Modified Gradient Search for Level Set Based Image Segmentation

Thord Andersson, Gunnar Läthén, Reiner Lenz and Magnus Borga

Linköping University Post Print

N.B.: When citing this work, cite the original article.

©2013 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

Thord Andersson, Gunnar Läthén, Reiner Lenz and Magnus Borga, Modified Gradient Search for Level Set Based Image Segmentation, 2013, IEEE Transactions on Image Processing, (22), 2, 621-630.

<http://dx.doi.org/10.1109/TIP.2012.2220148>

Postprint available at: Linköping University Electronic Press

<http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-87658>

Modified Gradient Search for Level Set Based Image Segmentation

Thord Andersson, Gunnar L, Reiner Lenz, and Magnus Borga, *Member, IEEE*

Abstract—Level set methods are a popular way to solve the image segmentation problem. The solution contour is found by solving an optimization problem where a cost functional is minimized. Gradient descent methods are often used to solve this optimization problem since they are very easy to implement and applicable to general nonconvex functionals. They are, however, sensitive to local minima and often display slow convergence. Traditionally, cost functionals have been modified to avoid these problems. In this paper, we instead propose using two modified gradient descent methods, one using a momentum term and one based on resilient propagation. These methods are commonly used in the machine learning community. In a series of 2-D/3-D-experiments using real and synthetic data with ground truth, the modifications are shown to reduce the sensitivity for local optima and to increase the convergence rate. The parameter sensitivity is also investigated. The proposed methods are very simple modifications of the basic method, and are directly compatible with any type of level set implementation. Downloadable reference code with examples is available online.

Index Terms—Active contours, gradient methods, image segmentation, level set method, machine learning, optimization, variational problems.

I. INTRODUCTION

ONE POPULAR approach for solving the image segmentation problem is to use the calculus of variations. The objective of the segmentation problem is defined with an energy functional, and the minimizer of this functional represents the resulting segmentation. The functional depends on properties of the image such as gradients, curvatures and intensities, as well as regularization terms, e.g. smoothing constraints. Early variational methods, such as *Snakes* and *Geodesic Active Contours* [1]–[4], often have boundary based terms such as an edge map. A parametrized curve, the active contour, is evolved according to the minimization of the cost functional until it converges to an equilibrium state representing the resulting segmentation. Later variational methods often

include terms which are more region-based, allowing segmentation of objects without distinctive edges. These methods are often based on the Mumford-Shah segmentation model [5] where the image is partitioned into piecewise smooth regions with short boundaries. Chan-Vese used this model together with implicit contours represented by level sets [6]–[9]. This made the optimization problem easier and it naturally handles segmentation topology changes.

In order to solve the optimization problem in level set segmentation methods, the gradient descent method is very often used. It deforms an initial contour in the steepest (gradient) descent of the energy. The equations of motion for the contour, and the corresponding energy gradients, are derived using the Euler-Lagrange equation [10] and the condition that the first variation of the energy functional should vanish at a (local) optimum. Then, the contour is evolved to convergence using these equations. The use of a gradient descent search commonly leads to problems with convergence to local optima and slow convergence in general. The problems are accentuated with noisy data or with a non-stationary imaging process, which may lead to varying contrasts for example. The problems may also be induced by bad initial conditions for certain applications. Traditionally, the energy functionals have been modified to avoid these problems by, for example, adding regularizing terms to handle noise, rather than to analyze the performance of the applied optimization method. This is however discussed in [11], [12], where the metric defining the notion of steepest descent (gradient) has been studied. By changing the metric in the solution space, local optima due to noise are avoided in the search path.

There are many, and much more advanced, alternatives to gradient descent. For nonconvex functionals, there are global optimization techniques such as subdivision schemes with much better performance [13]. Their high complexity, however, makes them applicable for small problems only. Stochastic methods such as the Monte-Carlo (MC) family is another alternative. Simulated annealing implemented with the Metropolis algorithm and local updates has been a popular choice [14], [15]. Unfortunately its convergence to a global solution is logarithmically slow, limiting its usefulness but spurring the development of more advanced MC methods, such as basin hopping [16]. This is used in the context of Mumford-Shah segmentation by Law et al. They propose a hybrid approach combining gradient descent based and stochastic optimization methods to resolve the problem of bad initial guesses, resulting in near global optimal solutions [17]. There are also recently developed methods which use a dual formulation of the Total Variation (TV) norm in order to

Manuscript received September 7, 2011; revised June 27, 2012; accepted September 6, 2012. Date of publication September 21, 2012; date of current version January 10, 2013. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Gang Hua.

T. Andersson and M. Borga are with the Center for Medical Image Science and Visualization, Linkping University, Linkping SE-581 85, Sweden, and also with the Department of Biomedical Engineering, Linkping University, Linkping SE-581 85, Sweden (e-mail: thord.andersson@liu.se; magnus.borga@liu.se).

G. L and R. Lenz are with the Center for Medical Image Science and Visualization, Linkping University, Linkping SE-581 85, Sweden, and also with the Department of Science and Technology, Linkping University, Linkping SE-581 83, Sweden (e-mail: gunnar.lathen@liu.se; reiner.lenz@liu.se).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TIP.2012.2220148

perform fast global minimization of the boundary [18]–[20]. Solem et al. propose using quotient functionals and partial extremal initialization in order to decrease the computation time of the traditional gradient descent method with several orders of magnitude [21]. Graph-based formulations have also been used where a low-energy solution is found by using techniques of combinatorial optimization on a modified Mumford-Shah functional, reformulated on an arbitrary graph [22]. Boykov et al. combined geodesic active contours and graph cuts to find globally minimum geodesic contours [23]. Chang et al. use an active contour model with gradient descent replaced by graph cut global optimization [24], [25].

In spite of these more advanced methods, optimization using gradient descent search is still very common and in active use. This is partly due to its simple implementation, but also to its direct applicability to general non-convex functionals. The focus of this paper is to show that the performance of these standard searches can easily be improved upon by very simple modifications.

