Reprint from Proceedings of the 16:th Scandinavian Conference on Image Analysis (SCIA). Oslo, Norway, 2009

A fast optimization method for level set segmentation

Thord Andersson[1,3], Gunnar Läthén[2,3], Reiner Lenz[2,3], and Magnus Borga[1,3]

[1] Department of Biomedical Engineering, Linköping University
[2] Department of Science and Technology, Linköping University
[3] Center for Medical Image Science and Visualization (CMIV), Linköping University

Abstract. Level set methods are a popular way to solve the image segmentation problem in computer image analysis. A contour is implicitly represented by the zero level of a signed distance function, and evolved according to a motion equation in order to minimize a cost function. This function defines the objective of the segmentation problem and also includes regularization constraints. Gradient descent search is the de facto method used to solve this optimization problem. Basic gradient descent methods, however, are sensitive for local optima and often display slow convergence. Traditionally, the cost functions have been modified to avoid these problems. In this work, we instead propose using a modified gradient descent search based on resilient propagation (Rprop), a method commonly used in the machine learning community. Our results show faster convergence and less sensitivity to local optima, compared to traditional gradient descent.

Key words: image segmentation, level set method, optimization, gradient descent, Rprop, variational problems, active contours

1 Introduction

In order to find objects such as tumors in medical images or roads in satellite images, an image segmentation problem has to be solved. One approach is to use calculus of variations. In this context, a contour parameterizes an energy functional defining the objective of the segmentation problem. The functional depends on properties of the image such as gradients, curvatures and intensities, as well as regularization terms, e.g. smoothing constraints. The goal is to find the contour which, depending on the formulation, maximizes or minimizes the energy functional. In order to solve this optimization problem, the gradient descent method is the de facto standard. 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 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 small local optima and slow/poor 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 [1, 2], 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.

In contrast, we propose using a modified gradient descent search based on *resilient propagation (Rprop)* [3][4], a method commonly used in the machine learning community. In order to avoid the typical problems of gradient descent search, *Rprop* provides a simple but effective modification which uses individual (one per parameter) adaptive step sizes and considers only the sign of the gradient. This modification makes Rprop more robust to local optima and avoids the harmful influence of the size of the gradient on the step size. The individual adaptive step sizes also allow for cost functions with very different behaviors along different dimensions because there is no longer a single step size that should fit them all. In this paper, we show how Rprop can be used for image segmentation using level set methods. The results show faster convergence and less sensitivity to local optima.

The paper will proceed as follows. In Section 2, we will describe gradient descent with Rprop and give an example of a representative behavior. Then, Section 3 will discuss the level set framework and how Rprop can be used to solve segmentation problems. Experiments, where segmentations are made using Rprop for gradient descent, are presented in Section 4 together with implementation details. In Section 5 we discuss the results of the experiments and Section 6 concludes the paper and presents ideas for future work.

2 Gradient descent with Rprop

Gradient descent is a very common optimization method which appeal lies in the combination of its generality and simplicity. It can handle many types of 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 and fast to calculate since they only involve the first order derivatives of the cost function. Unfortunately, gradient descent is known to exhibit slow convergence and to be sensitive to local optima for many practical problems. 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 [5]. Rprop, proposed by the machine learning community [3], provides an intermediate level between the simplicity of gradient descent and the complexity of these more theoretically sophisticated variants.

Gradient descent may be expressed using a standard line search optimization:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s}_k \tag{1}$$

$$\boldsymbol{s}_k = \alpha_k \boldsymbol{p}_k \tag{2}$$

where $\boldsymbol{x}_k$ is the current iterate, $\boldsymbol{s}_k$ is the next step consisting of length α_k and direction $\boldsymbol{p}_k$. To guarantee convergence, it is often required that $\boldsymbol{p}_k$ be a descent direction while α_k 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 $\boldsymbol{p}_k = -\boldsymbol{\nabla} f_k$, where f is the cost function, while α_k satisfies the *Wolfe conditions* [5].

