A Binary Level Set Model and some Applications to Mumford-Shah Image Segmentation

Johan Lie*, Marius Lysaker† and Xue-Cheng Tai ‡

***Abstract*— In this work we propose a variant of a PDE based level set method. Traditionally interfaces are represented by the zero level set of continuous level set functions. We instead let the interfaces be represented by discontinuities of piecewise constant level set functions. Each level set function can at convergence only take two values, i.e. it can only be 1 or -1. Some of the properties of the standard level set function are preserved in the proposed method, while others are not. Using this new level set method for interface problems, we need to minimize a smooth convex functional under a quadratic constraint. The level set functions are discontinuous at convergence, but the minimization functional is smooth and locally convex. We show numerical results using the method for segmentation of digital images.**

***Index Terms*— image segmentation, image processing, PDE, variational, level set, piecewise constant level set functions**

I. Introduction

THE level set method proposed by Osher and Sethian [1] is a versatile tool for tracing interfaces separating a domain Ω into subdomains. Interfaces are treated as the zero level set of some functions. Moving the interfaces can be done by evolving the level set functions instead of moving the interfaces directly. The level set idea is now used on a broad spectrum of problems, including image analysis, reservoir simulation, inverse problems, computer vision and optimal shape design [2]–[5]. For a recent survey on the level set methods see [6]–[9].

In this work, we propose a variant of the level set method. The primary concern for this approach is to remove the connection between the level set functions and the signed distance function and thus remove some of the computational difficulties associated with the calculation of Eikonal equations, see §II. Another motivation is to avoid the non-differentiability associated with the Heaviside and Delta functions used in some of the level set formulations [2], [10]. The third concern of this approach is to develop fast algorithms for level set methods. Due to the fact that the functional and the constraints for this approach is rather smooth, it is possible to apply Newton types of iterations to construct fast algorithms for the proposed model. The level set model proposed here bears some of the essential nature of the fast level set method of Song and Chan [11]. Our approach can be used for different applications, but in this paper we restrict ourselves to segmentation of digital images. For a given digital image $u_0 : \Omega \to \mathbb{R}$, the aim is to separate Ω into a set of subdomains Ω_i such that $\Omega = \cup_{i=1}^{n} \Omega_i$ and u_0 is nearly a constant in each Ω_i.

One general image segmentation model was proposed by Mumford and Shah in [12]. Numerical approximations are thoroughly treated in [13]. Using this model, the image u_0 is decomposed into $\Omega = \cup_i \Omega_i \cup \Gamma$, where Γ is a curve separating the different domains. Inside each Ω_i, u_0 is approximated by a smooth function. The optimal partition of Ω is found by minimizing the Mumford-Shah functional (8). This is explained in §II. Following the Mumford-Shah formulation for image segmentation, Chan and Vese [2], [10] solved the minimization problem by using level set functions. The interface Γ is traced by the level set functions. Motivated by the Chan-Vese approach, we will in this article solve the segmentation problem in a different way, i.e. by introducing a piecewise constant level set function. Instead of using the zero level of a function to represent the interface between subdomains, we let the interface be represented implicitly by the discontinuities of a level set function. A two phase segmentation is accomplished by requiring the level set function to take the value 1 in one of the regions and -1 in the other region by enforcing ϕ to satisfy $\phi^2 = 1$. In order to divide the domain into several subdomains, we use a set of functions ϕ_i satisfying $\phi_i^2 = 1$. By using N level set functions, we can identify 2^N phases. See also a recent work [14], where we have developed a technique which only needs one discontinuous level set function for representing multiple phases.

The rest of this article is structured as follows. In §II we give a brief review of the traditional level set method. Our new level set model is formulated in §III. In §IV, we apply this model for digital image segmentation. The segmentation problem is formulated as a minimization problem with a smooth cost functional under a quadratic constraint. The minimization functional is essentially the Mumford-Shah functional associated with the new level set model. We propose two algorithms to solve the segmentation problem. We conclude the paper with some numerical examples, using both synthetic and real images, in §V.

II. Standard Level Set Methods

The main idea behind the level set formulation is to represent an interface $\Gamma(t)$ bounding a possibly multiply connected region in $\mathbb{R}^n$ by a Lipschitz continuous function ϕ, having the

This work is supported by the Norwegian Research Council, Center of Mathematcis for Applications (CMA) and Center for Integrated Petroleum Research (CIPR).

*Department of Mathematics, University of Bergen, Johannes Brunsgate 12, N-5008, Norway, e-mail: johanl@mi.uib.no, web page: http://www.mi.uib.no/BBG

†Simula Research Laboratory, Martin Linges v 17, Fornebu P.O.Box 134, 1325 Lysaker, Norway, e-mail: mariul@simula.no

‡Department of Mathematics, University of Bergen, Johannes Brunsgate 12, N-5008, Norway, e-mail: tai@mi.uib.no, web page: http://www.mi.uib.no/˜tai.

following properties

$$\begin{cases} \phi(x,t) > 0, & \text{if x is inside } \Gamma(t), \\ \phi(x,t) = 0, & \text{if x is at } \Gamma(t), \\ \phi(x,t) < 0, & \text{if x is outside } \Gamma(t). \end{cases} \quad (1)$$

Some regularity must be imposed on ϕ to prevent the level set function of being too steep or flat near the interface. This is normally done by requiring ϕ to be a signed distance function to the interface

$$\begin{cases} \phi(x,t) = d(\Gamma(t),x), & \text{if x is inside } \Gamma(t), \\ \phi(x,t) = 0, & \text{if x is at } \Gamma(t), \\ \phi(x,t) = -d(\Gamma(t),x), & \text{if x is outside } \Gamma(t), \end{cases} \quad (2)$$

where $d(\Gamma(t),x)$ denotes Euclidian distance between x and $\Gamma(t)$. Having defined the level set function ϕ as in (2), there is a one to one correspondence between the curve Γ and the function ϕ. The distance function ϕ obeys the Eikonal equation

$$|\nabla\phi| = 1. \quad (3)$$

The solution of (3) is not unique in the distributional sense. Finding the unique vanishing viscosity solution of (3) is usually done by solving the following initial value problem to steady state

$$\begin{aligned} \phi_t + sgn(\tilde{\phi})(|\nabla\phi| - 1) &= 0 & (4) \\ \phi(x,0) &= \tilde{\phi}(x). & (5) \end{aligned}$$