The paper contains the following contributions: We present two modified gradient descent methods, one using a momentum term and one based on resilient propagation (Rprop). Our ideas stem from the machine learning community where gradient descent methods often have been used in order to train learning systems, e.g. adapting the weights in artificial neural networks. The proposed methods are very simple, but effective, modifications of the basic method and are directly compatible with any type of level set implementation. The first proposed method is based on a modification which basically adds a momentum to the motion in solution space [26]. This simulates the physical properties of momentum and often allows the search to overstep local optima and take larger steps in favorable directions. The second proposed method is based on *resilient propagation (Rprop)* [27], [28]. In order to avoid the typical problems of gradient descent search, Rprop provides a modification which uses individual (one per parameter) adaptive step sizes and considers only the signs of the gradient components. This modification makes Rprop less sensitive to local optima and avoids the harmful influence of the magnitude of the gradient on the step size. Individual adaptive step sizes also allow for cost functionals with very different behaviors along different dimensions because there is no longer a single step size that has to be appropriate for all dimensions. In this paper, we show how these ideas can be used for image segmentation in a variational framework using level set methods.

These ideas were initially presented in [29] and [30]. Here, the performances of the proposed methods are quantified in a series of 2D/3D-experiments using real and synthetic data with ground truth. The sensitivity to parameter settings is also investigated, providing greater insights into the performance of the methods. In comparison with standard and stochastic gradient descent (SGD), the modifications are shown to reduce the sensitivity for local minima and to increase the convergence rate. Downloadable reference code with examples is available online.

The paper proceeds as follows: In Section II, we describe the ideas of gradient descent with momentum and Rprop in

a general setting and give examples highlighting the benefits. Then, Section III discusses the level set framework and how the proposed methods can be used to solve segmentation problems. The method and the setup of the segmentation experiments in 2D and 3D are presented in Section IV, together with implementation details. The results of these experiments are then presented in Section V. In Section VI we discuss the results and Section VII concludes the paper and presents ideas for future work.

II. GRADIENT DESCENT WITH MOMENTUM AND RPROP

Gradient descent is a very common optimization method which appeal lies in the combination of its generality and simplicity. It can be applied to all cost functions and the intuitive approach of the method makes it easy to implement. The method always moves in the negative direction of the gradient, locally minimizing the cost function. The steps of gradient descent are also easy to calculate since they only involve the first order derivatives of the cost function. Unfortunately, as discussed in Section I, gradient descent exhibits slow convergence and is sensitive to local optima. Other, more advanced, methods have been invented to deal with the weaknesses of gradient descent, e.g. the methods of conjugate gradient, Newton, Quasi-Newton etc, see for instance [31] for a review of continuous numerical optimization methods. Alternatives in the context of Mumford-Shah functionals are discussed in Section I above.

A simple alternative to these, more theoretically sophisticated methods, is often applied in the machine learning community. To improve the convergence rate and the robustness against local optima for the gradient descent search, while avoiding the complexity of more sophisticated optimization algorithms, methods like gradient descent with *Momentum* [26] and *Rprop* [27] were proposed. The starting point of our derivation of the proposed methods is the following description of a standard line search optimization method:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + s_k \quad (1)$$

$$s_k = \alpha_k \hat{\mathbf{p}}_k \quad (2)$$

where $\mathbf{x}_k$ is the current solution, s_k is the next step consisting of length α_k and direction $\hat{\mathbf{p}}_k$. To guarantee convergence, it is often required that $\hat{\mathbf{p}}_k$ is a descent direction while $\alpha_k \geq 0$ gives a sufficient decrease in the cost function. A simple realization of this is gradient descent which moves in the steepest descent direction according to $\hat{\mathbf{p}}_k = -\hat{\nabla} f_k$, where f is the cost function, while α_k satisfies the *Wolfe conditions* [31].

A. Momentum Method

Turning to gradient descent with Momentum, we will adopt some terminology from the machine learning community and choose a search vector according to:

$$s_k = -\eta(1 - \omega)\nabla f_k + \omega s_{k-1} \quad (3)$$

where η is the *learning rate* and $\omega \in [0, 1]$ is the *momentum*. Note that $\omega = 0$ gives standard gradient descent $s_k = -\eta\nabla f_k$, while $\omega = 1$ gives “infinite momentum” $s_k = s_{k-1}$.

Fig. 1. 1-D error function in (3) without noise (right) and with noise (left). The starting points of the gradient searches are also indicated.

The intuition behind this strategy is that the current solution $\mathbf{x}_k$ has a momentum, which prohibits sudden changes in the velocity. This will effectively filter out high frequency changes in the cost function and allow for larger steps in favorable directions. Using appropriate parameters, the rate of convergence is increased while local optima may be overstepped.

The effect of the momentum term is illustrated by two examples shown in Figure 1. The 1D error function used in the examples is:

$$f(r) = 0.25r^2 - \cos(r)^2 \quad (4)$$

where r is a scalar variable. In the first example, Fig. 1 right side, there is no noise. In the second example, Fig. 1 left side, we added low-pass filtered noise. The function has both large and small gradients and, in the second example, two local minima due to noise, see Fig. 1. These properties allow us to demonstrate the characteristics of the different methods. The figure also indicates the starting points for the searches. Fig. 2 shows the remaining error per iteration and the convergence behavior for the different methods. SGD is included for comparison.

In the first noiseless example, Fig. 2(a), the standard gradient descent method with $(\eta = 8)$ does not get close to the global minimum due to its sensitivity to the magnitude of the gradients. In comparison, the Momentum method with $(\eta = 8, \omega = 0.5)$, succeeds in getting close to the minimum. However, since the Momentum method still uses the magnitude of the gradient, it will take smaller steps in the 'plateau' region resulting in slower convergence. For an appropriate choice of momentum ω and learning rate η , e.g. $(\eta = 16, \omega = 0.5)$, the solution approaches the optimum more rapidly. It should be noted however, that too large parameters leads to oscillations.

In the second example with noise, Fig. 2(b), the standard method never reaches the global minimum, even for $(\eta = 16)$. It gets trapped in the first local minimum, see Fig. 1. The first and second Momentum instances with $(\eta = \{16, 8\}, \omega = \{0.1, 0.5\})$ also get trapped in this minimum. The third Momentum instance with $(\eta = 16, \omega = 0.5)$ succeeds in reaching the global minimum. The momentum term effectively low pass filters the gradients which makes the method more robust against small local variations in the gradient field. The second example shows the potential of the Momentum method but also its dependence on appropriate parameter settings.