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 optima 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 was developed [3] to overcome these inherent disadvantages of standard gradient descent using adaptive step-sizes $\boldsymbol{\Delta}_k$ called *update-values*. There is one update-value per dimension in $\boldsymbol{x}$, i.e. $\dim(\boldsymbol{x}_k) = \dim(\boldsymbol{\Delta}_k)$. 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. There are other methods using both adaptive step-sizes and the size of the gradient, but the unpredictable behavior of the derivatives often counter the careful adaption of the step-sizes. 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 [6].

We will now describe the Rprop algorithm briefly, but for implementation details of Rprop we refer to [4]. For Rprop, we choose a search direction $\boldsymbol{s}_k$ according to:

$$\boldsymbol{s}_k = -\operatorname{sign}\left(\boldsymbol{\nabla} f_k\right) * \boldsymbol{\Delta}_k \tag{3}$$

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

$$\boldsymbol{\Delta}_k^i = \begin{cases} \min\left(\boldsymbol{\Delta}_{k-1}^i \cdot \eta^+, \Delta_{\max}\right) & , \boldsymbol{\nabla}^i f_k \cdot \boldsymbol{\nabla}^i f_{k-1} > 0 \\ \max\left(\boldsymbol{\Delta}_{k-1}^i \cdot \eta^-, \Delta_{\min}\right) & , \boldsymbol{\nabla}^i f_k \cdot \boldsymbol{\nabla}^i f_{k-1} < 0 \\ \boldsymbol{\Delta}_{k-1}^i & , \boldsymbol{\nabla}^i f_k \cdot \boldsymbol{\nabla}^i f_{k-1} = 0 \end{cases} \tag{4}$$

where $\boldsymbol{\nabla}^i f_k$ denotes the partial derivative i in the gradient. Note that this is Rprop without backtracking as described in [4]. 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 greater steps in favorable directions, causing the rate of convergence to be increased while overstepping eventual local optima.

3 Energy optimization for segmentation

As discussed in the introduction, segmentation problems can be approached by using the calculus of variations. Typically, an energy functional is defined representing the objective of the segmentation problem. The functional is described in terms of the contour and the relevant image properties. The goal is to find a contour that represents a solution which, depending on the formulation, maximizes or minimizes the energy functional. These extrema are found using the Euler-Lagrange equation which is used to derive equations of motion, and the corresponding energy gradients, for the contour [7]. Using these gradients, a gradient descent search in contour space is commonly used to find a solution to the segmentation problem. Consider, for instance, the derivation of the *weighted region* (see [7]) described by the following functional:

$$E(C) = \iint_{\Omega_C} f(x, y) dx dy \tag{5}$$

where C is a 1D curve embedded in a 2D space, Ω_C is the region inside of C, and $f(x, y)$ is a scalar function. This functional is used to maximize some quantity given by $f(x, y)$ inside C. If $f(x, y) = 1$ for instance, the area will be maximized. Calculating the first variation of Eq. 5 yields the evolution equation:

$$\frac{\partial C}{\partial t} = -f(x, y)\boldsymbol{n} \tag{6}$$

where $\boldsymbol{n}$ is the curve normal. If we anew set $f(x, y) = 1$, this will give a constant flow in the normal direction, commonly known as the "balloon force".

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 [8] and includes the advantages of being parameter free, implicit and topologically adaptive. Formally, a contour C is described by $C = \{\boldsymbol{x} : \phi(\boldsymbol{x}, t) = 0\}$. The contour C is evolved in time using a set of partial differential equations (PDEs). A motion equation for a parameterized curve $\frac{\partial C}{\partial t} = \gamma \boldsymbol{n}$ is in general translated into the level set equation $\frac{\partial \phi}{\partial t} = \gamma |\boldsymbol{\nabla} \phi|$, see [7]. Consequently, Eq. 6 gives the familiar level set equation:

$$\frac{\partial \phi}{\partial t} = -f(x, y) |\boldsymbol{\nabla} \phi| \tag{7}$$