In the above, $\tilde{\phi}$ may not be a distance function. When the steady state of equation (4) is reached, it will be a distance function having the same zero curve as $\tilde{\phi}$. This is commonly known as the reinitialization procedure. For numerical computations this procedure is crucial, and many numerical finite difference schemes exists. See [3], [6], [7], [15] for some details. The interface $\Gamma(t)$ is implicitly moved according to the nonlinear PDE

$$\frac{\partial\phi}{\partial t} + \boldsymbol{v}\cdot\nabla\phi = 0, \quad (6)$$

where $\boldsymbol{v}$ is a given velocity field. This vector field can depend on geometry, position, time and internal or external physics. Usually only the velocity normal to the interface v_N is needed, and ϕ is then moved according to the modified equation

$$\frac{\partial\phi}{\partial t} + v_N|\nabla\phi| = 0. \quad (7)$$

Level Set Methods and Image Segmentation

The active contour (snake) model evolves a curve $\Gamma(t)$ in order to detect objects in an image u_0 [16]. The curve is moved from an initial position $\Gamma(0)$ in the direction normal to the curve, subject to constraints in the image. An edge detector $g(\nabla u_0)$ determines when $\Gamma(t)$ is situated at the boundary of an object. One limitation of the snake model is that the curve is represented explicitly, thus topological changes like merging and breaking of the curve may be hard to handle. To address this problem, a level set formulation of the active contour model was introduced in [17]. Later, Chan-Vese introduced a level set model for active contour segmentation, with the very important property that the stopping criteria is independent of ∇u_0 [2]. This means that boundaries not defined by gradients can be detected. Instead, the evolvement of the curve is based on the general Mumford-Shah formulation of image segmentation, by minimizing the functional

$$F^{MS}(u,\Gamma) = \sum_i \int_{\Omega_i} |c_i - u_0|^2 dx + \mu|\Gamma| + \nu\int_{\Omega\backslash\Gamma} |\nabla u|^2 dx. \quad (8)$$

In the above, $|\Gamma|$ is the length of Γ. A minimizer of this functional is smooth in $\Omega\backslash\Gamma$. The piecewise constant Mumford-Shah formulation of image segmentation is to find a partition of Ω such that u in Ω_i equals a constant c_i, and $\Omega = \cup_i^n \Omega_i \cup \Gamma$. The two last terms in (8) are regularizers measuring curve-length of the curves bounding the phases, and smoothness of u in $\Omega\backslash\Gamma$. Based on (8), Chan and Vese [2] proposed the following minimization problem for a two phase segmentation

$$\begin{aligned} \min_{c_1,\,c_2,\,\phi} \Big\{ &\int_\Omega |u_0 - c_1|^2 H(\phi)dx + \int_\Omega |u_0 - c_2|^2(1-H(\phi))dx \\ &+\nu\int_\Omega H(\phi)dx + \mu\int_\Omega \delta(\phi)|\nabla\phi|dx\Big\}. \end{aligned} \quad (9)$$

Here ϕ is the level set function satisfying (1) and $H(\phi)$ is the Heaviside function: $H(\phi) = 1$ if $\phi \geq 0$ and $H(\phi) = 0$ if $\phi < 0$. Finding a minimum of (9) is done by introducing an artificial time variable, and moving ϕ in the steepest descent direction to steady state

$$\begin{aligned} \phi_t &= \delta_\epsilon(\phi)\Big(-(u_0 - c_1)^2 + (u_0 - c_2)^2 \\ &\quad - \nu + \mu\nabla\cdot\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big)\Big), & (10) \\ \phi(\boldsymbol{x},0) &= \phi_0(\boldsymbol{x}). & (11) \end{aligned}$$

Here δ_ϵ is a globally positive approximation to the δ function, see [2]. The recovered image is a piecewise constant approximation to u_0.

If we do not impose any other conditions on the level set functions ϕ, then the minimizer of F^{MS} with respect to ϕ may not be unique. Thus, we require the level set function ϕ to be a distance function. This means that the level set function is a steady state of both (10) and (4). In practice, this means we need to reinitialize the level set function.

This level set framework was later generalized to multiple phase segmentation using multiple level set functions [10]. A four phase segmentation can be accomplished by minimizing the functional

$$
\begin{aligned}
F(\boldsymbol{c},\phi_1,\phi_2) = & \int_\Omega |c_1-u_0|^2 H(\phi_1)H(\phi_2)dx \\
& + \int_\Omega |c_2-u_0|^2 H(\phi_1)(1-H(\phi_2))dx \\
& + \int_\Omega |c_3-u_0|^2 (1-H(\phi_1))H(\phi_2)dx \qquad (12) \\
& + \int_\Omega |c_4-u_0|^2 (1-H(\phi_1))(1-H(\phi_2))dx \\
& + \nu \int_\Omega H(\phi_1) + \nu \int_\Omega H(\phi_2)dx \\
& + \mu \int_\Omega \delta(\phi_1)|\nabla\phi_1|dx + \mu \int_\Omega \delta(\phi_2)|\nabla\phi_2|dx.
\end{aligned}
$$

Having determined $\boldsymbol{c} = \{c_i\}_{i=1}^4$, ϕ_1 and ϕ_2 by the minimization of (12), four different regions can be identified by the sign of the two level set functions such that

$$
u(\boldsymbol{x}) = \begin{cases} c_1, & \text{if } \phi_1(\boldsymbol{x}) > 0 \ , \ \phi_2(\boldsymbol{x}) > 0, \\ c_2, & \text{if } \phi_1(\boldsymbol{x}) > 0 \ , \ \phi_2(\boldsymbol{x}) < 0, \\ c_3, & \text{if } \phi_1(\boldsymbol{x}) < 0 \ , \ \phi_2(\boldsymbol{x}) > 0, \\ c_4, & \text{if } \phi_1(\boldsymbol{x}) < 0 \ , \ \phi_2(\boldsymbol{x}) < 0. \end{cases} \qquad (13)
$$

By utilizing the Heaviside function, the recovered cartoon image u consisting of four phases can be written as

$$
\begin{aligned}
u = & c_1 H(\phi_1)H(\phi_2) + c_2 H(\phi_1)(1-H(\phi_2)) \\
& + c_3(1-H(\phi_1))H(\phi_2) + c_4(1-H(\phi_1))(1-H(\phi_2)).
\end{aligned} \qquad (14)
$$

Increasing the number of phases is done by increasing the number of level set functions. N level set functions are needed to represent up to 2^N phases.

In our paper we solve the piecewise constant Mumford-Shah segmentation using a slightly different level set approach. We separate the connection between the level set function and the distance function. This means that we get rid of the reinitialization procedure. In our approach, we impose a quadratic constraint on the level set functions, i.e. $\phi_i^2 = 1$. Our approach is truely variational, i.e. the equations we need to solve are coming from the Euler-Lagrange equations for some smooth convex functions. The problem of non-differentiability associated with the Heaviside and Delta functions is avoided.