B. Rprop Method

In standard implementations of steepest descent search, $\alpha_k = \alpha$ is a constant not adapting to the shape of the cost-surface. Therefore if we set it too small, the number of iterations needed to converge to a local optimum may be prohibitive. On the other hand, a too large value of α may lead to oscillations causing the search to fail. The optimal α does not only depend on the problem at hand, but varies along the cost-surface. In shallow regions of the surface a large α may be needed to obtain an acceptable convergence rate, but the same value may lead to disastrous oscillations in neighboring regions with larger gradients or in the presence of noise. In regions with very different behaviors along different dimensions it may be hard to find an α that gives acceptable convergence performance.

The Resilient Propagation (Rprop) algorithm [27] was developed to overcome these inherent disadvantages of standard gradient descent using adaptive step-sizes Δ_k called *update-values*. There is one update-value per dimension in $\mathbf{x}$, i.e. $\dim(\Delta_k) = \dim(\mathbf{x}_k)$. However, the defining feature of Rprop is that the size of the gradient is never used. Only the signs of the partial derivatives are considered in the update rule. Another advantage of Rprop, very important in practical use, is the robustness of its parameters; Rprop will work out-of-the-box in many applications using only the standard values of its parameters [32].

We will now describe the Rprop algorithm briefly, but for implementation details of Rprop we refer to [28]. For Rprop, we choose a search vector $\mathbf{s}_k$ according to:

$$\mathbf{s}_k = -\text{sign}(\nabla f_k) * \Delta_k \quad (5)$$

where Δ_k is a vector containing the current update-values, a.k.a. *learning rates*, $*$ denotes elementwise multiplication and $\text{sign}(\cdot)$ the elementwise sign function. The individual update-value Δ_k^i for dimension i is calculated according to the rule:

$$\Delta_k^i = \begin{cases} \min(\Delta_{k-1}^i \cdot \eta^+, \Delta_{\max}), & \nabla^i f_k \cdot \nabla^i f_{k-1} > 0 \\ \max(\Delta_{k-1}^i \cdot \eta^-, \Delta_{\min}), & \nabla^i f_k \cdot \nabla^i f_{k-1} < 0 \\ \Delta_{k-1}^i, & \nabla^i f_k \cdot \nabla^i f_{k-1} = 0 \end{cases} \quad (6)$$

where $\nabla^i f_k$ denotes the partial derivative i in the gradient. Note that this is Rprop without backtracking as described in [28]. The update rule will accelerate the update-value with a factor η^+ when consecutive partial derivatives have the same sign, decelerate with the factor η^- if not. This will allow for larger steps in favorable directions, causing the rate of convergence to be increased while decreasing the risk of convergence to local optima. The convergence rate behavior of Rprop is illustrated in Fig. 2 when run on the examples in Fig. 1 as before. In both examples the Rprop method succeeds in reaching the global minimum neighborhood within ten iterations. It passes the area with small gradients quickly since it only uses the signs of the gradient components, not the magnitude. In areas with smooth gradients it will in addition accelerate the step lengths. This adaptivity makes it less sensitive to the initial step length which can also be seen in the examples.

Fig. 2. Gradient descent search: standard, stochastic, momentum, and Rprop methods used on an example error function with/without noise. (a) Error reduction (noiseless). (b) Error reduction (noise).

III. ENERGY OPTIMIZATION FOR SEGMENTATION

As discussed in the introduction, segmentation problems can be approached by using the calculus of variations where an energy functional is defined representing the objective of the problem. The extrema to the functional are found using the Euler-Lagrange equation [10] which is used to derive equations of motion, and the corresponding energy gradients, for the contour [33]. Using these gradients, a gradient descent search in contour space is performed to find a solution to the segmentation problem. Consider, for instance, the derivation of the *weighted region* (see [33]) described by the following functional:

$$f(C) = \iint_{\Omega_C} g(x, y) dx dy \quad (7)$$

where C is a 1D curve embedded in a 2D space, Ω_C is the region inside of C , and $g(x, y)$ is a scalar function. This functional is used to maximize some quantity given by $g(x, y)$ inside C . If $g(x, y) = 1$ for instance, the area will be maximized. Calculating the first variation of Eq. 7 yields the evolution equation:

$$\frac{\partial C}{\partial t} = -g(x, y)\mathbf{n} \quad (8)$$

where $\mathbf{n}$ is the curve normal. Using $g(x, y) = 1$ gives the commonly known “balloon force,” which is a constant flow in the (negative) normal direction.

The contour is often implicitly represented by the zero level of a time dependent signed distance function, known as the level set function. The *level set method* was introduced by Osher and Sethian [6] and includes the advantages of being parameter free, implicit and topologically adaptive. Formally, a contour C is described by $C = \{\mathbf{x} : \phi(\mathbf{x}, t) = 0\}$. The contour C is evolved in time using a set of partial differential equations (PDEs). A motion equation for a parameterized curve $\frac{\partial C}{\partial t} = \gamma \mathbf{n}$ is in general translated into the level set equation $\frac{\partial \phi}{\partial t} = \gamma |\nabla \phi|$, see [33]. Consequently, Eq. 8 gives the familiar

level set equation:

$$\frac{\partial \phi}{\partial t} = -g(x, y) |\nabla \phi| \quad (9)$$

A. Using Momentum and Rprop for Minimizing Level Set Flows

We have noted that evolving the contour according to the Euler-Lagrange equation yields a gradient descent search. Recall that each contour can be represented as a point in the solution space. Thus, we can approximate the direction of the gradient by computing the vector between two subsequent points. In the level set framework we achieve this by taking the difference between two subsequent time instances of the level set function, representing the entire level set function as one vector, $\phi(t_n)$:

$$\nabla f(t_n) \approx \frac{\phi(t_n) - \phi(t_{n-1})}{\Delta t} \quad (10)$$

where $\phi(t)$ is the level set function corresponding to the image, $\Delta t = t_n - t_{n-1}$ and ∇f is the gradient of a cost function f with respect to ϕ .