3.1 Rprop for energy optimization using level set flow

When solving an image segmentation problem, we can represent the entire level set function (corresponding to the image) as one vector, $\phi(t_n)$. In order to perform a gradient descent search as discussed earlier, we can approximate the gradient as the finite difference between two time instances:

$$\boldsymbol{\nabla} f(t_n) \approx \frac{\phi(t_n) - \phi(t_{n-1})}{\Delta t} \tag{8}$$

where $\Delta t = t_n - t_{n-1}$ and $\boldsymbol{\nabla} f$ is the gradient of a cost function f as discussed in Section 2. Using the update values estimated by Rprop (as in Section 2), we can update the level set function:

$$s(t_n) = -\text{sign}\left(\frac{\widetilde{\phi}(t_n) - \phi(t_{n-1})}{\Delta t}\right) * \boldsymbol{\Delta}(t_n) \tag{9}$$

$$\phi(t_n) = \phi(t_{n-1}) + s(t_n) \tag{10}$$

where $*$ as before denotes elementwise multiplication. The complete procedure works as follows:

```
Procedure UpdateLevelset
1 Given the level set function $\phi(t_{n-1})$, compute the next (intermediate)
  time step $\widetilde{\phi}(t_n)$. This is performed by evolving $\phi$ according to a PDE
  (such as Eq. 7) using standard techniques (e.g. Euler integration).
2 Compute the approximate gradient by Eq. 8.
3 Compute a step $s(t_n)$ according to Eq. 9. This step effectively modifies
  the gradient direction by using the Rprop derived update values.
4 Compute the next time step $\phi(t_n)$ by Eq. 10. Note that this replaces the
  intermediate level set function computed in Step 1.
```

The procedure is very simple and can be used directly with any type of level set implementation.

4 Experiments

We will now evaluate our idea by solving two example segmentation tasks using a simple energy functional. Both examples use 1D curves in 2D images but our approach also supports higher dimensional contours, e.g. 2D surfaces in 3D volumes.

4.1 Implementation details

We have implemented Rprop in Matlab as described in [4]. The level set algorithm has also been implemented in Matlab based on [9, 10]. Some notable implementation details are:

– 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.

– The level set function is reinitialized (reset to a signed distance function) after Step 1 and Step 4. This is typically performed using the fast marching [11] or fast sweeping algorithms [12]. This is required for stable evolution in time due to the use of explicit Euler integration in Step 1.
– The reinitializations of the level set function can disturb the adaptation of the individual step sizes outside the contour, causing spurious "islands" close to the contour. In order to avoid them we set the maximum step size to a low value once the target function integral has converged:

$$\left(\iint_{\Omega_{C(t)}} f(x,y)dxdy - \iint_{\Omega_{C(t-k)}} f(x,y)dxdy\right) < 0 \qquad (11)$$

where k denotes the time under which the target function integral should not have increased.

4.2 Weighted region based flow

In order to test and evaluate our idea, we have used a simple energy functional to control the segmentation. It is based on a weighted region term (Eq. 5) combined with a penalty on curve length for regularization. The goal is to maximize:

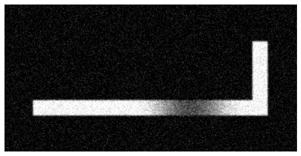
$$E(C) = \iint_{\Omega_C} f(x,y)dxdy - \alpha \oint_C ds \qquad (12)$$

where α is a regularization parameter adjusting the penalty of the curve length. The target function $f(x,y)$ is here the real part of a global phase image, derived from the original image using the method in [13]. This method uses quadrature filters [14] across multiple scales to generate a global phase image that represents line structures. The function $f(x,y)$ will have positive values on the inside of linear structures, negative on the outside, and zero on the edges. A level set PDE can be derived from Eq. 12 (see [7]) just as in section Section 3:

$$\frac{\partial \phi}{\partial t} = -f(x,y)\left|\boldsymbol{\nabla}\phi\right| + \alpha\kappa\left|\boldsymbol{\nabla}\phi\right| \qquad (13)$$

where κ is the curvature of the contour.