III. Our Level Set Approach

To introduce our level set idea, let us first assume that the interface is enclosing $\Omega_1 \subset \Omega$. By standard level set methods the interior of Ω_1 is represented by points $\boldsymbol{x} : \phi(\boldsymbol{x}) > 0$, and the exterior of Ω_1 is represented by points $\boldsymbol{x} : \phi(\boldsymbol{x}) < 0$, as in (1). We instead use a discontinuous level set function ϕ, with $\phi(\boldsymbol{x}) = 1$ if $\boldsymbol{x}$ is an interior point of Ω_1 and $\phi(\boldsymbol{x}) = -1$ if $\boldsymbol{x}$ is an exterior point of Ω_1

$$
\phi(\boldsymbol{x}) = \begin{cases} & \text{if } \boldsymbol{x} \ \text{int}(\Omega_1) \ , \\ -1 & \text{if } \boldsymbol{x} \ \text{ext}(\Omega_1) \ . \end{cases} \qquad (15)
$$

Γ is implicitly defined as the discontinuity of ϕ. This representation can be used for different applications where subdomains need to be identified. In order to use this idea for image segmentation, we use (15) to represent piecewise constant functions. Let us assume that u_0 is an image consisting of two distinct regions Ω_1 and Ω_2, and we want to construct a piecewise continuous approximation u to u_0. Let $u(\boldsymbol{x}) = c_1$ in Ω_1, and $u(\boldsymbol{x}) = c_2$ in Ω_2. If $\phi(\boldsymbol{x}) = 1$ in Ω_1, and $\phi(\boldsymbol{x}) = -1$ in Ω_2, u can be written as the sum

$$
u = \frac{c_1}{2}(\phi+1) - \frac{c_2}{2}(\phi-1). \qquad (16)
$$

The formula (16) can be generalized to represent functions with more than two constant values by using multiple level set functions following the essential ideas of the level set formulation used in [4], [10]. A function having four constant values can be associated with two level set functions $\{\phi_i\}_{i=1}^2$ satisfying $\phi_i^2 = 1$. More precisely, a function given as

$$
\begin{aligned}
u = & \tfrac{c_1}{4}(\phi_1+1)(\phi_2+1) - \tfrac{c_2}{4}(\phi_1+1)(\phi_2-1) \\
& -\tfrac{c_3}{4}(\phi_1-1)(\phi_2+1) + \tfrac{c_4}{4}(\phi_1-1)(\phi_2-1).
\end{aligned} \qquad (17)
$$

is a piecewise constant function of the form

$$
u(\boldsymbol{x}) = \begin{cases} c_1, & \text{if } \phi_1(\boldsymbol{x}) = 1, & \phi_2(\boldsymbol{x}) = 1, \\ c_2, & \text{if } \phi_1(\boldsymbol{x}) = 1, & \phi_2(\boldsymbol{x}) = -1, \\ c_3, & \text{if } \phi_1(\boldsymbol{x}) = -1, & \phi_2(\boldsymbol{x}) = 1, \\ c_4, & \text{if } \phi_1(\boldsymbol{x}) = -1, & \phi_2(\boldsymbol{x}) = -1. \end{cases}
$$

Introducing basis functions ψ_i as in the following

$$
\begin{aligned}
u = & c_1 \underbrace{\frac{1}{4}(\phi_1+1)(\phi_2+1)}_{\psi_1} \\
& + c_2 \underbrace{(-1)\frac{1}{4}(\phi_1+1)(\phi_2-1)}_{\psi_2} + \cdots ,
\end{aligned} \qquad (18)
$$

we see that u can be written as

$$
u = \sum_{i=1}^4 c_i \psi_i. \qquad (19)
$$

For more general cases, we could use N level set functions for to produce 2^N number of phases. To simplify notation, we define the vectors $\boldsymbol{\phi} = \{\phi_1,\phi_2,\ldots,\phi_N\}$ and $\boldsymbol{c} = \{c_1,c_2,\ldots,c_{2^N}\}$. For $i = 1,2,\ldots,2^N$, let $(b_1^{i-1}, b_2^{i-1}, \ldots, b_N^{i-1})$ be the binary representation of $i-1$, where $b_j^{i-1} = 0 \vee 1$. Furthermore, set

$$
s(i) = \sum_{j=1}^N b_j^{i-1}, \qquad (20)
$$

and write ψ_i as the product

$$
\psi_i = \frac{(-1)^{s(i)}}{2^N} \prod_{j=1}^N (\phi_j + 1 - 2b_j^{i-1}). \qquad (21)
$$

Then a function u having 2^N constant values can be written as the weighted sum

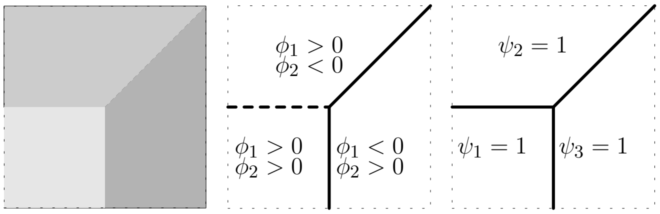
$$
u = \sum_{i=1}^{2^N} c_i \psi_i. \qquad (22)
$$

If the level set functions ϕ_i satisfy $\phi_i^2 = 1$ and ψ_i are defined as in (19) or (21), then $\text{supp}(\psi_i) = \Omega_i$, $\psi_i = 1$ in Ω_i, and $\text{supp}(\psi_i) \cap \text{supp}(\psi_j) = \emptyset$ when $j \neq i$. This ensures non-overlapping phases, and in addition $\bigcup_i \text{supp}(\psi_i) = \Omega$, which prevents vacuums.

If the level set functions satisfy $\phi_i^2 = 1$, then we can use the basis functions ψ_i to calculate the length of the boundary of Ω_i and the area inside Ω_i, i.e.

$$|\partial\Omega_i| = \int\limits_\Omega |\nabla\psi_i| dx, \qquad \text{and} \qquad |\Omega_i| = \int\limits_\Omega \psi_i dx. \quad (23)$$

In fact, measuring the length of boundaries by using this representation is more accurate than using the $\int\limits_\Omega \delta(\phi_i)|\nabla\phi_i|dx$ which is done in [2], [10]. This is due to the fact that their regularizer does not treat every edge equally, by measuring some edges once and other edges two times, i.e. some edges are treated as more important than other edges, as pointed out by Chan and Vese [10]. Our method on the contrary counts every edge two times, and thus all the edges are treated equally. A simple example illustrating the difference between the two regularizers is shown in Fig. 1. Uisng $\int\limits_\Omega \delta(\phi_i)|\nabla\phi_i|dx$ as the regularizer, the length of dashed line in Fig. 1(b) is counted once while the thick line is counted twice. Using our approach, the length for all the lines are counted twice.



(a) (b) (c)

Fig. 1. (a) A simple image consisting of three phases. (b) If $\int\limits_\Omega \delta(\phi_i)|\nabla\phi_i|dx$ is used as a regularizer, the different edges are not treated in a similar fashion.The edge with the (thick) dashed line is measured once and the other (thick) edges are measured twice. Moreover, in general it is impossible to determine how each edge is supposed to be measured. (c) Using our representation, all edges are measured two times.