We can now present the update procedures for the Momentum and Rprop methods. Note that the required modifications to the standard gradient search algorithm are very simple and are directly compatible with any type of level set implementation. This makes it very easy to test and evaluate the proposals in your current implementation.

1) *Level Set Updates*: For the Momentum method, we follow the ideas from Section II-A and incorporate a momentum term in the update of the level set function:

$$s(t_n) = -\eta(1 - \omega) \frac{\tilde{\phi}(t_n) - \phi(t_{n-1})}{\Delta t} + \omega s(t_{n-1}) \quad (11)$$

$$\phi(t_n) = \phi(t_{n-1}) + s(t_n) \quad (12)$$

Here, $\tilde{\phi}(t_n)$ is an intermediate solution, computed by standard evolution of the level set function (see Step 1 below).

For the Rprop method, we can just use Eq. 13 instead of Eq. 11. In Eq. 13 we use the update values estimated by Rprop as described in Section II-B:

$$s(t_n) = -\text{sign}\left(\frac{\tilde{\phi}(t_n) - \phi(t_{n-1})}{\Delta t}\right) * \Delta(t_n) \quad (13)$$

where $*$, as before, denotes elementwise multiplication and $\text{sign}(\cdot)$ the elementwise sign function. The complete procedure works as follows:

- 1) Given the level set function $\phi(t_{n-1})$, compute the next (intermediate) time step $\tilde{\phi}(t_n)$. This is performed by evolving ϕ according to a PDE (such as Eq. 9) using standard techniques (e.g. Euler integration).
- 2) Compute the approximate gradient by Eq. 10.
- 3) Compute a step $s(t_n)$ according to Eq. 11 or Eq. 13 for the Momentum and Rprop method respectively.
- 4) Compute the next time step $\phi(t_n)$ by Eq. 12. Note that this replaces the intermediate level set function computed in Step 1.

IV. METHOD

We will now evaluate the methods by solving five example segmentation tasks. We use both synthetic and real datasets in 2D and 3D. They will hereafter be referred to as *Spiral*, *Retina*, *Shell*, *Angio* and *VascuSynth*. The data sets have ground truth segmentations which are used to evaluate the performance of the methods. We compare the methods with standard and stochastic gradient descent (SGD) [34]. Using the notation in Section II, SGD has the following search vector:

$$s_k = -\eta \nabla f_k + \sigma e^{-\tau k} \xi_k \quad (14)$$

where σ and τ are the noise level and time decay parameter respectively. ξ_k is a standard multivariate Gaussian noise term. Note that $\sigma = 0$ gives the standard gradient descent. The purpose of the time decaying noise in SGD is to avoid early local minima. Eq. 14 is also called Langevin updating and is effectively a simulation of an annealed diffusion process [35].

A. Weighted Region Based Flow

In order to test and evaluate our methods, we have used a simple energy functional to control the segmentation. It is based on a weighted region term (Eq. 7) combined with a penalty on curve length for regularization. The goal is to maximize:

$$f(C) = \iint_{\Omega_C} g(x, y) dx dy - \alpha \oint_C ds \quad (15)$$

where α is a regularization parameter controlling the penalty of the curve length. The target function $g(x, y)$ can be based on intensity values alone, e.g. $g(x, y) = I(x, y) - T$ where $I(x, y)$ is the image intensity and T is a constant threshold intensity for the segmented objects. This choice will result in a regularized thresholding for $\alpha > 0$. However, objects in many real images are not robustly segmented by one threshold parameter, so our experiments use a function $g(x, y)$ which is based on filtering methods. These filters (hereafter denoted as the *target filters*) detect and output positive values on the inside

of line structures, negative values on the outside, and zero on the edges. We refer the reader to [36] for more information on the filter parameters and [37] for details on how to generate the filters.

For 3D datasets, the surface and line integrals in Eq. 15 are translated to volume and surface integrals over a 2D surface embedded in a 3D space. Irrespective of dimensionality, a level set PDE can be derived from Eq. 15 (see [33]):

$$\frac{\partial \phi}{\partial t} = -g(\mathbf{x}) |\nabla \phi| + \alpha \kappa |\nabla \phi| \quad (16)$$

where κ is the curvature of the curve in 2D, and the mean curvature of the surface in 3D.

B. Implementation Details

We have implemented Rprop in Matlab as described in [28]. The level set algorithm has also been implemented in Matlab based on [9], [38]. Reference code can be found online at the site <http://dmforge.itn.liu.se/lsopt/>. Some notable implementation details are:

- 1) Any explicit or implicit time integration scheme can be used in Step 1. Due to its simplicity, we have used explicit Euler integration which might require several inner iterations in Step 1 to advance the level set function by Δt time units.
- 2) In practice we want to apply the momentum and Rprop on the gradient of the *shape* of the contour, rather than on the level set gradient. The latter is in general not related to the shape, since the contour is merely represented as the zero level set. However, if the level set function is confined to a signed distance function, there is a mapping between the contour and the level set function, making the connection between the shape gradient and level set gradient more clear. To enforce this mapping, we reinitialize ϕ after Step 1 and Step 4, using standard methods such as fast marching [39] or fast sweeping [40].
- 3) The parameters η^+ and η^- of Rprop have been fixed to their default values ($\eta^+ = 1.2$ and $\eta^- = 0.5$ respectively) during all experiments.

