Variational Image Segmentation using Discontinuous Level Set Formulations

Cand. Scient. Thesis in Applied Mathematics

Johan Lie



Department of Mathematics
University of Bergen

December 12th 2003

Preface

In this Cand. Scient Thesis, I have studied image processing by using a variational approach. Xue-Cheng Tai, Marius Lysaker and I have been working together, and this has resulted in two scientific papers on level set methods, appended to this Cand. Scient. Thesis.

I have been working on the first implementation of the method in appendix A, to see how the ideas worked. This was later reimplemented by Marius. Then I have implemented all of the numerics needed for the article in appendix B. All numerical computations are done using MatLab. The article in appendix B is written by me, closely guided by Xue-Cheng and Marius.

The first part of this Cand. Scient. Thesis is written as a short introduction to the two appendices, mainly to introduce the methods and ideas used in the two articles.

I want to thank my supervisor Xue-Cheng Tai for all his intuition, for being very kind and helpful, and for guiding me through this study.

I will also thank Marius Lysaker and Erlend Hodneland for all the hours of help, and fruitful discussions.

Thanks to my fiancée Siv for always being supportive, and little Ingrid for making me smile.

Least but not last, thank YOU for considering reading this stuff!

Bergen, dec. 2003
Johan Lie

Contents

1 Introduction to Image Processing **1**

2 Machinery **5**
2.1 Minimization Problems 5
2.1.1 Lagrangian Methods 6
2.1.2 Functional Derivative 7
2.1.3 Lagrangian Functional 8
2.1.4 Method of Steepest Descent 8
2.2 Level Set Method 9
2.3 Variational Image Processing 10
2.3.1 Some Functionals used in Image Processing 11
2.3.2 Image Segmentation 12

3 Discontinuous Formulation **15**
3.1 Polynomial 16
3.2 Binary 18

Chapter 1

Introduction to Image Processing

In this work we shall study image processing, in a variational setting. This has been common setting for image processing for the last ten or fifteen years. A considerable amount of pioneering work on image processing problems has been done by the image processing group of the Applied Mathematic institute of University of California, Los Angeles:

`www.math.ucla.edu/~imagers`

Some people from the Mathematics institute of the University in Bergen have been working on image problems for the last few years. This was initiated by Xue-Cheng Tai.

`www.mi.uib.no/BBG`

The subject of variational image processing is huge, and in this Cand. Scient. thesis methods applicable to image segmentation has been studied. First we consider some basic facts about images, and image processing. We only treat two dimensional images, but the generalizations to higher dimensions are straight forward.

Two dimensional digital images consist of a finite number of small squares called pixels. Each pixel x_{ij} in a grayscale image has an intensity value $u(x_{ij})$. Color images have three intensity values, one for each of the three colors -red, green and blue. Each component is usually an integer in the range $[0\,, 255]$. We will work with grayscale images. A grayscale image u can be represented by a matrix. Having defined images as matrices, we are able to manipulate them using the machinery of linear algebra, calculus, and statistics. It is natural to interpret u as a surface defined over Ω, a rectangular subset of $\mathbb{R}^2$. Thus we consider an image as a mapping:

$$u : \Omega \to \mathbb{R}. \tag{1.1}$$

High intensities corresponds to peaks and low intensities corresponds to valleys. The interpretation of u as a surface, suggests looking at u as a piecewise constant discretization of a function belonging to a proper function space. This function space must allow discontinuities, and at the same time not allow to much oscillations. One possible choice is the space of functions of Total Variation.

Image processing tasks are of a very diverse nature. We will in the following shortly describe a few fundamental tasks.

Registration Image registration is the process of finding a transformation between two or more images, such that they can be compared subject to a similarity measure. The human eye knows how to make this transformation automatically, but computers don't. A simple example is shown in Figure 1.1.

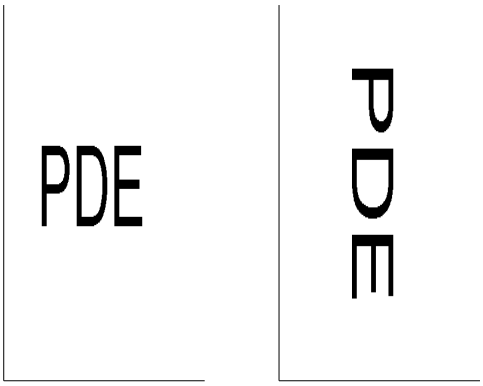


Figure 1.1: Two images containing the same information. To be able to compare them using a computer, a transformation between them must be found.

Noise Removal Very often images are imposed with noise. This noise is usually considered to be additive:

$$u_0 = u + \eta \tag{1.2}$$

Here η is noise, u is the real image, and u_0 is the observed image. η can for example be Gaussian distributed. It will be favorable to remove the noise η, without damaging any structures in the image. The noise removal could be done for aesthetical reasons, but the most important reason for doing noise removal on images is that the presence of noise is troublesome when further processing the images using a computer.

Segmentation Image segmentation is the process of recognizing various features in an image, and to point out where these characteristic features appear. From an image u_0 one would like to construct a simplified image u, a cartoon showing the essential features of u_0, and a rule how to pick out sub domains of Ω.

Inpainting The process of image inpainting is to reconstruct images containing domains where the data is damaged or missing. Generally the domain where the data is missing is much greater than a few pixels, and noise removal algorithms cannot be applied. Since no information from the missing parts of the image is available, either information from the surroundings, or from a data base can be used for reconstructing the image. The term inpainting is also used about the process of interpolation

performed for making it possible to zoom into sub pixel details in images. In many digital cameras there is built-in inpainting hardware for interpolation from the set of recorded pixels to a bigger set of pixels.

Often a combination of the above tasks is considered. This is the case for the process we have been studying, an image segmentation problem, where the image could contain noise. The noise removal is done as an integrated part of the process. Our segmentation process is explained in chapter 3, after some machinery is developed in chapter 2.

Chapter 2

Machinery

In this chapter we introduce the methods used for solving the segmentation problem. We start with minimization problems for functions, and later generalize this to functionals. Then we explain some basics about level set methods, introduce the variational setting for image processing, and conclude this chapter by looking at an existing level set formulation for the image segmentation problem.