IV. Minimization Problem

We have now introduced a way to represent a piecewise constant function u by using the binary level set functions. Based on this we propose to minimize the following functional to find a segmentation for a given image u_0

$$F(\boldsymbol{\phi}, \boldsymbol{c}) = \frac{1}{2}\int\limits_\Omega |\, u - u_0 \,|^2 dx + \beta \sum_{i=1}^{2^N} \int\limits_\Omega |\nabla\psi_i| dx. \quad (24)$$

In the above, β is a nonnegative parameter controlling the regularizing, u is a piecewise constant function depending on $\boldsymbol{\phi}$ and $\boldsymbol{c}$, as in (22). The first term of (24) is a least square functional, measuring how well the piecewise constant image u approximates u_0. The second term is a regularizer measuring the Total Variation (TV) of the basis functions. Essentially this regularizer measures length of the edges in the image u_0. Considering the constraints imposed on the level set functions, we find that the segmentation problem is the following constrained minimization problem

$$\boxed{\min_{\boldsymbol{\phi}, \boldsymbol{c}} F(\boldsymbol{\phi}, \boldsymbol{c}), \quad \text{subject to} \quad \phi_i^2 = 1, \forall\, i.} \quad (25)$$

Recall that $\boldsymbol{\phi}$ is a vector having N elements ϕ_i. For notational simplicity, we introduce a vector $\boldsymbol{K}(\boldsymbol{\phi})$ of the same dimension as $\boldsymbol{\phi}$ with $K_i(\boldsymbol{\phi}) = \phi_i^2 - 1$. It is easy to see that

$$\phi_i^2 = 1, \forall i \Leftrightarrow \boldsymbol{K}(\boldsymbol{\phi}) = \boldsymbol{0}. \quad (26)$$

We use two related methods to solve the minimization problem (25), a projection Lagrangian approach, and an augmented Lagrangian approach. This leads to two related iterative algorithms for image segmentation. The projection Lagrangian method is first described.

A. A Projection Lagrangian Algorithm

The Lagrangian functional involves both F and the constraint $\boldsymbol{K}$

$$L(\boldsymbol{\phi}, \boldsymbol{c}, \boldsymbol{\lambda}) = F(\boldsymbol{\phi}, \boldsymbol{c}) + \sum_{i=1}^{N} \int\limits_\Omega \lambda_i K_i dx. \quad (27)$$

Here $\boldsymbol{\lambda} = \{\lambda_i\}_{i=1}^N$ is a vector of functions of the same dimension as $\boldsymbol{\phi}$, called the Lagrange multipliers. Note that when the constraint is fulfilled, the Lagrangian term vanishes, and $L = F$. We search for a saddle point of (27), which in turn will give a minimizer of (25). At a saddle point of L we must have

$$\frac{\partial L}{\partial \phi_i} = 0, \ \frac{\partial L}{\partial c_i} = 0, \ \frac{\partial L}{\partial \lambda_i} = 0, \ \forall i. \quad (28)$$

The saddle point is sought by minimizing L with respect to $\boldsymbol{\phi}$ and $\boldsymbol{c}$, and maximizing with respect to $\boldsymbol{\lambda}$. By maximizing $\boldsymbol{\lambda}$ the constraint must be fulfilled at convergence, otherwise the Lagrangian term of (27) will not vanish. From the definition of L, we see that

$$\begin{aligned} \frac{\partial L}{\partial \phi_i} &= \frac{\partial F}{\partial \phi_i} + \sum_{j=1}^{N} \lambda_j \frac{\partial K_j}{\partial \phi_i} \\ &= (u - u_0) \sum_{j=1}^{2^N} c_j \frac{\partial \psi_j}{\partial \phi_i} - \beta \sum_{j=1}^{2^N} \nabla \cdot \Big(\frac{\nabla \psi_j}{|\nabla \psi_j|} \Big) \frac{\partial \psi_j}{\partial \phi_i} + 2\lambda_i \phi_i. \end{aligned} \quad (29)$$

Using formulas (16), (17) and (21), it is easy to get $\partial\psi_j / \partial\phi_i$.

Since $u = \sum_i c_i \psi_i$, and only the first term of F depends on u, the derivative with respect to c_i becomes

$$\frac{\partial L}{\partial c_i} = \int\limits_\Omega \frac{\partial L}{\partial u} \frac{\partial u}{\partial c_i} dx = \int\limits_\Omega \frac{\partial F}{\partial u} \frac{\partial u}{\partial c_i} dx = \int\limits_\Omega (u - u_0)\, \psi_i \, dx. \quad (30)$$

The derivative of L with respect to λ_i essentially recovers the constraint:

$$\frac{\partial L}{\partial \lambda_i} = K_i = \phi_i^2 - 1. \quad (31)$$

All the derivatives (29), (30), (31) must equal zero at a saddle point of L. To find the saddle point, we shall use an iterative algorithm. From initial guesses $\boldsymbol{\phi}^0, \boldsymbol{c}^0, \boldsymbol{\lambda}^0$, we iterate towards better approximations $\boldsymbol{\phi}^k, \boldsymbol{c}^k, \boldsymbol{\lambda}^k$. Since we want the three derivatives (29), (30), (31) to equal zero, we increase k until none of $\boldsymbol{\phi}$, $\boldsymbol{c}$, $\boldsymbol{\lambda}$ changes anymore. Then we have arrived at a saddle point. Using this approach, we need to choose three different schemes to get $\boldsymbol{\phi}^k, \boldsymbol{c}^k, \boldsymbol{\lambda}^k$ from $\boldsymbol{\phi}^{k-1}, \boldsymbol{c}^{k-1}, \boldsymbol{\lambda}^{k-1}$.

First consider the minimization w.r.t. $\boldsymbol{\phi}$, which is done by introducing an artificial time variable and solving the PDE

$$\boldsymbol{\phi}_t = -\frac{\partial L}{\partial \boldsymbol{\phi}}, \tag{32}$$

to steady state. At steady state, $\boldsymbol{\phi}_t = 0$, which means $\frac{\partial L}{\partial \phi_i} = 0 \;\forall\, i$. This is exactly what is needed for a saddle point of L. We discretize the time derivative using a forward Euler scheme

$$\boldsymbol{\phi_t} \approx \frac{\boldsymbol{\phi}^{new} - \boldsymbol{\phi}^{old}}{\Delta t}. \tag{33}$$

Here Δt is a small positive time step. Combining (32) with (33), and rearranging the terms gives an updating scheme for ϕ_i

$$\phi_i^{new} = \phi_i^{old} - \Delta t \frac{\partial L}{\partial \phi_i}(\boldsymbol{\phi}^{old}, \boldsymbol{c}^{k-1}\, \boldsymbol{\lambda}^{k-1}). \tag{34}$$

Observe that ϕ_i is moved in the steepest descent direction, so this is essentially the gradient method. We use a fixed time step Δt, determined by a trial and error approach. The curvature term in $\frac{\partial L}{\partial \phi}$ is the most restrictive term to the size of Δt. After a fixed number of iterations we let $\boldsymbol{\phi}^k = \boldsymbol{\phi}^{new}$. If an infinite number of iterations were done, i.e. $t \to \infty$, we would end up with the exact minimizer of L w.r.t. $\boldsymbol{\phi}$ at $\boldsymbol{c}^{k-1}, \boldsymbol{\lambda}^{k-1}$.