C. Experiments

The parameters for all five experiments are given in Table I. The narrow band parameter N controls the size of the region close to the zero level set where calculations are performed [38]. This is a computational optimization which is valuable especially in 3D. In theory, this parameter should not affect the result of the propagation. In practice however, the narrow band might provide a “barrier” for the contour, especially for large learning rates, which effectively stops very large steps. This motivates including the narrow band parameter in the experiments. The “curvature weight” parameter, α in Eq. 16, controls the cost of having long contours. It is the balance between the target function $g()$ gain and the curvature cost that creates the local optima, see Eq. 16. The last parameter specifies the convergence criterion, i.e. the segmentation has converged when the infinity norm of ∇f in Eq. 10 is less

TABLE I
EXPERIMENT PARAMETERS

	<i>Spiral</i>	<i>Retina</i>	<i>Shell</i>	<i>Angio</i>	<i>VascuSynth</i>
#Sets	1	3	1	1	100
α	0.001	0.15	0.01		
Δt	10	5	2.5		2
N	{2,4,8,16}		16	12	8
ω	{0.1,0.3,0.5,0.7,0.9}		{0.3,0.5,0.7}		
η, Δ_0	{1,2,4, 8,16}	{1,2,3, 4,5,6}	{1,1.5, 2,3,4}	{0.5,1, 1.5,2,3,4}	{0.5,1, 1.5,2,3,4}
$\Delta_{\min}$	0.1				
$\Delta_{\max}$	{4,40}			{4,8}	
σ	{0, 0.2, 0.4, 0.8, 1.6}		{0, 0.4, 0.8}		
τ	0.02				
$ \nabla f _{\infty}$	< 0.02	< 0.1	< 0.2

Fig. 3. Synthetic test image *Spiral*. (a) Synthetic image. (b) Target function $g(x, y)$.

than the specified value. If the convergence criteria was not fulfilled within 400 iterations the execution was aborted. For our examples, this either means that the solution has left the meaningful part of solution space (i.e. “leaked” into noise) or that it oscillates around a minimum.

The first experiment uses the 2D synthetic image *Spiral* shown in Fig. 3(a). The image shows a spiral with noise. There is a local dip in magnitude along a vertical line in the middle. A magnitude gradient from top to bottom has also been added. These effects may result in local optima in the solution space for certain parameter choices, and will help us test the robustness of our methods. We let the target function $g(x, y)$, see Fig. 3(b), be the target filter output as discussed above in Section IV-A. The bright and dark colors indicate positive and negative values respectively.

The second experiment uses three retinal images from the DRIVE database [41]. One of these is shown in Fig. 4(a). The target function $g(x, y)$ is, as before, the target filter output, see Fig. 4(b). The images have been cropped to the vicinity of the macula lutea.

The third experiment is a 3D variant of experiment 1, a synthetic volume with a shell disturbed by noise as shown in Fig. 5(a). Similar to experiment 1, the volume has a local dip in magnitude along a vertical plane in the middle. There is also a magnitude gradient field going through the volume. In combination with certain parameter choices, these effects may result in local optima in the solution space which will help us evaluate the performance of the methods. The function

Fig. 4. Real test image *Retina*. (a) *Retina* image. (b) Target function $g(x, y)$.

Fig. 5. Visualization of synthetic test volume *Shell*. (a) Synthetic volume. (b) Target function $g(x, y, z)$.

Fig. 6. Visualization of real test volume *Angio*. (a) *Angio* volume. (b) Target function $g(x, y, z)$.

$g(x, y, z)$, see Fig. 5(b), is as usual the output from the target filter.

The fourth test volume is the 3D “Head MRT Angiography” data volume.¹ The dataset is displayed as a volume rendering in Fig. 6(a). The target function $g(x, y, z)$ is the target filter output as before, see Fig. 6(b).

For the fifth and final experiment, we used 100 different 3D volumes generated by *VascuSynth*.² *VascuSynth* is an algorithm and software for synthesizing vascular structures

¹ Available at <http://www.gris.uni-tuebingen.de/edu/areas/scivis/volren/datasets/new.html>.

² Available at <http://vascusynth.cs.sfu.ca/Welcome.html>.

Fig. 7. Visualization of a generated volume *VascuSynth*. (a) *VascuSynth* volume. (b) Target function $g(x, y, z)$.

Fig. 8. Example results for the synthetic *Spiral* image. (a) Momentum $\{N = 4, \omega = 0.7, \text{ and } \eta = 8\}$. (b) ROC curves (momentum). (c) Rprop $\{N = 8, \Delta_{max} = 40, \text{ and } \Delta_0 = 8\}$. (d) ROC curves (Rprop)

[42], [43]. The default settings were used in the generation of the volumes. One of these volumes is displayed as a volume rendering in Fig. 7(a). The target function $g(x, y, z)$ is the target filter output as before, see Fig. 7(b).

V. RESULTS

In order to quantify the quality of a segmentation, we use its corresponding standard *Receiver Operating Characteristic* (ROC) curve [44]. The ROC is a plot of the true positive rate (TPR) vs. the false positive rate (FPR) of the segmentation for different threshold choices T . It describes the relation between the *sensitivity* ($=$ TPR) and the *specificity* ($= 1 - \text{FPR}$), i.e. the relation between correctly segmenting the object and correctly ignoring the background. As a quality measure we use *Youden's Index* Y [45], which is a summary statistic of the ROC curve. It is defined as:

$$Y = \max_T (\text{Sensitivity} + \text{Specificity} - 1) \quad (17)$$

$$= \max_T (\text{TPR} - \text{FPR}) \quad (18)$$

Fig. 9. Example results for the *Retina* experiment. (a) Momentum $\{N = 2, \omega = 0.7, \text{ and } \eta = 2\}$. (b) Rprop $\{N = 4, \Delta_{max} = 40, \text{ and } \Delta_0 = 4\}$. (c) ROC curves (Momentum). (d) ROC curves (Rprop)

where T is the threshold. Y is simply the point on the ROC curve with the largest height over the diagonal ($\text{TPR} = \text{FPR}$). A perfect segmentation without false positives will give $Y = 1$. The worst possible (random) segmentation would lie on the diagonal, yielding $Y = 0$. In the context of level set segmentation, the optimal threshold is per definition $T = 0$, i.e. the zero level set.

In the following we define *acceptable segmentations* and *good segmentations* as runs where $Y > 0.75$ and $Y > 0.90$ respectively. “ Y_C ” denotes the Youden’s index for converged runs only; the convergence criterion is defined in IV-C above. The numbers in parentheses give the standard deviation of the measures. In addition, we define *Weighted iterations* as the number of iterations before convergence divided by the resulting Y of the run. This lets us compare the effectiveness of the methods even if they converge to different solutions. In order to compare real time performance we also look at the clocktime per iteration. The results when testing all combinations of the parameters in Table I are shown in Table II for experiment 1–4, Table III for experiment 5.