2.1 Minimization Problems

Given a function $f : \mathbb{R}^n \to \mathbb{R}$, we can formulate the unconstrained minimization problem: Find $\boldsymbol{x}^*$ such that $f(\boldsymbol{x}^*) < f(\boldsymbol{x})\ \forall \boldsymbol{x} \in \mathbb{R}^n$. This is usually written as:

$$\min f(\boldsymbol{x}). \tag{2.1}$$

If such an $\boldsymbol{x}^*$ exists, $f(\boldsymbol{x}^*)$ is called a global minimum of f. We refer to $f(\boldsymbol{x}^*)$ as a local minimum of f if $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})\ \forall\, \boldsymbol{x}$ sufficiently near $\boldsymbol{x}^*$. $f(\boldsymbol{x}^*)$ is called a strict local minimum if $f(\boldsymbol{x}^*) < f(\boldsymbol{x})\ \forall\, \boldsymbol{x}$ sufficiently near $\boldsymbol{x}^*$. Local and global maxima of f are similarly defined, by switching around the binary operators. A function f can have a local or global extremum point $\boldsymbol{x}^*$ if:

$\boldsymbol{x}^*$ is a critical point; $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$, or
$\boldsymbol{x}^*$ is a singular point; $\nabla f(\boldsymbol{x}^*)$ does not exist, or
$\boldsymbol{x}^*$ is a boundary point of the domain of f.

A critical point can be a minimum, a maximum or a saddle point. In many applications we need to find a minimum $f(\boldsymbol{x}^*)$ of f, constrained to $K \subset \mathbb{R}^n$. Then we can formulate the equality constrained minimization problem: Find $\boldsymbol{x}^*$ such that $f(\boldsymbol{x}^*) < f(\boldsymbol{x})\ \forall \boldsymbol{x} \in K$. If $g(\boldsymbol{x}) = 0 \Leftrightarrow \boldsymbol{x} \in K$, we can write the constrained minimization as:

$$\min f(\boldsymbol{x}) \quad \text{subject to} \quad g(\boldsymbol{x}) = 0. \tag{2.2}$$

We can similarly define inequality constrained problems, for example by minimizing f subject to $g(\boldsymbol{x}) < 0$.

We will illustrate the constrained minimization problem with an example. Let $f(x, y) : \mathbb{R}^2 \to \mathbb{R}$. A typical constrained minimization problem is: Find (x^*, y^*) s.t $f(x^*, y^*) < f(x, y)\ \forall\ (x, y)$ and s.t (x^*, y^*) is on the unit circle in $\mathbb{R}^2$. This can be formulated as the constrained minimization problem:

$$\min f(x, y) \quad \text{subject to} \quad x^2 + y^2 - 1 = 0. \tag{2.3}$$

By introducing the equality constraint $g(x, y) = x^2 + y^2 - 1 = 0$ we only allow solutions (x, y) s.t $g(x, y) = 0$. For the particular choice of g the solution will be found at the unit circle in $\mathbb{R}^2$. If we change to the inequality constraint $g \leq x^2 + y^2 - 1$, the minimum of f must be found somewhere in the unit disc in $\mathbb{R}^2$.

2.1.1 Lagrangian Methods

One technique for finding extreme values of $f(\boldsymbol{x})$ subject to the equality constrained problem is the method of Lagrangian multipliers. This method can be used to find solutions both analytically and numerically. Define the Lagrangian function by:

$$L(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) + \lambda(\boldsymbol{x})g(\boldsymbol{x}), \tag{2.4}$$

where λ is a scalar called a Lagrangian multiplier. We look for an extreme value $\boldsymbol{x}^*$. Assume f and g have continuous partial derivatives near $\boldsymbol{x}^*$, and on the curve $C : g(\boldsymbol{x}^*) = 0$. Suppose also that f has a minimum or maximum $\boldsymbol{x}^*$ when restricted to the set $g = 0$. Assume $\boldsymbol{x}^*$ is not an endpoint of C, and $\nabla g(\boldsymbol{x}^*) \neq \mathbf{0}$. Then there exists a number λ^* such that $(\boldsymbol{x}^*, \lambda^*)$ is a critical point of L. This suggests that searching an extremum point $f(\boldsymbol{x}^*)$ can be accomplished by looking for critical points of $L(\boldsymbol{x}^*, \lambda)$. At a critical point of $L(\boldsymbol{x}^*, \lambda)$ we must have:

$$\frac{\partial L}{\partial x_i} = 0\ \forall\, i \leq \dim(\boldsymbol{x}), \quad \frac{\partial L}{\partial \lambda} = 0. \tag{2.5}$$

If f and g are smooth functions in a neighborhood of the extremum point $\boldsymbol{x}^*$, and there exists a $\boldsymbol{x}^*$ constrained to the set determined by $g = 0$, one can expect to find the extremum point by using the method of Lagrangian multipliers, if $\boldsymbol{x}^*$ is not at the endpoint of the domain. Thus the constrained optimization problem can be solved by finding critical points of L. The method can be generalized to finding extremum points of

$$f(x_1, x_2, \cdots, x_n), \tag{2.6}$$

subject to $m < n$ constraints,

$$g_1(x_1, x_2, \cdots, x_n) = 0, \cdots, g_m(x_1, x_2, \cdots, x_n) = 0, \tag{2.7}$$

by finding extremum points of the corresponding Lagrangian function:

$$L(x_1, \cdots, x_n, \lambda_1 \cdots \lambda_m) = f(x_1, \cdots, x_n) + \lambda_1 g_1(x_1, \cdots, x_n) + \cdots + \lambda_m g_m(x_1, \cdots, x_n), \quad (2.8)$$

as long as $f, g_1, \cdots, g_m$ are differentiable, and $\nabla g_1 \cdots, \nabla g_m$ are all linearly independent. There is one drawback to the method of Lagrangian multiplier. No general method exists for finding λ^* and $\boldsymbol{x}^*$.

2.1.2 Functional Derivative

The terminology of minimization problems earlier used about functions can be adapted to functionals as well. We only need to introduce the concept of derivative of a functional. Let V be a Banach space. Let $F : V \to \mathbb{R}$ be a continuous functional. The derivative of F in the direction of v is defined by:

$$F'(u) \cdot v = \lim_{\epsilon \to 0} \frac{F(u + \epsilon v) - F(u)}{\epsilon}. \quad (2.9)$$

Here $F' : V \to V'$. We say that u^* is a critical point of F if

$$F'(u^*) \cdot v = 0 \ \ \forall\, v \in V. \quad (2.10)$$

This is called the critical point condition, or the Euler Lagrange equation for F. Note that a critical point of F is a function $u^* : V \to \mathbb{R}$, while a critical point of a function f is a point $\boldsymbol{x}^* \in \mathbb{R}$. The existence of (2.9) does not imply F is differentiable. This requires another condition to hold. We say that F is Frechet differentiable at u if:

$$F(u + v) = F(u) + F'(u) \cdot v + o(||v||_V) \ \text{ as } \ ||v||_V \to 0. \quad (2.11)$$

Other definitions concerning the differentiability of F exists, one of which is the Gateaux definition of differentiability. G is said to be Gateaux differentiable if

$$\lim_{\epsilon \to 0} \frac{F(u + \epsilon v) - F(u)}{\epsilon} = \int_\Omega G(u) v dx \ \ \forall v \in C_0^\infty(\Omega). \quad (2.12)$$

We note here that the definition of functional derivative is very similar to the definition of derivative of a function. Therefore it is not surprising that actually computing derivatives of a functional is done using similar techniques as when computing derivatives of functions.

