computation, 43, 167, 1-19 (1984)

[7] Eriksson, Estep, Hansbo, Johnson: *Computational differential equations,* Cambridge University Press, 1996

[8] Griebel, Dornseifer, Neunhoeffer: *Numerische Simulation in der Strömungslehre: Eine praxisorientierte Einführung,* Braunschweig; Wiesbaden: Vieweg, 1995

[9] Hou, Li, Osher, Zhao: *A hybrid method for moving interface problems with application to the Hele-Shaw flow,* Journal of Computational Physics 134, 236-252 (1997)

[10] Isenberg: *The science of soap films and soap bubbles,* Tieto Ltd., Clevedon, Avon, Endland 1978

[11] Jiang, Peng: *Weighted ENO schemes for Hamilton-Jacobi equations,* Preprint (1996)

[12] KANG: *A level set approach for the motion of soap bubbles with curvature dependent velocity or acceleration,* PhD thesis (unpublished) (1996)

[13] KREYSZIG: *Differentialgeometrie,* Akademische Verlagsgesellschaft Geest & Portig KG, Leipzig 1968

[14] LEVEQUE: *Numerical Methods for Conservation Laws,* / Randall J. LeVeque. - Basel; Boston; Berlin: Birkhäuser, 1992

[15] MERRIMAN, BENCE, OSHER: *Motion of multiple junctions: A level set approach,* Journal of Computational Physics 112, 334-363 (1994)

[16] MULDER, OSHER, SETHIAN: *Computing interface motion in compressible gas dynamics,* Journal of Computational Physics 100, 209-228 (1992)

[17] OSHER, SETHIAN: *Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations,* Journal of Computational Physics 79, 12-49 (1988)

[18] OSHER, SHU: *High-order essentially nonoscillatory schemes for Hamilton-Jacobi equations,* SIAM Journal on Numerical Analysis, 28, (4), 907-922, (1991)

[19] ROUY, TOURIN: *A viscosity solutions approach to shape-from-shading,* SIAM Journal on Numerical Analysis 29, (3), 867-884 (1992)

[20] SETHIAN: *Numerical algorithms for propagating interfaces: Hamilton-Jacobi equations and conservation laws,* Journal of Differential Geometry, 31, 131-161 (1990)

[21] SETHIAN: *Level set methods: Evolving interfaces in geomertiy, fluid mechanics, computer vision, and material science,* Cambridge University Press 1996

[22] SUSSMAN, FATEMI: *An efficient, interface preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow,* Preprint 1996

[23] SUSSMAN, SMEREKA, OSHER: *A level set approach for computing solutions to incompressible two-phase flow,* Journal of Computational Physics 114, 146-159 (1994)

[24] UNVERDI, TRYGGVASON: *A front-tracking method for viscous, incompressible, multi-fluid flow,* Journal of Computational Physics 100, 25-37 (1992)

[25] WEICKERT: *Anisotropic diffusion in image processing,* B.G. Teubner Stuttgart (1998)

Acknowledgements

First and foremost, I would like to thank Professor Helmut Neunzert for initiating this interesting subject and drawing my attention to the essential aspects of the topic. Moreover, I want to thank Dr. Aivars Zemitis, who has been a very inspiring supervisor for valuable hints during all periods of this thesis. In addition to that I thank Mathias Moog for joining our discussions and giving helpful advises on the manuscript, which is written using LaTeX. Eventually, I thank many fellow students for fruitful discussions.
Apart from the mathematical aspect I am grateful to Catherine Beck and Michaela Cetto for having a closer look on the English language. Furthermore, I am thankful to the 'UNIX-AG' at the University of Kaiserslautern for much assistance during the installation and administration of the Linux Red Hat 4.2 distribution on my PC.

Finally, I would like to express my deep gratitude to my parents for giving me much more than financial support.

Affirmation

Hereby I assure that I have done this thesis by myself and that I have exclusively used the indicated literature and resources.

Kaiserslautern, 20th June 1998 Rainer Keck