Examples of the resulting segmentations for the proposed methods are shown in Fig. 8, Fig. 9, Fig. 10 and Fig. 11. The figures show both the best segmentation achieved and all the ROC curves which give an indication of parameter sensitivities for the methods.

VI. DISCUSSION

Note that since the focus of this paper is to compare different optimization methods given a certain input, the quality and accuracy of the resulting segmentations may not be comparable to the reference segmentations. The point of these experimental segmentations is to highlight the advantages

TABLE II
RESULTS FOR EXPERIMENT 1–4

(a) <i>Spiral</i>					(b) <i>Retina</i>				
	GD	SGD	Momentum	R-prop	GD	SGD	Momentum	R-prop	
# of runs per dataset	20	80	100	40	24	96	120	48	
Convergence ratio	30%	40%	43%	82%	19% (2.4)	27% (0.6)	30% (0.5)	95% (1.2)	
Y (best/mean/std)	0.99/0.74/0.11	0.99/0.78/0.13	0.99/0.81/0.13	0.99/0.98/0.05	0.79/0.69/0.06	0.80/0.72/0.12	0.80/0.73/0.11	0.80/0.77/0.06	
Y_C (best/mean/std)	0.71/0.71/0.00	0.99/0.73/0.10	0.99/0.74/0.08	0.99/0.99/0.00	0.79/0.70/0.05	0.79/0.73/0.10	0.80/0.74/0.06	0.80/0.77/0.06	
Ratio $Y, Y_C > 0.75$	35%, 0%	46%, 5%	48%, 6%	100%, 82%	36%, 4%	46%, 12%	49%, 14%	76%, 76%	
Ratio $Y, Y_C > 0.90$	15%, 0%	32%, 5%	32%, 6%	88%, 82%	0%, 0%	0%, 0%	0%, 0%	0%, 0%	
# weighted iter. (std)	199.6 (75.6)	194.3 (70.4)	189.3 (69.9)	184.5 (77.0)	399.6 (95.4)	352.6 (79.5)	324.5 (83.8)	195.2 (81.4)	
Clock time/iter. (std)	0.2 (0.03)	0.2 (0.05)	0.2 (0.11)	0.2 (0.04)	0.4 (0.10)	0.4 (0.12)	0.5 (0.14)	0.5 (0.19)	

(c) <i>Shell</i>					(d) <i>Angio</i>				
	GD	SGD	Momentum	R-prop	GD	SGD	Momentum	R-prop	
# of runs per dataset	5	10	15	10	6	12	18	12	
Convergence ratio	80%	80%	87%	90%	50%	50%	56%	100%	
Y (best/mean/std)	0.75/0.71/0.10	0.99/0.75/0.11	0.96/0.77/0.08	0.99/0.97/0.04	0.84/0.71/0.16	0.84/0.75/0.14	0.84/0.77/0.09	0.84/0.84/0.01	
Y_C (best/mean/std)	0.75/0.71/0.06	0.99/0.73/0.05	0.76/0.74/0.02	0.99/0.99/0.00	0.84/0.63/0.18	0.84/0.71/0.12	0.83/0.71/0.09	0.84/0.84/0.01	
Ratio $Y, Y_C > 0.75$	40%, 40%	40%, 20%	40%, 27%	100%, 90%	67%, 17%	75%, 33%	78%, 33%	100%, 100%	
Ratio $Y, Y_C > 0.90$	0%, 0%	20%, 10%	13%, 0%	90%, 90%	0%, 0%	0%, 0%	0%, 0%	0%, 0%	
# weighted iter. (std)	105.1 (35.0)	108.3 (39.6)	96.9 (32.5)	84.1 (13.2)	253.3 (123.6)	249.1 (119.0)	240.7 (109.8)	145.2 (36.6)	
Clock time/iter. (std)	12.9 (0.7)	13.2 (0.6)	17.5 (0.9)	18.9 (1.0)	26.9 (10.0)	25.9 (10.1)	34.5 (19.9)	45.6 (3.4)

Fig. 10. Example results for the synthetic *Shell* volume. (a) Momentum $\{N = 16, \omega = 0.5, \text{ and } \eta = 4\}$. (b) ROC curves (momentum). (c) Rprop $\{N = 16, \Delta_{max} = 40, \text{ and } \Delta_0 = 4\}$. (d) ROC curves (Rprop).

of the Momentum and Rprop methods in contrast to the ordinary gradient descent. By optimizing the input data (i.e. the filters generating the target function) the segmentations can be improved.

The experiments show the standard gradient method, i.e. SGD with $\sigma = 0$, converges to the closest local optimum with resulting poor segmentations. SGD with $\sigma > 0$ can escape early local optima with the right amount of noise.

However, choosing σ is nontrivial since too much noise causes the solution contour to jump into the surrounding noise. In addition, GD and SGD converges slowly as their step lengths are proportional to the magnitude of the gradient. These magnitudes can be very low in regions with low signal to noise ratios. A large learning rate η can compensate for this, but will often give rise to oscillations. For certain combinations of parameters, SGD results in good segmentations.

We can also see that the Momentum method can be fast and robust to local optima given an appropriate set of parameters. In experiment *Spiral* for example, Fig. 8(b), 32% of the runs achieve good segmentation ($Y > 0.90$). Finding these parameters are not easy however as about 52% of the runs do not reach an acceptable quality. In the *Retina* experiment, Fig. 9(c), the ROC-curves of the Momentum method have a lot of variation, again showing large sensitivity to its parameters. A set of parameters that is robust to local optima and gives high segmentation velocity may not converge as it instead oscillates around a solution. This can also be seen in *Retina*, where 49% of the runs achieve acceptable segmentations ($Y > 0.75$) even though only 30% converge. Because the Momentum method still uses the magnitude of the gradient, it is sensitive to larger regions with low gradient magnitudes. This can be seen in experiment *Shell*, where all convergent runs stop close to the first wide saddle point. The Momentum method is however robust to large but very local changes in gradient due to the momentum term.

Rprop, in contrast, is very sensitive to large changes in the direction of the gradient since it uses the signs of the gradient components. Rprop is, for the same reason, completely insensitive to the magnitude of the gradients. This can be seen in e.g. *Shell*, where Rprop passes the first wide saddle point where the other methods stop or struggle. In the *Spiral* experiment, see Fig. 8(d), Rprop have 40 ROC-curves with small variation.