The extremum points of certain functionals can be studied by the Calculus of Variations, a generalization of ordinary calculus. The strategy is as follows. Assume $A[u] = 0$, where A is a given possibly nonlinear partial differential operator, and u is unknown. Also assume that we are lucky, and there exists an energy functional $E[u]$, such that

$$E'[u] = A[u]. \quad (2.13)$$

If $E'[u] = 0$, then u is a critical point of E. This in turn means that $A[u] = E'[u] = 0$, and the critical point is also a solution to the PDE $A[u] = 0$.

By turning this argument around, one can solve a PDE $A[u] = 0$ in order to find critical points of the energy functional $E[u]$. That is a strategy used for solving many image processing problems. Of course, some conditions must be imposed on E to make sure this strategy is usable. We do not address this question here.

2.1.3 Lagrangian Functional

The Lagrange multiplier method is also applicable for finding extreme points of functionals. Let $F(u) : V \to \mathbb{R}$ be the functional to be minimized, subject to a constraint $K(u) = 0$. $u \in V$ is a function $u : \Omega \to \mathbb{R}$. The Lagrangian functional is defined by:

$$L(u, \lambda) = F(u) + \int_\Omega \lambda K(u) dx \tag{2.14}$$

Now we treat $\lambda : \Omega \to \mathbb{R}$ as a function, as opposed to earlier when λ was a scalar. Similarly as earlier we look for critical points of L to find a minimum of F. Recalling (2.10), at a critical point of L, we must have:

$$\frac{\partial L}{\partial u} = 0 \text{ and } \frac{\partial L}{\partial \lambda} = 0. \tag{2.15}$$

Sometimes, finding a point u^* such that (2.15) is fulfilled is not possible using the Lagrangian multiplier method. In this work we have used both the multiplier method, and in addition the augmented Lagrangian method, for numerically finding minima of a functional using an iterative approach. Define the augmented Lagrangian functional as:

$$L_\mu(u, \lambda) = F(u) + \int_\Omega \lambda K(u) dx + \frac{1}{2}\mu \int_\Omega K^2(u) dx, \tag{2.16}$$

where μ is a nonnegative number. The last term of (2.16) is referred to as a penalty term. If $\lambda = 0$, L_μ is the so-called penalty functional. The idea of a penalty method is to let $\mu \to \infty$ in a iterative manner. Let p denote the sum of the multiplier term and the penalty term:

$$p = \int_\Omega \lambda K(u) dx + \frac{1}{2}\mu \int_\Omega K^2(u) dx, \tag{2.17}$$

If λ is a bounded function, $p = \infty$ if $K(u) \neq 0$, and $p = 0$ if $K(u) = 0$. The idea of the augmented Lagrangian method is to let the penalty parameter increase slowly, and update λ and μ in a clever way to make sure we minimize F, and at the same time $K = 0$. Then $p = 0$, and we have solved the problem.

2.1.4 Method of Steepest Descent

Finding critical points of L and L_μ can be done by using various optimization methods. We use a gradient method. Gradient methods are applicable on both constrained and unconstrained optimization problems. It is an iterative method, and we denote the iterations

by superscript. From an old value u^k we get a new value u^{k+1} by the gradient iteration:

$$u^{k+1} = u^k + \alpha^k d^k \tag{2.18}$$

Here α is a positive parameter, called the step size, and d is a term involving various derivatives of the functional evaluated at the current step. Let us first look at the unconstrained case. Two widely used gradient methods are the method of steepest descent,

$$d^k = -\frac{\partial F}{\partial u}(u^k), \tag{2.19}$$

and Newtons method,

$$d^k = -\left[\frac{\partial^2 F}{\partial u^2}(u^k)\right]^{-1} \frac{\partial F}{\partial u}(u^k). \tag{2.20}$$

We use the method of steepest descent. Using the Lagrangian multiplier method and the augmented Lagrangian method, in combination with the gradient iteration, we end up with the schemes:

$$u^{k+1} = u^k - \alpha^k \frac{\partial L}{\partial u}(u^k), \tag{2.21}$$

$$u^{k+1} = u^k - \alpha^k \frac{\partial L_\mu}{\partial u}(u^k). \tag{2.22}$$

The step size can either be fixed, or properly chosen at each step, according to a minimization rule. We have used a fixed step size.

2.2 Level Set Method

The level set method was first introduced in 1987, by Osher and Sethian in [1], for capturing moving fronts of nonlinear hyperbolic PDEs. The method has been used for extremely many applications, some of which are related to image processing. The idea of the level set method is to implicitly represent fronts by the zero level of higher dimensional functions ϕ. Let us for now assume we want to represent a curve Γ in $\mathbb{R}^2$. This curve is one dimensional. Using a level set approach, this curve is represented by the zero level of a two dimensional surface $(x, y, \phi(x, y))$, defined over $\mathbb{R}^2$, i.e the curve is the intersection of the plane $(x, y, 0)$ and $\phi(x, y)$. This is shown in Figures 2.1 and 2.2. The implicit representation of Γ makes it possible to treat the set interior to Γ, int(Γ), as the set $(x, y) \ : \ \phi(x, y) > 0$, and exterior of Γ, ext(Γ) as the set $(x, y) \ : \ \phi(x, y) < 0$. Note that using the level set representation of Γ, these sets need not be connected. By introducing a time parameter t, $\Gamma(t)$ can be implicitly moved by evolving $\phi(\boldsymbol{x}, t)$. How $\phi(\boldsymbol{x}, t)$ is moved, is application dependent. In order to implicitly move the curve, the convection equation

$$\phi_t + \boldsymbol{v} \cdot \nabla\phi = 0 \tag{2.23}$$

is solved. Here $\boldsymbol{v}$ is a velocity field which can be dependent of time, geometry ,internal or external physics, etc. When considering the evolvement of the curve in $\mathbb{R}^2$, $\boldsymbol{v} \cdot \nabla\phi =$

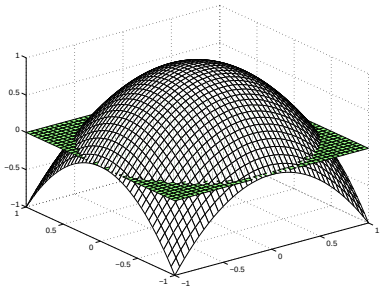


Figure 2.1: The levelset function $(x, y, \phi(x, y))$ and (x,y,0).