Secondly, we consider the minimization of L w.r.t. $\boldsymbol{c}$, which is done by using (22). u is a linear combination of the basis functions, thus L is quadratic in $\boldsymbol{c}$. This means the minimization w.r.t. $\boldsymbol{c}$ can be done by solving the $2^N \times 2^N$ linear system $A\boldsymbol{c} = \boldsymbol{b}$, where $A_{ij} = (\psi_i, \psi_j)_{L_2(\Omega)}$, and $\boldsymbol{b} = (u_0, \psi_i)_{L_2(\Omega)}$.

$$\sum_{i=1}^{2^N} (\psi_i^k, \psi_j^k)_{L_2(\Omega)}\, c_i^k = (u_0, \psi_i^k)_{L_2(\Omega)}, \quad i = 1, 2 \ldots 2^N. \tag{35}$$

Last, an updating scheme for $\boldsymbol{\lambda}$ is constructed by combining (28) with (29) and (31). A saddle point of L must satisfy

$$0 = \frac{\partial L}{\partial \phi_i} = \frac{\partial F}{\partial \phi_i} + 2\lambda_i \phi_i. \tag{36}$$

By multiplying this equation with ϕ_i, and noting that at a saddle point of L the constraint gives $\phi_i^2 = 1$, we can set this into (36) to get

$$\lambda_i = -\frac{1}{2}\phi_i \frac{\partial F}{\partial \phi_i}. \tag{37}$$

This is used as an updating scheme for $\boldsymbol{\lambda}$:

$$\lambda_i^k = -\frac{1}{2}\phi_i^k \frac{\partial F}{\partial \phi_i}(\boldsymbol{\phi}^k, \boldsymbol{c}^k). \tag{38}$$

Now the three updating formulas (34), (35) and (38) are combined to construct an algorithm using the Lagrangian approach. This scheme is essentially a projection Lagrangian algorithm.

Algorithm 1 (A Projection Lagrangian Method.)

```
Initialize $\boldsymbol{c}^0$, $\boldsymbol{\phi}^0$, $\boldsymbol{\lambda}^0$.
  1. Update $\boldsymbol{\phi}^k$ by (34), to approximately solve
       $L(\boldsymbol{c}^{k-1}, \boldsymbol{\phi}^k, \boldsymbol{\lambda}^{k-1}) = \min_{\phi} L(\boldsymbol{c}^{k-1}, \boldsymbol{\phi}, \boldsymbol{\lambda}^{k-1})$.
  2. Construct $u(\boldsymbol{c}^{k-1}, \boldsymbol{\phi}^k)$ by
       $u = \sum_{i=1}^{2^N} c_i^{k-1} \psi_i^k$.
  3. Update $\boldsymbol{c}^k$ by (35), to solve
       $L(\boldsymbol{c}^k, \boldsymbol{\phi}^k, \boldsymbol{\lambda}^{k-1}) = \min_{\boldsymbol{c}} L(\boldsymbol{c}, \boldsymbol{\phi}^k, \boldsymbol{\lambda}^{k-1})$.
  4. Update the multiplier by
       $\lambda_i^k = -\frac{1}{2}\phi_i^k \frac{\partial F}{\partial \phi_i}(\boldsymbol{\phi}^k, \boldsymbol{c}^k), \quad \forall i = 1, 2, \ldots, 2^N$.
  5. Test convergence.
  If necessary, $k \leftarrow k+1$, repeat.
```

***Remark** 1:* The minimization w.r.t. $\boldsymbol{c}$ done in step 3 should not be done too early in the process, e.g. not before $|\phi_i| \approx 1$, otherwise the matrix inversion in (35) becomes ill-conditioned. Minor pertubations of the level set functions will result in large errors of the c_i-values. If $|\phi_i|$ is far from 1, then ψ_i is far from orthogonal to ψ_j, and the inner product (ψ_i, ψ_j) in (35) will give contributions at points where it should not. This means that the matrix inversion in (35) does not give a good approximation to $\boldsymbol{c}$ unless $|\phi_i| \approx 1 \;\forall i$.

***Remark** 2:* The time step used in the gradient iteration in step 1 is controlled by the size of $\nabla \cdot \left(\frac{\nabla \psi_i}{|\nabla \psi_i|}\right)$. This term controls the curvature of ψ_i, i.e. essentially the second order derivatives of ψ_i. If the curvature becomes big, (this is the case when ψ becomes discontinuous), it might violate the CFL stability condition of the numerical scheme, unless the value of β is small. Thus larger β values require smaller time steps and vice versa.

***Remark** 3:* The number n of gradient iterations performed in step 1 is usually set to a small number $n \approx 10$. This means that a gradient iteration is performed n times before the other steps in the algorithm are done. The minimization w.r.t. ϕ is therefore not exact, but increasing $n \to \infty$ would hopefully give an exact minimizer. We have observed that using ten gradient iterations usually gives a sufficiently good approximation to the exact minimizer before the other steps of the algorithm are performed.

B. Augmented Lagrangian Algorithm

We can also solve the minimization problem by the augmented Lagrangian method. This is a combination of the multiplier method and the penalization method. Define the augmented Lagrangian functional as

$$L_\mu(\boldsymbol{\phi}, \boldsymbol{c}, \boldsymbol{\lambda}) = F(\boldsymbol{\phi}, \boldsymbol{c}) + \sum_{i=1}^{N} \int_\Omega \lambda_i K_i dx + \frac{1}{2}\mu \sum_{i=1}^{N} \int_\Omega K_i^2 dx. \tag{39}$$

Here $\mu > 0$ is a penalization parameter, and the last term of (39) is called a penalization term. Similarly as in the Lagrangian approach, to minimize $F(\boldsymbol{\phi}, \boldsymbol{c})$, we need to find a saddle point of (39). Thus we need updating schemes for $\boldsymbol{\phi}$, $\boldsymbol{c}$ and $\boldsymbol{\lambda}$. Both $\boldsymbol{\phi}$ and $\boldsymbol{c}$ are updated using the same techniques as in the Lagrangian approach. Hence we only need a new scheme for updating $\boldsymbol{\lambda}$, in addition to a scheme for updating the penalization parameter μ. These two schemes are interconnected.

The original idea of a penalty method is to iteratively force the constraint to be fulfilled by increasing μ to ∞. For the augmented Lagrangian method, due to the Lagrangian multipliers, the constraints are satisfied even if we use a fixed penalization parameter μ. In practice, better convergence can be obtained if we increase the value of the penalization parameter. Let $\boldsymbol{\lambda}^k$ denotes $\boldsymbol{\lambda}$ at the k^{th} iteration. Following the approach in [18], [19], we use the following updating scheme

$$\boldsymbol{\lambda}^k = \boldsymbol{\lambda}^{k-1} + \mu \boldsymbol{K}(\boldsymbol{\phi}^k). \tag{40}$$

Having determined $\boldsymbol{\lambda}^{k-1}$, we minimize L_μ w.r.t. $\boldsymbol{\phi}$ by the gradient method updating scheme

$$\phi_i^{new} = \phi_i^{old} - \Delta t \frac{\partial L_\mu}{\partial \phi_i}(\boldsymbol{\phi}^{old}, \boldsymbol{c}^{k-1}, \lambda_i^{k-1}), \tag{41}$$

where

$$\begin{aligned} \frac{\partial L_\mu}{\partial \phi_i} &= (u - u_0) \sum_{j=1}^{2^N} c_j \frac{\partial \psi_j}{\partial \phi_i} && (42) \\ &- \beta \sum_{j=1}^{2^N} \nabla \cdot \Big(\frac{\nabla \psi_j}{|\nabla \psi_j|} \Big) \frac{\partial \psi_j}{\partial \phi_i} && (43) \\ &+ 2\lambda_i \phi_i + 2\mu(\phi_i^2 - 1)\phi_i. && (44) \end{aligned}$$

Like in the first algorithm, after a few iterations we set $\boldsymbol{\phi}^k = \boldsymbol{\phi}^{new}$. The constraints K_i are independent of the constant values c_i and thus the updating for the c_i values will still be the same.