TABLE III
RESULTS OF EXPERIMENT 5 *VascuSynth*

	GD	SGD	Momentum	R-prop
# of runs per dataset	6	12	18	12
Conv. ratio (std)	67% (3.3)	67% (3.0)	73% (2.9)	99% (2.5)
Y (best/mean/std)	0.93/0.86/0.04	0.93/0.89/0.05	0.93/0.91/0.02	0.93/0.93/0.01
Y_C (best/mean/std)	0.93/0.93/0.01	0.93/0.93/0.01	0.93/0.93/0.01	0.93/0.93/0.01
Ratio $Y, Y_C > 0.75$	83%, 100%	92%, 100%	95%, 100%	100%, 100%
Ratio $Y, Y_C > 0.90$	83%, 100%	83%, 100%	91%, 100%	100%, 100%
# weighted iter. (std)	86.4 (12.4)	83.8 (11.2)	79.2 (11.8)	75.1 (10.0)
Clock time/iter. (std)	8.8 (0.7)	9.0 (0.6)	13.4 (1.2)	15.1 (0.9)

Fig. 11. Example results for the *Angio* volume. (a) Momentum $\{N = 12, \omega = 0.5, \text{ and } \eta = 2\}$. (b) ROC curves (Momentum). (c) Rprop $\{N = 12, \Delta_{\max} = 8, \text{ and } \Delta_0 = 3\}$. (d) ROC curves (Rprop).

88% of these runs are good segmentations. For *Retina*, Fig. 9, the ROC-curves of Rprop have low variation. About 76% of these runs result in acceptable segmentations. The experiments clearly demonstrate Rprops robustness against parameter changes and insensitivity towards local optima.

For data sets where the magnitude of the gradient has a wide distribution of values, it can be difficult to find a constant learning rate that works for the SGD and Momentum method, see experiment *Spiral* and *Retina* for instance. Rprop solves this by continuously adapting the learning rates according to the behavior of the local gradients. The ROC-curves for the experiments confirm this as they show low sensitivity for parameter changes for Rprop in contrast to the Momentum method.

The *VascuSynth* experiment evaluates the methods on 100

different volumes. The results and behaviors we discussed above are confirmed in this experiment, see Table III. The parameter insensitivity of RPROP stands out as 99% of the runs consistently converges to good solutions.

The implementations of the methods are simple, but the interpretation of the gradients must be given special care as described in section IV-B above. Currently we rely on reinitializing the level set function to maintain the signed distance property, but we plan to investigate more elegant solutions to this problem, following the ideas in [46].

The Momentum and Rprop methods converge in fewer iterations than the standard/SGD method but are slower per iteration. The difference in time per iteration is insignificant in 2D but larger in 3D, where the standard/SGD method is between 50-100% faster than the others. The Momentum method can be as fast or faster than the Rprop method for a good set of parameters. However, looking at the mean performance for a large set of parameters, Rprop converges in significantly less number of iterations as the experiments show. They have approximately the same time complexity per iteration.

VII. CONCLUSION

Image segmentation using the level set method involves optimization in contour space. In this context, gradient descent is the standard optimization method. We have discussed the weaknesses of this method and proposed using the Momentum and Rprop methods, very simple modifications of gradient descent, commonly used in the machine learning community. The modifications are directly compatible to any type of level set implementation, and downloadable reference code with examples is available online. In addition, we have shown in a series of experiments how the solutions are improved by these methods. Using Momentum and Rprop, the optimization gets less sensitive to local optima and the convergence rate is improved. Rprop in particular is also shown to be very insensitive to parameter settings and to different gradient behaviors. This is very important in practical use since Rprop will work out-of-the-box in many applications using only the standard values of its parameters. In contrast to much of the previous work, we have improved the solutions by changing the method of solving the optimization problem rather than modifying the energy functional.