We will now evaluate gradient descent with and without Rprop using Eq. 13 on a synthetic test image shown in Figure 1(a). The image illustrates a line-like structure with a local dip in contrast. This dip results in a local optimum in the contour space, see Figure 2, and will help us test the robustness of our method. We let the target function $f(x,y)$, see Figure 1(b), be the real part of the global phase image as discussed above. The bright and dark colors indicate positive and negative values respectively. Figure 2 shows the results after an ordinary gradient search has converged. We define convergence as $\left|\boldsymbol{\nabla} f\right|_\infty < 0.03$ (using the L^∞-norm), with $\boldsymbol{\nabla} f$ given in Eq. 8. For this experiment we used parameters $\alpha = 0.7$ and we reinitialized the level set function every fifth iteration. For comparison, Figure 3 shows the results after running our method using


(a) Synthetic test image

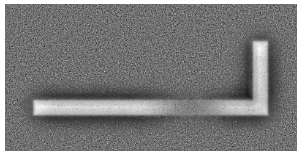

(b) Target function $f(x, y)$

Fig. 1. Synthetic test image spawning a local optima in the contour space.

default Rprop parameters $\eta^+ = 1.2, \eta^- = 0.5$, and other parameters set to $\Delta_0 = 2.5, s_{max} = 30$ and $\Delta t = 5$. Plots of the energy functional for both experiments are shown in Figure 4. Here, we plot the weighted area term and the length penalty term separately, to illustrate the balance between the two. Note that the functional without Rprop in Figure 4(a) is monotonically increasing as would be expected of gradient descent, while the functional with Rprop visits a number of local maxima during the search. The effect of setting the maximum step size to a low value at $t = 160$, as discussed above (Eq. 11), effectively cancels the issue of spurious "islands" close to the contour in only two iterations. As a

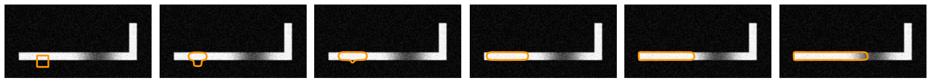

(a) $t = 0$ (b) $t = 40$ (c) $t = 100$ (d) $t = 170$ (e) $t = 300$ (f) $t = 870$

Fig. 2. Iterations without Rprop. (Time units per iteration: $\Delta t = 5$)

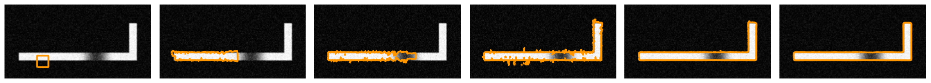
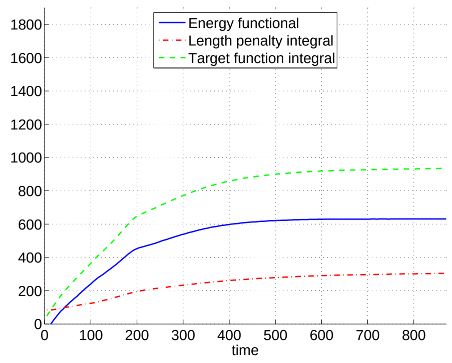
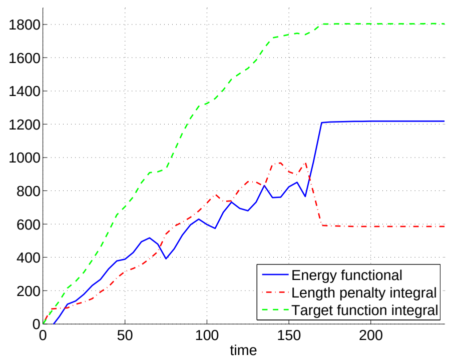

(a) $t = 0$ (b) $t = 60$ (c) $t = 75$ (d) $t = 160$ (e) $t = 170$ (f) $t = 245$

Fig. 3. Iterations using Rprop. (Time units per iteration: $\Delta t = 5$)

second test image we used a 458×265 retinal image from the DRIVE database [15], as seen in Figure 5. The target function $f(x, y)$ is, as before, the real part of the global phase image. Figure 5 shows the results after an ordinary gradient search has converged using the parameter $\alpha = 0.15$, reinitialization every tenth time unit and with the initial condition given in Figure 5(a). We have again used $|\nabla f|_\infty < 0.03$ as convergence criteria. If we instead use Rprop together with the parameters $\alpha = 0.15, \Delta_0 = 4, s_{max} = 10$ and $\Delta t = 10$, we get the result in Figure 6. The energy functionals are plotted in Figure 7, showing the convergence of both methods.