Algorithm 2 (An augmented Lagrangian Method.)

```
Initialize $\boldsymbol{c}^0$, $\boldsymbol{\phi}^0$, $\boldsymbol{\lambda}^0, \mu$.
  1. Update $\boldsymbol{\phi}^k$ by (41), to approximately solve
     $L_\mu(\boldsymbol{c}^{k-1}, \boldsymbol{\phi}^k, \boldsymbol{\lambda}^{k-1}) = \min_{\phi} L_\mu(\boldsymbol{c}^{k-1}, \boldsymbol{\phi}, \boldsymbol{\lambda}^{k-1})$.
  2. Construct $u(\boldsymbol{c}^{k-1}, \boldsymbol{\phi}^k)$ by
     $u = \sum_{i=1}^{2^N} c_i^{k-1} \psi_i^k$.
  3. Update $\boldsymbol{c}^k$ by (35), to solve
     $L_\mu(\boldsymbol{c}^k, \boldsymbol{\phi}^k, \boldsymbol{\lambda}^{k-1}) = \min_{\boldsymbol{c}} L_\mu(\boldsymbol{c}, \boldsymbol{\phi}^k, \boldsymbol{\lambda}^{k-1})$.
  4. Update the multiplier by
     $\boldsymbol{\lambda}^k = \boldsymbol{\lambda}^{k-1} + \mu \boldsymbol{K}(\boldsymbol{\phi}^k)$.
  5. Test convergence.
  If necessary, $k \leftarrow k+1$, iterate again.
```

This algorithm has a linear convergence and its convergence has been analyzed in Kunisch and Tai [20] under a slightly different context. This algorithm has also been used in Chan and Tai [4], [21] for a level set method for elliptic inverse problems.

Remark *1:* In most of our simulations we have set μ to be constant during the processing. This is done to make the simulations as "safe" as possible. Better convergence behavior can be expected if μ is increased during the iterations, but be aware of ill-conditioning if μ is increased too quickly. This is a common approach when using the augmented Lagrangian method. See [18], [19] for details concerning the general algorithm.

Remark *2:* As in the first algorithm, $\boldsymbol{c}$ should not be updated too early in the process, to avoid ill-conditioning when inverting the matrix A. See Remark 1 of Algorithm 1.

Remark *3:* In this algorithm, Δt in the gradient iteration depends on both β and μ. A large β or μ require a small Δt. The constant β can be looked at as a parameter controlling the connectivity or oscillations of the different phases. A bigger β value will suppress oscillations, while a bigger μ makes the level set functions ϕ_i converging to ± 1 quicker. Choosing μ too big will reduce the influence of the fitting term $F(\boldsymbol{\phi}, \boldsymbol{c})$ and thus may increase the iteration number needed to converge to the true solution. For practical problems, it is normally not too difficult to find a approximate range for these two parameters.

V. Numerical Experiments

In this section numerical results are presented. Essentially our model has two parameters, β and μ. When using the projection Lagrangian algorithm, the only parameter is β. We keep the penalty parameter μ constant in the simulations when the augmented Lagrangian method is used. This means the number of iterations needed before convergence is not necessarily optimal, but it makes the problem less ill-conditioned than by increasing μ. For most of the tests we have done, the parameters are chosen in the following range

$$10^{-3} \leq \beta \leq 1 \quad \text{and} \quad 10^{-4} \leq \mu \leq 10^{-1}. \tag{45}$$

These ranges are not strict, but can serve as a guide. In each numerical test, we report the specific values used for β and μ. In all the simulations shown, we use $\boldsymbol{\phi}^0 \equiv 0$. Most of the images are imposed with noise, and we assume the noise is additive

$$u_0 = u + \eta, \tag{46}$$

where η is Gaussian distributed noise. For each of the examples containing noise, we report the Signal to Noise Ratio (SNR):

$$SNR = \frac{\text{Variance of Data}}{\text{Variance of Noise}}. \tag{47}$$

If $SNR \approx 1$, the observation data is very noisy.

Even though the framework developed in this article is applicable for multiple level set functions, we only show numerical results using one or two level set functions. When using two functions, there is a need for an initial approximation of $\boldsymbol{c}$. This is done by the following process: First a median filter is applied to the image, to produce a smoothed temporary image u_{tmp}, e.g. $u_{tmp}(x_{ij})$ is taken to be the mean of a set of neighbor points of $u_0(x_{ij})$. Afterwards, a simple isodata approach is applied on u_{tmp} to find $\boldsymbol{c}^0$, an approximation to the optimal $\boldsymbol{c}$. We refer the reader to [22] for a discussion of the isodata algorithm. When using only one level set function, the initial value for $\boldsymbol{c}$ is not important, the algorithm converges to the same solution even if we start with an initial value far from the true one. This is due to the uniform convex nature of the objective functional in the two subdomain case. All the examples shown are processed using the augmented Lagrangian Algorithm.

A. *One Level Set Function*

Example 1. We start with an example where one level set function is used to detect two different subdomains. We want to test our algorithms on a really challenging image with scattered data, i.e. a satellite image of Europe showing clusters of light. For our algorithms to work, the level set functions must converge to ± 1. Which point the level set function should equal to 1 and which point the level set function should equal to -1 is influenced by the value of the regularization parameter β. With a bigger β, the obtained image is more ”connected”, see Fig. 2. With a smaller β the image is less ”connected”. No matter what kind of value we choose for β, the algorithms are able to get the level set function to converge to ± 1. Three of the computed results are shown in Figs. 2(b),(c) and (d). The recovered images are very similar to the results obtained in [2], [14].

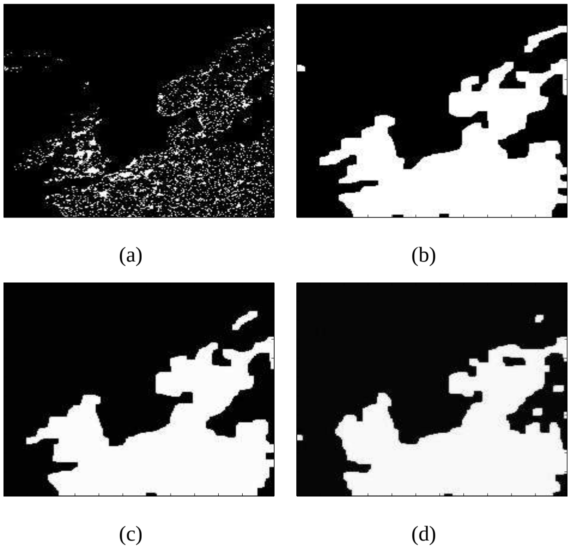


Fig. 2. Segmentation of a scattered data of a satelite image of Europe using different values for β. We have used a fixed $\mu = 10$. Different values of β influence the connectivity of the different parts of the resulting image. (b), (c) and (d) u piecewise constant approximations using $\beta = 1e^{-3}, \beta = 5e^{-3}$ and $\beta = 1e^{-2}$.