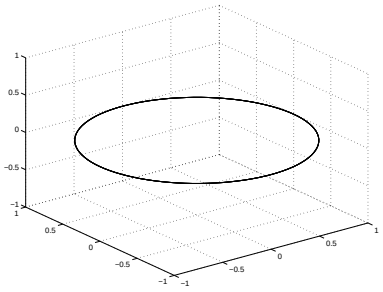


Figure 2.2: $\Gamma : \ \phi = 0$.

$v_1\phi_x + v_2\phi_y$. Here we note that this equation only says how to move the zero level of ϕ. In some applications computations are done only in a narrow band around the zero level, while in other applications, computations are done in the entire domain. We will in the next section look at an application for image segmentation, where computations are done in the whole domain.

2.3 Variational Image Processing

The idea of variational image processing is that matrices representing images can be viewed as functions, which can be processed using PDEs. The PDEs are usually derived from minimization problems, recall the discussion in section 2.1.2.

Even though the PDEs used in image processing does not need not be elliptic nor parabolic, some of the ideas from the behavior of these equations are used. A solution to a parabolic PDE goes towards a steady state. This steady state is the solution to a corresponding elliptic PDE. By processing an image using PDEs we want to mimic this behavior. This is done by introducing an artificial time parameter to the problem. For a $t > 0$ we define $u(\boldsymbol{x}, t)$ as the processed image. This is called defining a 'scale space' to the problem, with t as a scale. We look for a steady state, which means

$$u(\boldsymbol{x}, t) \to u(\boldsymbol{x}) \quad \text{as} \quad t \to \infty. \tag{2.24}$$

Now, let u_0 be a matrix representing an image to be processed. We assume $\Omega \subset \mathbb{R}^2$.

$$u_0 : \Omega \to \mathbb{R}. \tag{2.25}$$

The image can be processed by finding a steady state of an evolutionary PDE, with $u(\boldsymbol{x}, 0) = u_0$. That is:

$$u_t = \mathcal{F}(u, Du, D^2u, \boldsymbol{x}, t), \quad \boldsymbol{x} \in \Omega. \tag{2.26}$$

$$u(\boldsymbol{x}, 0) = u_0(\boldsymbol{x}). \tag{2.27}$$

In addition some boundary conditions must be imposed, usually a Diriclet or Neumann type. The equation (2.26) is called a flow equation. Let $\boldsymbol{x} = (x, y)$. Then the terms of $\mathcal{F}$ are defined by:

$$Du = (u_x, u_y), \tag{2.28}$$

and

$$D^2u = \begin{pmatrix} u_{xx} & u_{xy} \\ u_{yx} & u_{yy} \end{pmatrix}. \tag{2.29}$$

If we let $\mathcal{F} = \Delta u$, we get the heat equation. Oscillations in u_0 will be smoothed out, and a steady state u will be reached after a finite time if the boundary condition is independent of time. The resulting image will be a blurred version of u_0. Some of the flow equations used in image processing are related to this equation.

The flow-equation (2.26) is usually found by minimizing an energy functional $E : V \to \mathbb{R}$. Using the definition of a critical point (2.10), at a minimum $u^* \in V$ of E we must have $E'(u^*) = 0$. By setting

$$u_t = -E'(u), \tag{2.30}$$

finding a steady state u^* of this equation implies $E'(u^*) = 0$. Usually the time-derivative in (2.30) is discretized using the simple first order accurate forward Euler scheme:

$$u_t \approx \frac{u^{k+1} - u^k}{\Delta t}. \tag{2.31}$$

Using this discretization leads to the gradient iteration scheme (2.18). When the level set method is used, instead a flow equation for ϕ is solved:

$$\phi_t = -E'(\phi), \tag{2.32}$$

to find the minimum $E(\phi^*)$. Of course we do not generally know if a unique minimizer exists. This is related to wether or not the functional E is convex. If the functional is convex, there exists only one minimum, and this minimum is global.

2.3.1 Some Functionals used in Image Processing

In this section we will present some image functionals used in the literature. This is meant as a motivation for introducing the functional that we later use for image segmentation.

One well-known functional in image processing is the Total Variation (TV) functional. Minimizations of this functional has been used in image denoising and image inpainting models. It is also commonly used as a regularizer. We will later use the TV functional in the minimization problem for image segmentation. The definition of the TV functional for a function u of one variable is:

$$TV(u) = \sup_{h>0} \int \left| \frac{u(x+h) - u(x)}{h} \right| dx \tag{2.33}$$

Note the similarity with the definition of the derivative. Let now u be a two-dimensional function. Then the TV functional is usually interpreted as:

$$TV(u) = \int_\Omega |\nabla u| dx = \int_\Omega \sqrt{u_x + u_y} dx. \tag{2.34}$$

When used as an image enhancer [2], the functional to be minimized has an extra term to make sure the solution is close to the initial function. The closeness is determined using a Lagrange multiplier λ. λ is either set to be constant or updated during the minimization process.

$$E(u) = \int_\Omega |\nabla u| dx + \frac{\lambda}{2} \int_\Omega (u - u_0)^2 dx \tag{2.35}$$

By choosing $\mathcal{F} = -E'(u)$, the flow-equation to be solved is given by:

$$u_t = \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) - \lambda(u - u_0) \tag{2.36}$$

This is called the Rudin/Osher/Fatemi (ROF) model of image denoising. This model has been studied extensively, and several extensions has been made. We refer to [3] for a discussion of this. The same flow equation can also be used for inpainting, by introducing a small modification. Let $D \subset \Omega$ be a domain in the image where information is missing or damaged. If we let E denote an extended domain around D, with $E \subset D^c$ such that $E \subseteq \Omega$, we get the TV-inpainting functional:

$$E = \int\limits_{E \cup D} |\nabla u| dx + \frac{\lambda}{2} \int\limits_E (u - u_0)^2 dx \tag{2.37}$$

Introducing the multiplier:

$$\lambda_e = \begin{cases} \lambda, & z \in E, \\ 0, & z \in D, \end{cases} \tag{2.38}$$

leads to the flow equation for TV-inpainting:

$$\frac{\partial u}{\partial t} = \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) - \lambda_e(u - u_0). \tag{2.39}$$