REFERENCES

- [1] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Comput. Vis.*, vol. 1, no. 4, pp. 321–331, 1988.
- [2] L. D. Cohen, "On active contour models and balloons," *CVGIP, Image Understand.*, vol. 53, no. 2, pp. 211–218, Mar. 1991.
- [3] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, "Gradient flows and geometric active contour models," in *Proc. Int. Conf. Comp. Vis.*, Jun. 1995, pp. 810–815.
- [4] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," in *Proc. IEEE Int. Conf. Comput. Vis.*, Jun. 1995, pp. 694–699.
- [5] D. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions and associated variational problems," *Commun. Pure Appl. Math.*, vol. 42, no. 5, pp. 577–685, 1989.
- [6] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations," *J. Comput. Phys.*, vol. 79, no. 1, pp. 12–49, Nov. 1988.
- [7] T. Chan and L. Vese, "A level set algorithm for minimizing the Mumford–Shah functional in image processing," in *Proc. IEEE Workshop Variat. Level Set Meth. Comput. Vis.*, Mar. 2001, pp. 161–168.
- [8] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Trans. Image Process.*, vol. 10, no. 2, pp. 266–277, Feb. 2001.
- [9] S. Osher and R. Fedkiw, *Level Set Dynamic Implicit Surfaces*. New York: Springer-Verlag, 2003.
- [10] P. M. Morse and H. Feshbach, "The variational integral and the Euler equations," in *Proc. Meth. Theor. Phys.*, I, May 1953, pp. 276–280.
- [11] G. Charpiat, R. Keriven, J.-P. Pons, and O. Faugeras, "Designing spatially coherent minimizing flows for variational problems based on active contours," in *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2, Oct. 2005, pp. 1403–1408.
- [12] G. Sundaramoorthi, A. Yezzi, and A. Mennucci, "Sobolev active contours," *Int. J. Comput. Vis.*, vol. 73, no. 3, pp. 345–366, 2007.
- [13] R. B. Kearfott, *Rigorous Global Search: Continuous Problems (Non-convex Optimization and Its Applications)*, vol. 13. Dordrecht, The Netherlands: Kluwer, 1996.
- [14] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
- [15] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, no. 6, pp. 1087–1092, 1953.
- [16] D. J. Wales and J. P. K. Doye, "Global optimization by basin-hopping and the lowest energy structures of Lennard–Jones clusters containing up to 110 atoms," *J. Phys. Chem. A*, vol. 101, no. 28, pp. 5111–5116, 1997.
- [17] Y. N. Law, H. K. Lee, and A. Yip, "A multiresolution stochastic level set method for Mumford–Shah image segmentation," *IEEE Trans. Image Process.*, vol. 17, no. 12, pp. 2289–2300, Dec. 2008.
- [18] A. Chambolle, "An algorithm for total variation minimization and applications," *J. Math. Imag. Vis.*, vol. 20, nos. 1–2, pp. 89–97, 2004.
- [19] T. F. Chan, S. Esedoglu, and M. Nikolova, "Algorithms for finding global minimizers of image segmentation and denoising models," *SIAM J. Appl. Math.*, vol. 66, no. 5, pp. 1632–1648, 2006.
- [20] X. Bresson, S. Esedoglu, P. Vanderghynst, J.-P. Thiran, and S. Osher, "Fast global minimization of the active contour/snake model," *J. Math. Imag. Vis.*, vol. 28, no. 2, pp. 151–167, Jun. 2007.
- [21] J. Solem, N. Overgaard, M. Persson, and A. Heyden, "Fast variational segmentation using partial extremal initialization," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, Jun. 2006, pp. 1099–1105.
- [22] L. Grady and C. Alvino, "The piecewise smooth Mumford–Shah functional on an arbitrary graph," *IEEE Trans. Image Process.*, vol. 18, no. 11, pp. 2547–2561, Nov. 2009.
- [23] Y. Boykov and V. Kolmogorov, "Computing geodesics and minimal surfaces via graph cuts," in *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 1, Oct. 2003, pp. 26–33.
- [24] H. Chang, Q. Yang, M. Auer, and B. Parvin, "Modeling of front evolution with graph cut optimization," in *Proc. IEEE Int. Conf. Image Process.*, vol. 1, Oct. 2007, pp. 241–244.
- [25] H. Chang, M. Auer, and B. Parvin, "Structural annotation of em images by graph cut," in *Proc. IEEE Int. Symp. Biomed. Imag., Nano Macro*, Jul. 2009, pp. 1103–1106.
- [26] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning Internal Representations Error Propagation*. Cambridge, MA: MIT Press, 1986, ch. 8, pp. 318–362.
- [27] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 1, Jun. 1993, pp. 586–591.
- [28] M. Riedmiller and H. Braun, "Rprop, description and implementation details," Inst. für Logik, Univ. Karlsruhe, Karlsruhe, Germany, Tech. Rep. 94S145, 1994.
- [29] G. Låthén, T. Andersson, R. Lenz, and M. Borga, "Momentum based optimization methods for level set segmentation," in *Proc. Int. Conf. Scale Space Variat. Meth. Comput. Vis.*, LNCS 5567, Jun. 2009, pp. 124–136.
- [30] T. Andersson, G. Låthén, R. Lenz, and M. Borga, "A fast optimization method for level set segmentation," in *Proc. Scandinavian Conf. Image Anal.*, LNCS 5575, Jun. 2009, pp. 400–409.
- [31] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer-Verlag, 2006.
- [32] W. Schiffmann, M. Joost, and R. Werner, "Comparison of optimized backpropagation algorithms," in *Proc. Eur. Symp. Artif. Neural Netw.*, 1993, pp. 97–104.
- [33] R. Kimmel, "Fast edge integration," in *Geometric Level Set Methods in Imaging, Vision and Graphics*. New York: Springer-Verlag, 2003.
- [34] W. A. Gardner, "Learning characteristics of stochastic-gradient-descent algorithms: A general study, analysis, and critique," *Signal Process.*, vol. 6, no. 2, pp. 113–133, Apr. 1984.
- [35] T. Rögnvaldsson, "On Langevin updating in multilayer perceptrons," *Neural Comput.*, vol. 6, no. 5, pp. 916–926, 1993.
- [36] G. Låthén, J. Jonasson, and M. Borga, "Blood vessel segmentation using multi-scale quadrature filtering," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 762–767, Jun. 2010.
- [37] G. H. Granlund and H. Knutsson, *Signal Processing for Computer Vision*. Norwood, MA: Kluwer, 1995.
- [38] D. Peng, B. Merriman, S. Osher, H.-K. Zhao, and M. Kang, "A PDE-based fast local level set method," *J. Comput. Phys.*, vol. 155, no. 2, pp. 410–438, 1999.
- [39] J. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proc. Nat. Acad. Sci. USA*, vol. 93, pp. 1591–1595, Feb. 1996.
- [40] H.-K. Zhao, "A fast sweeping method for Eikonal equations," *Math. Comput.*, vol. 74, pp. 603–627, May 2005.
- [41] J. Staal, M. Abramoff, M. Niemeijer, M. Viergever, and B. van Ginneken, "Ridge based vessel segmentation in color images of the retina," *IEEE Trans. Med. Imag.*, vol. 23, no. 4, pp. 501–509, Apr. 2004.
- [42] G. Hamarneh and P. Jassi, "Vascusynth: Simulating vascular trees for generating volumetric image data with ground truth segmentation and tree analysis," *Comput. Med. Imag. Graph.*, vol. 34, no. 8, pp. 605–616, 2010.
- [43] P. Jassi and G. Hamarneh, "Vascusynth: Vascular tree synthesis software," *Insight J.*, pp. 1–12, Jan.–Jun. 2011.
- [44] D. Green and J. Swets, *Signal Detection Theory and Psychophysics*. New York: Wiley, 1966.
- [45] W. J. Youden, "Index for rating diagnostic tests," *Cancer*, vol. 3, no. 1, pp. 32–35, 1950.
- [46] S. Chen, G. Charpiat, and R. J. Radke, "Converting level set gradients to shape gradients," in *Proc. 11th Eur. Conf. Comp. Vis.*, 2010, pp. 715–728.

Thord Andersson, photograph and biography are not available at the time of publication.

Gunnar Låthén, photograph and biography are not available at the time of publication.

Reiner Lenz, photograph and biography are not available at the time of publication.

Magnus Borga, photograph and biography are not available at the time of publication.