In this example, 100 iterations were needed for convergence.

Example 2. In this example, we introduce a technique which can be used for accelerating the convergence of the two algorithms. At convergence, the level set functions should approach ± 1. After a few gradient iterations, the level set functions could already have the correct sign, but it might take many iterations to get ϕ exactly to ± 1. This is a common behavior of the steepest descent method, due to the slow convergence rate. To accelerate the convergence, we can start the algorithms with an initial value for the level set functions and do a few iterations. We then take the sign of the obtained iterative level set functions as an initial value and start the algorithm again. The step size Δt can be chosen by a linesearch. The technique is tested, and we show the results for a specific image in Fig. 3.

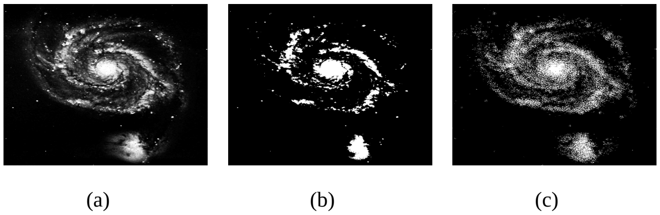


Fig. 3. An image of a galaxy is processed using the modified version of the augmented Lagrangian algorithm with a linesearch. (a) The original image u_0. (b) u, a piecewise constant approximation to u_0 after 10 iterations with $\beta = 1e{-}3$. (c) u, another piecewise constant approximation with $\beta = 2e{-}4$.

As observed in this example, the number of iterations is decreased dramatically when using the modified algorithm. In the case of two phase segmentation, the functional $L(\phi, \boldsymbol{c}, \lambda)$ is globally convex w.r.t. $\boldsymbol{c}$ and ϕ. For multiple phases, $L(\boldsymbol{\phi}, \boldsymbol{c}, \boldsymbol{\lambda})$ is only locally convex w.r.t. $\boldsymbol{c}$ and $\boldsymbol{\phi}$. The above technique can accelerate the convergence. However more careful tests are needed to draw some solid conclusions concerning the class of problems where the speedup-technique is applicable.

We have tested our algorithms for some more images than what are reported here for two-phase problems. We have observed that the accuracy and quality of the recovered images are as good as the one produced by the algorithms of [2], [14]. Even though our level set functions are discontinuous, we are always able to use them for tracing interfaces with sharp corners and complicated geometries. In term of capturing corners and geometry, it seems that our method is in no way less capable than the other level set approaches. For two phase problems, our algorithms converge to the true solutions independent of the initial values ϕ and $\boldsymbol{c}$.

B. *Two Level Set Functions*

Example 3. For our method, we can start with continuous functions for the level set functions. At convergence, the level set functions are discontinuous functions having values ± 1. In some sense, we are not moving curves. For a given initial value for the level set functions, our algorithms are just finding the correction directions where the level set functions shall move, i.e. up or down at a given point. This makes it easy for our algorithms to capture objects with ”holes” inside the objects. For other level set methods they need to start with curves inside the object, or use approximations to $H-$ and $\delta-$functions having global support if they want to identify inside ”holes”.

To demonstrate the capability of handling holes, we have tested the algorithms using two level set functions for the image depicted in the upper left corner of Fig. 4. The same image has been tested by other level set methods in [10], [14] giving similar results. The image contains convex and concave shapes and ”holes” inside some of the objects. We have imposed the image with noise having $SNR \approx 7.5$. Our algorithms are able find all the objects with rather good accuracy even under a moderate amount of noise, see upper left corner of Fig. 4. The sharp corners, concave shapes and inside ”holes” presents no problems for our algorithms.

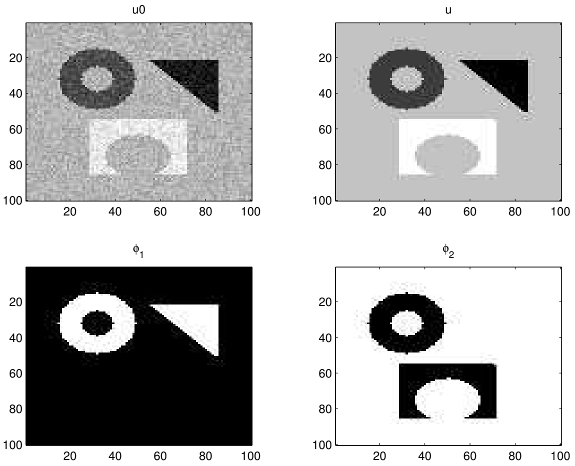


Fig. 4. Two level set functions are utilized for detecting four regions having distinct intensities. The values for the parameters are $\mu = 1 \cdot 10^{-4}$ and $\beta = 1 \cdot 10^{-4}$.

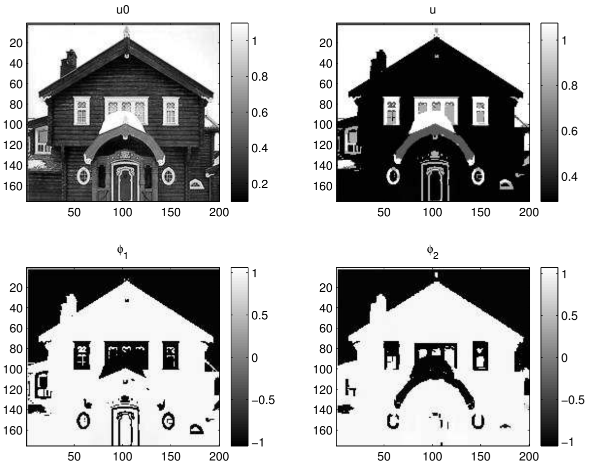


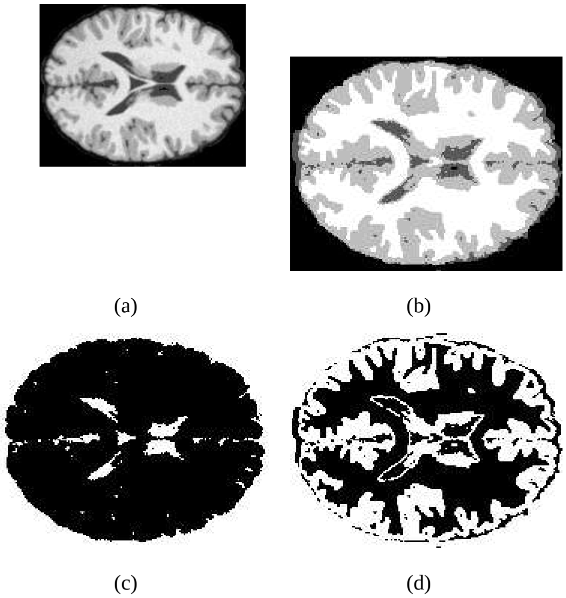
Fig. 5. An image of a house is processed using two level set functions. The values for the parameters are $\mu = 1 \cdot 10^{-2}$ and $\beta = 5 \cdot 10^{-5}$.