2.3.2 Image Segmentation

We will now turn to a popular existing method for image segmentation. Variational image segmentation using a level set formulation was first introduced in [4], but the probably best known work on segmentation using a level set formulation is done by Chan/Vese [5, 6]. We first consider two phase segmentation. Assume u_0 is an image having two approximately constant intensity values, an object and background. Define the fitting functional:

$$E(\Gamma, c_1, c_2) = \int\limits_{int(\Gamma)} |u_0 - c_1|^2 dx + \int\limits_{ext(\Gamma)} |u_0 - c_2|^2 dx. \tag{2.40}$$

Γ is a closed curve in Ω. int(Γ) is the region interior of Γ, and int(Γ) is the region exterior to Γ. $c_1 = mean(u_0(\boldsymbol{x}))$ $\boldsymbol{x} \in$ int(Γ) and $c_2 = mean(u_0(\boldsymbol{x}))$ $\boldsymbol{x} \in$ ext(Γ). The energy E is minimized if Γ is placed on the boundary between the two the object and the background. For the purpose of regularizing, the length of Γ and the area of the interior of Γ can be added to the functional. Using the level set formulation, the energy can be written as:

$$\begin{aligned} E_{cv}(\phi, c_1, c_2) &= \int_\Omega |u_0 - c_1|^2 H(\phi) dx + \int_\Omega |u_0 - c_2|^2 (1 - H(\phi)) dx \\ &+ \nu \int_\Omega H(\phi) dx + \mu \int_\Omega \delta(\phi) |\nabla \phi| dx. \end{aligned} \tag{2.41}$$

By using the level set representation, the integrals are evaluated all over the entire domain. Note that the argument of E_{cv} is essentially the level set function ϕ. This is opposed to the two foregoing functionals (2.35),(2.37), where the argument of the functional was u. This indicates that the corresponding Euler Lagrange equation is a flow equation for ϕ. u which is the piecewise constant approximation to u_0 taking two different constant values c_1 and c_2 can be written as:

$$u(\Gamma, c_1, c_2) = c_1 H(\phi) + c_2 (1 - H(\phi)). \tag{2.42}$$

Differentiating the functional with respect to ϕ, and using (2.30),with ϕ replacing u, the flow equation for the Chan/Vese model for image segmentation becomes:

$$\phi_t = \delta_\epsilon(\phi) \left(\mu \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right) - \nu - (u_0 - c_1)^2 + (u_0 - c_2)^2 \right), \tag{2.43}$$

where $\delta_\epsilon(\phi)$ is a globally positive approximation to $\delta(\phi)$. This model can be extended to multiple phase segmentation by using multiple level set functions. N level set functions can be used for representing up to 2^N phases. By using the sign of two level set functions, the functional to be minimized is:

$$\begin{aligned} F(\boldsymbol{c}, \phi_1, \phi_2) &= \int_\Omega |c_1 - u_0|^2 H(\phi_1) H(\phi_2) dx \\ &+ \int_\Omega |c_2 - u_0|^2 H(\phi_1)(1 - H(\phi_2)) dx \\ &+ \int_\Omega |c_3 - u_0|^2 (1 - H(\phi_1)) H(\phi_2) dx \\ &+ \int_\Omega |c_4 - u_0|^2 (1 - H(\phi_1))(1 - H(\phi_2)) dx \\ &+ \nu \left(\int_\Omega H(\phi_1) dx + \int H(\phi_2) dx \right) \end{aligned} \tag{2.44}$$

$$+ \quad \mu \left(\int_{\Omega} \delta(\phi_1) |\nabla \phi_1| dx + \int \delta(\phi_2) |\nabla \phi_2| dx \right),$$

now with two corresponding Euler-Lagrange equations, on for each of the two level set functions. The piecewise constant function u taking four different values can be written as:

$$u(\boldsymbol{x}) = \begin{cases} c_1, & \text{if } \phi_1(\boldsymbol{x}) > 0 \ , \ \phi_2(\boldsymbol{x}) > 0, \\ c_2, & \text{if } \phi_1(\boldsymbol{x}) > 0 \ , \ \phi_2(\boldsymbol{x}) < 0, \\ c_3, & \text{if } \phi_1(\boldsymbol{x}) < 0 \ , \ \phi_2(\boldsymbol{x}) > 0, \\ c_4, & \text{if } \phi_1(\boldsymbol{x}) < 0 \ , \ \phi_2(\boldsymbol{x}) < 0. \end{cases} \tag{2.45}$$

We note here that the Euler-Lagrange equation include an approximation to $\delta(\phi)$, and u is represented using $H(\phi)$. This can complicate computations. We will in the next section introduce two different ways to avoid using these functions.

Chapter 3

Discontinuous Formulation

Motivated by the Chan/Vese approach [5, 6], we have solved a related minimization problem for image segmentation. Using two special level set formulations introduced in [7] [8], we search for a constrained minimum of the energy functional:

$$F(\phi, \boldsymbol{c}) = \frac{1}{2}\int_{\Omega} |\, u - u_0 \,|^2 dx + \beta R(\phi), \qquad (3.1)$$

where $R : \Omega \rightarrow \mathbb{R}$ is a regularizer, introduced to keep u piecewise constant and avoid unwanted oscillations. The energy functional is of the same form as (2.41). The underlying level set formulation is different from the standard formulation. There are two advantages using our approach: We avoid using the δ and H functions, and we avoid the reinitialization procedure used in standard level set methods.

In the article [7], we use one level set function ϕ for representing multiple phases. In the second article, [8], we use multiple level set functions. For both approaches, we consider dividing an image $u_0 : \Omega \rightarrow \mathbb{R}$ into n distinct sub domains Ω_i, such that $\Omega = \cup_i^n \Omega_i$. The common strategy for both approaches is as follows:

Constraint K: For each phase Ω_i in Ω we only allow the level set function to take a special form. To accomplish this, we introduce an equality constraint $K(\phi) = 0$ to the minimization problem.

Basis functions ψ: Each of the phases Ω_i is represented by a basis function ψ_i, which takes the value 1 in Ω_i, and is zero in $\Omega_j \ \ \forall j \neq i$. ψ is a function of ϕ.

Representation of u: The basis functions are put together to construct u by the linear combination:

$$u = \sum_{i=1}^{n} c_i \psi_i. \tag{3.2}$$

Regularizer R: The basis functions are also used to construct a regularizer:

$$R = \sum_{i=1}^{n} \int_{\Omega} |\nabla \psi_i| dx. \tag{3.3}$$

Solution process: The minimization problem:

$$\min F(\phi, \boldsymbol{c}) \quad \text{subject to} \quad K(\phi) = 0 \tag{3.4}$$

is solved by using Lagrangian methods and steepest descent iterations.