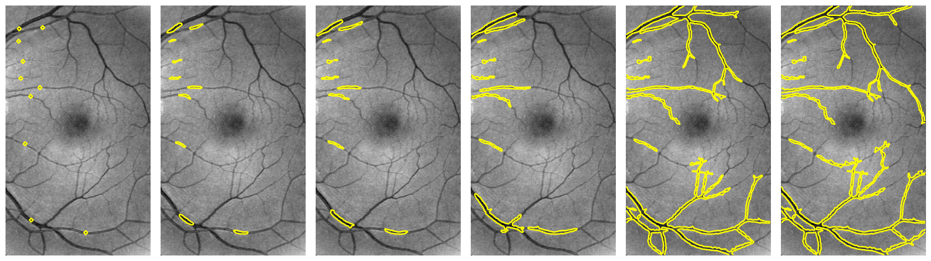


(a) Without Rprop



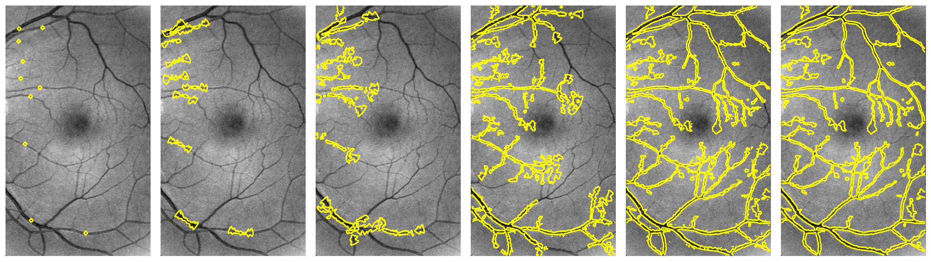
(b) With Rprop

Fig. 4. Plots of energy functionals for synthetic test image in Figure 1(a).



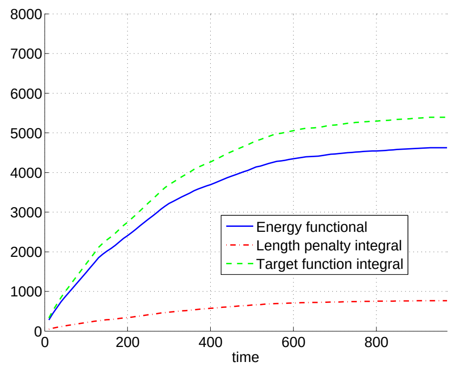
(a) $t = 0$ (b) $t = 20$ (c) $t = 40$ (d) $t = 100$ (e) $t = 500$ (f) $t = 970$

Fig. 5. Iterations without Rprop. (Time units per iteration: $\Delta t = 10$)

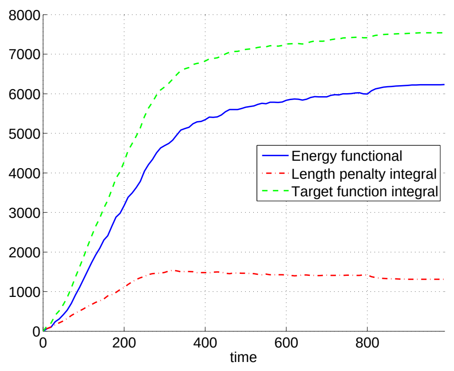


(a) $t = 0$ (b) $t = 40$ (c) $t = 80$ (d) $t = 200$ (e) $t = 600$ (f) $t = 990$

Fig. 6. Iterations using Rprop. (Time units per iteration: $\Delta t = 10$)



(a) Without Rprop



(b) With Rprop

Fig. 7. Plots of energy functionals for the retinal image as seen in Figure 5.