Example 4. In this and the next example, we want to test our algorithms for some realistic images. First we have used an image of a house, where the human eye can see that there are approximately four distinct regions in the images, even though the image is more complicated than that. The algorithm is terminated after 3000 iterations. Some of the objects are rather thin and complicated, see the objects around the door, the windows and the roof. Some of the objects only contain a few pixels. Our discontinuous level set functions seem to have no problems to capture all these fine detailed structures. For the traditional level set methods, the level set functions are continuous, the curvature term, i.e. the term $\nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right)$ in (10) may not be a problem. For our level set method, the level set functions are discontinuous at convergence, thus the curvature term in (29) is very large at the discontinuity but zero in the regions where the level set functions are constant. Our numerical tests show that such a discontinuous term produces no problem for our method. On the contrary, the large values of the curvature terms suppress unnecessary oscillations in the curves.

Example 5. To conclude the numerical section we process a MR-image of a brain. We have picked out an image from the Brainweb database. This is an online database from where synthetic MR-images of the human brain can be downloaded. The input image u_0 in Fig. 6 contain 13% noise and 20% inhomogeneity. We refer the reader to (www.bic.mni.mcgill.ca/brainweb) for explanation of the amount of noise and inhomogeneity. This image is difficult due to the fact that the curves are complicated and the intensity values are not nearly constant inside each phase, since the intensity values have a 20% inhomogeneity inside each phase. The computed result is shown in Fig. 6. We are able to extract four different regions. This is the only example using two level set functions where we do update the penalty parameter. We let $\mu^0 = 5 \cdot 10^{-3}$ for the first 3000 iterations. Then we set $\mu = 5 \cdot 10^{-1}$, and iterate for another 1000 iterations. The accuracy and quality of the segmented image is as good as the results produced by traditional level set methods.

VI. CONCLUSION

In this work and also in [14], we have proposed some piecewise constant level set methods for capturing interfaces. Numerical experiments show that these methods are able to trace interfaces with complicated geometries and sharp corners. The level set functions are discontinuous at convergence, but the minimization functionals are smooth and at least locally convex. In this work, we have only tested these methods for image segmentation, where we have used some simple gradient methods for the iterative algorithms. Due to the fact that the functionals are smooth, we shall be able design fast iterative algorithms to solve the minimization problems, i.e. by using Newton type of iterations instead of the simple gradient methods. The numerical results indicate that our methods give as good results as the continuous level set functions. Since our methods are not moving the interfaces during the iterative procedure, it has some advantages in treating geometries, for example in a situation where inside ”holes” need to be identified. Another virtue of not moving the interfaces is that the restrictions on the time-step are weakened. By using our level set approach, we removed the reinitilization procedure needed for the traditional level set approach. In addition, we also avoided the non-differentiability associated with the Heaviside and Delta functions. It is not necessary to use some smoothed and regularized counter parts for the Heaviside and Delta functions. Our approach is truely variational, i.e. the equations we need to solve are coming from the Euler-Lagrange equations for some smooth convex functions.



(a) (b)

(c) (d)

Fig. 6. Two level set functions are used to find four regions in a MR-image u_0 of a brain (a). This is a synthetically produced image, downloaded from www.bic.mni.mcgill.ca/brainweb. The value for $\beta = 1 \cdot 10^{-3}$. (b) u, a piecewise constant approximation to u_0. (c),(d) The piecewise constant level set functions ϕ_1 and ϕ_2. In (c) and (d), white represents 1 and black represents -1.

References

[1] 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, 1988.

[2] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Trans. Image Processing*, vol. 10, no. 2, pp. 266–277, 2001.

[3] M. Sussman, P. Smereka, and S. Osher, "A level set approach for computing solutions to incompressible two phase flow," *J. Comput. Phys.*, vol. 114, pp. 146–159, 1994.

[4] T. F. Chan and X.-C. Tai, "Level set and total variation regularization for elliptic inverse problems with discontinuous coefficients," *J. Comput. Phys.*, vol. 193, pp. 40–66, 2003.

[5] R. P. Fedkiw, G. Sapiro, and C.-W. Shu, "Shock capturing, level sets, and PDE based methods in computer vision and image processing: a review of Osher's contributions," *J. Comput. Phys.*, vol. 185, no. 2, pp. 309–341, 2003.

[6] S. Osher and R. Fedkiw, "Level set methods: An overview and some recent results," *J: Comput. Phys.*, vol. 169, no. 2, pp. 463–502, 2001.

[7] ——, *Level set methods and dynamic implicit surfaces*, ser. Applied Mathematical Sciences. New York: Springer-Verlag, 2003, vol. 153.

[8] J. A. Sethian, *Level set methods and fast marching methods*, 2nd ed., ser. Cambridge Monographs on Applied and Computational Mathematics. Cambridge: Cambridge University Press, 1999, vol. 3, evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science.

[9] X.-C. Tai and T. F. Chan, "Multiple level set methods and some applications for identifying piecewise constant functions," *CAM-UCLA*, no. 03-68, 2003.

[10] L. A. Vese and T. F. Chan, "A multiphase level set framevork for image segmentation using the mumford and shah model," *International Journal of Computer Vision*, vol. 50, no. 3, pp. 271–293, 2002.

[11] B. Song and T. Chan, "A fast algorithm for level set based optimization," *CAM-UCLA*, no. 68, 2002.

[12] D. Mumford and J. Shah, "Optimal approximation by piecewise smooth functions and assosiated variational problems," *Comm. Pure Appl. Math*, vol. 42, p. 577685, 1989.

[13] A. Chambolle, "Image segmentation by variational methods: Mumford and Shah functional and the discrete approximations," *SIAM J. Appl. Math.*, vol. 55, no. 3, pp. 827–863, 1995.

[14] J. Lie, M. Lysaker, and X.-C. Tai, "A variant of the level set method and applications to image segmentation," *CAM-UCLA*, 2003.

[15] Y. R. Tsai, "Rapid and accurate computation of the distance function using grids." *J. Comput. Phys.*, vol. 178, pp. 175–195, 2002.

[16] M. Kass, A.Witkin, and D. Terzopulos, "Snakes: Active contour models," *Int. J. Comput. Vision*, vol. 1, pp. 321–331, 1988.

[17] V. Caselles, F. Catté, T. Coll, and F. Dibos, "A geometric model for active contours in image processing," *Numer. Math.*, vol. 66, no. 1, pp. 1–31, 1993.

[18] D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods*, ser. Computer Science and Applied Mathematics. New York: Academic Press Inc., 1982.

[19] J. Nocedal and S. J. Wright, *Numerical optimization*, ser. Springer Series in Operations Research. New York: Springer-Verlag, 1999.

[20] K. Kunisch and X.-C. Tai, "Sequential and parallel splitting methods for bilinear control problems in Hilbert spaces," *SIAM J. Numer. Anal.*, vol. 34, pp. 91–118, 1997.

[21] T. F. Chan and X.-C. Tai, "Identification of discontinuous coefficients in elliptic problems using total variation regularization," *SIAM J. Sci. Comput.*, vol. 25, pp. 881–904, 2003.

[22] F. R. Dias Velasco, "Thresholding using the ISODATA clustering algorithm," *IEEE Trans. Systems Man Cybernet.*, vol. 10, no. 11, pp. 771–774, 1980.