We will refer to the article [7] as the polynomial approach, and the article [8] as the binary approach.

3.1 Polynomial

If we consider dividing an image $u_0 : \Omega \to \mathbb{R}$ into n distinct sub domains Ω_i, such that $\Omega = \cup_i \Omega_i$, we can search a minimum of (3.1) such that $\phi = i$ in Ω_i. Thus we only let ϕ take positive integer values $\leq n$. This implies ϕ must be a discontinuous function. To ensure ϕ only takes integer values, we formulate a constraint $K(\phi)$:

$$K(\phi) = (\phi - 1)(\phi - 2) \cdots (\phi - n). \tag{3.5}$$

If ϕ take one of the values $1, 2, \cdots, n$ at a point $\boldsymbol{x}$, $K(\phi(\boldsymbol{x})) = 0$ at that point. Thus we look for a minimum of the objective functional such that $K(\phi) = 0$ at every point.

Constructing the basis functions ψ_i is most easily done by considering how u must be constructed for a two phase segmentation, and generalize this to multiple phases. Assume $u_0(\Omega_1) = c_1$ and $u_0(\Omega_2) = c_2$. If $\phi = 1$ in Ω_1 and $\phi = 2$ in Ω_2, we can construct u by the following formula:

$$u = -(\phi - 2)c_1 + (\phi - 1)c_2. \tag{3.6}$$

Introducing one more phase Ω_3 by $u_0(\Omega_3) = c_3$, we can construct a three phase u by:

$$u = \frac{1}{2}(\phi - 2)(\phi - 3)c_1 - (\phi - 1)(\phi - 3)c_2 + \frac{1}{2}(\phi - 1)(\phi - 2)c_3. \tag{3.7}$$

We define the polynomial in front of each c_i to be the basis function ψ_i:

$$u = \underbrace{\frac{1}{2}(\phi - 2)(\phi - 3)}_{\psi_1} c_1 + \underbrace{-(\phi - 1)(\phi - 3)}_{\psi_2} c_2 + \underbrace{\frac{1}{2}(\phi - 1)(\phi - 2)}_{\psi_3} c_3. \tag{3.8}$$

This suggests we can write u on the form:

$$u = \sum_i^n c_i \psi_i. \tag{3.9}$$

It is now easily seen by insertion that if ϕ is of the form required, which means $\phi(\boldsymbol{x})$ is a positive integer $i \leq n$, all of the basis functions $\psi_j(\boldsymbol{x}) = 0 \ \forall j \neq i$, and $\psi_i(\boldsymbol{x})$=1.

The structure of each ψ_i is very simple. Since each of them is piecewise constant, and only takes the values 0 and 1, the spatial derivatives of ψ_i are zero except at the discontinuity representing the boundary of phase i. If we measure $|\nabla \psi_i|$ we therefore measure the length of the discontinuity of ψ, or in other words the length of the boundary of phase Ω_i. This has been utilized as a tool for regularizing the minimization problem. Define the regularizer by:

$$R(\psi) = \sum_i^n \int_\Omega |\nabla \psi_i| dx. \tag{3.10}$$

By minimizing this regularizer, we minimize the length of the boundaries of ψ_i.

Using the machinery introduced in chapter 2, minimizing the functional F subject to the constraint K is done by finding a saddle point of the augmented Lagrangian functional:

$$L_\mu(\phi, \boldsymbol{c}, \lambda) = F(\phi, \boldsymbol{c}) + \int_\Omega \lambda K(\phi) dx + \frac{\mu}{2} \int_\Omega |K(\phi)|^2 dx. \tag{3.11}$$

Note that in the article, we have denoted the penalty parameter as r, instead of μ. The saddle point is a minimum of L_μ with respect to ϕ and $\boldsymbol{c}$, and a maximum with respect to λ. Following essentially the same approach as introduced in 2.3, we solve the equation:

$$\phi_t = -\frac{\partial L_\mu}{\partial \phi} \tag{3.12}$$

to steady state to find the minimum of $F(\phi, \boldsymbol{c})$.

3.2 Binary

In [8] we have searched for a minimum of the objective functional by using multiple level set functions. Assume we want to segment an image into two regions Ω_1 and Ω_2. The strategy used is to search a minimum of (3.1) such that $\phi = 1$ in Ω_1, and $\phi = -1$ in Ω_2. This can be accomplished in the same manner as earlier by introducing the constraint:

$$K(\phi) = \phi^2 - 1. \tag{3.13}$$

If $\phi(\boldsymbol{x}) = 1$ or $\phi(\boldsymbol{x}) = -1$ at every point in the image, $K = 0$. Multiple regions can be detected by introducing more level set functions. This is done by using essentially the same idea used for standard level set functions. If we use N level set functions, we can distinguish between $n = 2^N$ regions. We arrange the functions in a vector, $\boldsymbol{\phi} = \{\phi_1, \phi_2, \cdots, \phi_N\}$. Then the corresponding constraint is a vector having the same number of elements as $\boldsymbol{\phi}$:

$$\boldsymbol{K} = \{K_1, K_2, \cdots, K_N\}, \tag{3.14}$$

with:

$$K_i(\phi_i) = \phi_i^2 - 1. \tag{3.15}$$

If $\phi_i(\boldsymbol{x}) = 1$ or $\phi_i(\boldsymbol{x}) = -1$ at every point in the domain, then $\boldsymbol{K} = \boldsymbol{0}$.