5 Discussion

The synthetic test image in Figure 1(a) spawns a local optimum in the contour space when we apply the set of parameters used in our first experiment. The standard gradient descent method converges as expected, see Figure 2, to this local optimum. Gradient descent with Rprop, however, accelerates along the linear structure due to the stable sign of the gradient in this area. The adaptive step-sizes of Rprop consequently grow large enough to overstep the local optimum. This is followed by a fast convergence to the global optimum. The progress of the method is shown in Figure 3.

Our second example evaluates our method using real data from a retinal image. The standard gradient descent method does not succeed to segment blood vessels where the signal to noise ratio is low. This is due to the local optima in these areas, induced by noise and blood vessels with low contrast. Gradient descent using Rprop, however, succeeds to segment practically all visible vessels, see Figure 6. Observe that the quality and accuracy of the segmentation have not been verified and is out of scope of this paper. The point of this experimental segmentation was instead to highlight the advantages of Rprop in contrast to the ordinary gradient descent.

6 Conclusions and future work

Image segmentation using the level set method involves optimization in contour space. In this context, the working horse of optimization methods is the gradient descent method. We have discussed the weaknesses of this method and proposed using Rprop, a modified version of gradient descent based on resilient propagation, commonly used in the machine learning community. In addition, we have shown examples on how the solution is improved by Rprop, which adapts its individual update values to the behavior of the cost surface. Using Rprop, the optimization gets less sensitive to local optima and the convergence rate is improved. In contrast to much of the previous work, we have improved the solution by changing the method of solving the optimization problem rather than modifying the energy functional.

Future work includes further study of the general optimization problem of image segmentation and verification of the segmentation quality in real applications. The issue of why the reinitializations disturb the adaptation of the step sizes also has to be studied further.

References

1. Charpiat, G., Keriven, R., Pons, J.P., Faugeras, O.: Designing spatially coherent minimizing flows for variational problems based on active contours. Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on **2** (Oct. 2005) 1403–1408 Vol. 2
2. Sundaramoorthi, G., Yezzi, A., Mennucci, A.: Sobolev active contours. International Journal of Computer Vision **73**(3) (2007) 345–366
3. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The rprop algorithm. In: In Proceedings of the IEEE International Conference on Neural Networks. (1993) 586–591
4. Riedmiller, M., Braun, H.: Rprop – description and implementation details. Technical report, Universitat Karlsruhe (1994)
5. Nocedal, J., Wright, S.J.: Numerical Optimization. 2nd edn. Springer (2006)
6. Schiffmann, W., Joost, M., Werner, R.: Comparison of optimized backpropagation algorithms. In: Proc. of ESANN'93, Brussels. (1993) 97–104
7. Kimmel, R.: Fast edge integration. In: Geometric Level Set Methods in Imaging, Vision and Graphics. Springer Verlag (2003)
8. Osher, S., Sethian, J.A.: Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. Journal of Computational Physics **79** (1988) 12–49
9. Osher, S., Fedkiw, R.: Level Set and Dynamic Implicit Surfaces. Springer-Verlag New York Inc. (2003)
10. Peng, D., Merriman, B., Osher, S., Zhao, H.K., Kang, M.: A pde-based fast local level set method. Journal of Computational Physics **155**(2) (1999) 410–438
11. Sethian, J.: A fast marching level set method for monotonically advancing fronts. In: Proceedings of the National Academy of Science. Volume 93. (1996) 1591–1595
12. Zhao, H.K.: A fast sweeping method for eikonal equations. Mathematics of Computation (74) (2005) 603–627
13. Läthén, G., Jonasson, J., Borga, M.: Phase based level set segmentation of blood vessels. In: Proceedings of 19th International Conference on Pattern Recognition, Tampa, FL, USA, IAPR (December 2008)
14. Granlund, G.H., Knutsson, H.: Signal Processing for Computer Vision. Kluwer Academic Publishers, Netherlands (1995)
15. Staal, J., Abramoff, M., Niemeijer, M., Viergever, M., van Ginneken, B.: Ridge based vessel segmentation in color images of the retina. IEEE Transactions on Medical Imaging **23**(4) (2004) 501–509