Constructing the basis functions ψ_i corresponding to Ω_i is done using very similar techniques as for the previous approach. The main difference is that now a set of (ϕ_i+1) and (ϕ_i-1) are used as building blocks of each ψ_i function instead of $(\phi-1), (\phi-2), \cdots, (\phi-n)$. Let us first consider the two sub-domain case. In Ω_1, $u = c_1$, and in Ω_2, $u = c_2$. If $\phi = 1$ in Ω_1 and $\phi = -1$ in Ω_2, a representation formula u is given by:

$$u = \frac{c_1}{2}(\phi + 1) - \frac{c_2}{2}(\phi - 1). \tag{3.16}$$

Extending this for distinguishing between four different phases is done by using two level set functions. If $u = c_i$ in Ω_i, $i = 1, \cdots, 4$, u can be written as:

$$u = \frac{c_1}{4}(\phi_1 + 1)(\phi_2 + 1) - \frac{c_2}{4}(\phi_1 + 1)(\phi_2 - 1) - \frac{c_3}{4}(\phi_1 - 1)(\phi_2 + 1) + \frac{c_4}{4}(\phi_1 - 1)(\phi_2 - 1).$$

This is extended to multiple phases by a similar technique as for the polynomial approach. One difference is that when using multiple level set functions, the basis functions will not be polynomials of as high degree as the basis functions used in the polynomial approach. The basis functions for the four phase u are given as:

$$\begin{aligned} u &= \underbrace{\frac{1}{4}(\phi_1 + 1)(\phi_2 + 1)}_{\psi_1} c_1 + \underbrace{\frac{-1}{4}(\phi_1 + 1)(\phi_2 - 1)}_{\psi_2} c_2 \\ &+ \underbrace{\frac{-1}{4}(\phi_1 - 1)(\phi_2 + 1)}_{\psi_3} c_3 + \underbrace{\frac{1}{4}(\phi_1 - 1)(\phi_2 - 1)}_{\psi_4} c_4. \end{aligned} \tag{3.17}$$

We see that just like when using the polynomial approach, we end up with a representation formula for u as:

$$u = \sum_{i}^{n} c_i \psi_i. \tag{3.18}$$

We use the same regularizer for the Binary approach as for the Polynomial approach. The only difference is the way ψ is represented:

$$R(\psi) = \sum_{i}^{2^N} \int_{\Omega} |\nabla \psi_i| dx. \tag{3.19}$$

Each basis function ψ_i, $i = 1 \cdots 2^N$ is here dependent of N level set functions ϕ_j, $j = 1 \cdots N$.

The minimum of the objective functional is found by using two different Lagrangian methods, by searching saddlepoints of a Lagrangian and an augmented Lagrangian functional. This is very similar as in the Polynomial approach.

Bibliography

[1] S. Osher and J. Sethian, "Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulation." *J. Comput. Phys.*, vol. 79, pp. 12–49, 1988.

[2] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithm," *Physica D.*, vol. 60, pp. 259–268, 1992.

[3] B. Song, "Topics in variational pde image segmentation, inpainting and denoising," Ph.D. dissertation, University of California Los Angeles, 2003.

[4] V. Caselles, F. Catte, T. Coll, and F. Dibos, "A geometric model for active contours in image processing," *Numerische Mathematik*, vol. 66, pp. 1–31, 1993.

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

[6] 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.

[7] J. Lie, M. Lysaker, and X.-C. Tai, "A variant of the levelset method and applications to image segmentation," *Submitted to SIAM Journal on Applied Mathematics*, 2003.

[8] ——, "A binary level set model and some applications to image processing," 2003.

Appendix A

A VARIANT OF THE LEVEL SET METHOD AND APPLICATIONS TO IMAGE SEGMENTATION

JOHAN LIE*, MARIUS LYSAKER†, AND XUE-CHENG TAI‡

Abstract. In this paper we propose a variant of the level set formulation for identifying curves separating regions into different phases. In classical level set approaches, the sign of n level set functions are utilized to identify up to 2^n phases. The novelty in our approach is to introduce a piecewise constant level set function and use each constant value to represent an unique phase. If 2^n phases should be identified, the level set function must approach 2^n predetermined constants. We just need one level set function to represent 2^n unique phases and this gain the storage capacity. Further, the reinitializing procedure requested in classical level set methods is superfluous using our approach. The minimization functional for our approach is convex and differentiable and thus avoid some of the problems with the non-differentiability of the Delta and Heaviside functions. Numerical examples will be given and we shall also compare our method with related approaches.

Key words. Level set, energy minimizing, partial differential equations, segmentation.

1. Introduction to Related Level Set Methods.

A function u defined on an open and bounded domain $\Omega \in \mathbb{R}^m$ may have different properties in distinct regions of Ω. In many applications, one needs to separate Ω into an union of subdomains, i.e. $\Omega = \cup_{i=1}^{n} \Omega_i$, so that distinctive properties of u can be linked to different subdomains and incorporated into a modeling process. Several approaches have been used to find a propitious partition of Ω, and some of them are shortly described below. For the purpose of illustration, only the two-dimensional case is considered here.

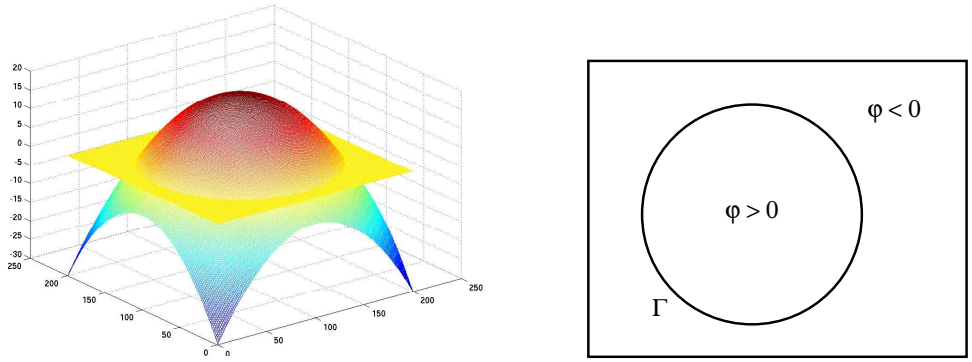
One approach to find a partition of Ω is to use an active contour. The basic idea is to let a closed curve C separate Ω into two disjunctive subdomains. Define the region inside C for Ω_1, and the region outside C for Ω_2. The curve evolves subject to constraints from u together with some smoothness requirements of C. Traditionally, the curve C is parameterized as $\mathbf{x} = \mathbf{x}(s)$, $s \in [0, 1]$. A convenient way to approximate an explicit representation of C, is to divide s into a finite set of points $0 = s_0 < \cdots < s_{i-1} < s_i < s_{i+1} \cdots < s_n = 1$, where the subintervals $[s_i, s_{i+1}]$ can vary in size and then update the particle $\mathbf{x}$ in an iterative way. For some applications, it might be difficult to handle geometrical changes of the curve using this kind of explicit approach. Numerical instability could also be a problem, since the points on the curve may get clustered.

A recent and successful way to represent an evolving curve is the level set method invented by Osher and Setian in [11]. With the level set representation a change of topology, such as merging and breaking of the curve, can be handled in an efficient way. The curve that divide Ω into two subdomains is represented implicitly as the zero level set curve $\Gamma = \{\mathbf{x} \in \Omega,\ \phi(\mathbf{x}) = 0\}$ of a higher dimension level set function ϕ. Rather than evolving the curve itself, the level set approach builds Γ into a function ϕ. Normally, ϕ is initialized as the signed distance function:

$$\phi(\mathbf{x}) = \begin{cases} \text{distance}(\mathbf{x}, \Gamma), & \mathbf{x} \in \text{interior of } \Gamma \\ -\text{distance}(\mathbf{x}, \Gamma), & \mathbf{x} \in \text{exterior of } \Gamma. \end{cases} \tag{1.1}$$

In Fig. 1.1(a) a simple level set function ϕ is illustrated, and its associated interface Γ is given in Fig. 1.1(b). Γ divide Ω into two separate sub-domains. By moving ϕ

*Department of Mathematics, University of Bergen, Norway (johanl@mi.uib.no).
†Department of Mathematics, University of Bergen, Norway (mariusl@mi.uib.no).
‡Department of Mathematics, University of Bergen, Norway (tai@mi.uib.no).



(a) The level set function ϕ.

(b) The interface Γ.

FIG. 1.1. *Plot of the higher dimension function* $\phi(\mathbf{x})$ *and the interface* $\Gamma = \{\mathbf{x} \in \Omega, \ \ \phi(\mathbf{x}) = 0\}$.

up or down at different locations, the interface evolves. To divide Ω into more than two domains, one needs to use multiple level set functions [17, 12, 5, 16]. One of the concerns of multiple level set methods is to avoid vacuum and overlap between the different phases [17]. In [16], the following approach was used to create multiple phases without vacuum and overlaps. Assume we introduce two closed curves: Γ_1 and Γ_2, where each curve Γ_i are associated with a level set function ϕ_i, $i = 1, 2$. The domain Ω can now be divided into 4 parts:

$$\begin{aligned} \Omega_1 =&\{\mathbf{x} \in \Omega, \ \ \phi_1 > 0, \ \ \phi_2 > 0\}, \quad \Omega_2 = \{\mathbf{x} \in \Omega, \ \ \phi_1 > 0, \ \ \phi_2 < 0\}, \\ \Omega_3 =&\{\mathbf{x} \in \Omega, \ \ \phi_1 < 0, \ \ \phi_2 > 0\}, \quad \Omega_4 = \{\mathbf{x} \in \Omega, \ \ \phi_1 < 0, \ \ \phi_2 < 0\}. \end{aligned} \tag{1.2}$$

By generalizing further, n level set functions give the possibility of 2^n regions [17, 5]. These regions will be disjunct (no overlap) and their union is Ω (no vacuum).

In this work, we propose an alternative approach. Assume that we need to find n regions $\{\Omega_i\}_{i=1}^n$ which form a portion of Ω. Some of the regions can be empty. This means that we can over-estimate the total number of phases for practical applications and superfluous phases will disappear at convergence. In order to find the regions, we just need to find a piecewise constant function which takes values:

$$\phi = i \text{ in } \Omega_i, \quad i = 1, 2, \cdots, n. \tag{1.3}$$

The discontinuities of ϕ give us the curves that separate the regions. For some applications, we could formulate the identification of the curves as a minimization problem with respect to ϕ under some constraints. We believe that this level set formulation can be applied in different applications where the classical level set methods are used today. In this work, we restrict ourself to one special case: image segmentation. For such an application, we shall formulate the segmentation problem as a minimization problem with respect to ϕ.

The idea of using one level set function to represent several phases is however not completely new. In island dynamics models for epitaxial growth of thin films a large number of individual interfaces have to be identified. In [9, 6] they use one level set function to track islands on different monolayers, i.e., they let $\phi = 0$ represent the

island boundaries of the first monolayer, $\phi = 1$ represent the island boundaries of the second monolayer and so on. By using just one level set function rather than one per layer, their algorithm is kept simple and memory costs are kept low. We will use a similar idea to track different segments in an image. However, to get this terrace level set formulation to fit into a segmentation framework some innovative work has been done in this paper. These issues are addressed in the next section.

Further, we would also like to mention that piecewise constant level set methods have been used in image segmentation before. In [14] they defined an energy functional dependent on $H(\phi_i)$ where $\phi_i = \pm 1$, and they simply checked if the energy decrease or not when one of the level set functions changed sign at a given point. This method is extremely fast, but the authors give some examples where this method will fail.

Let us conclude this first section with the outline for this paper. In section 2 we introduce a functional motivated from an optimal approximation problem. For the approximation problem we will use the partition described in (1.3) to express a general function as a piecewise constant function. In section 3 the augmented Lagrangian method is introduced as a tool to solve the optimal approximation problem. In this section we also describe our algorithm, together with some other implementation details. The novelty in our approach, compared with classical level set methods, is reported in section 4. Finally, section 5 is devoted to numerical examples.

2. Representing Piecewise Constant Functions. We use image segmentation as an example to show that our method can be utilized to identify curves separating regions. Let u_0 be a function defined in a domain Ω, where u_0 reflects the intensity values for a given gray scale image, possibly imposed with noise. It is assumed that the intensity value of the true image is slowly varying inside each phase, but has large jumps between the phases. In order to find the discontinuities, we try to find a piecewise constant function to approximate the function u_0 and the discontinuities of the piecewise constant function will give us the boundary between the different phases.

In order to find a piecewise constant function, we need to find the value for the constants and the location of the discontinuities. To find the locations of the discontinuities, the curve that separates the regions needs to be determined. Mumford and Shah [10] proposed to solve the "minimal" partition problem to approximate a function u_0 by a piecewise constant function u:

$$\inf_{u,\Gamma} \Big\{ F^{MS}(u,\Gamma) = \sum_i \int_{\Omega_i} |c_i - u_0|^2 dx + \nu|\Gamma| \Big\}, \tag{2.1}$$

where $\Omega = \cup_i \Omega_i \cup \Gamma$. They tried to find a decomposition Ω_i of Ω, where $u{=}c_i$ (constant) inside each connected component Ω_i. The positive parameter ν controls the length of the curve Γ. For a fixed Γ, we see that (2.1) is minimized when $c_i = mean(u_0)$ in Ω_i. A challenge when solving (2.1) is to find a unique representation of the parameterized curve Γ. To accommodate this problem, Chan and Vese [2] reformulated and solved (2.1) using a level set approach:

$$\inf_{\mathbf{c},\phi} \Big\{ F(c_1,c_2,\phi) = \int_\Omega |c_1{-}u_0|^2 H(\phi)\, dx + \int_\Omega |c_2{-}u_0|^2 (1{-}H(\phi))\, dx + \nu \int_\Omega |\nabla H(\phi)|\, dx \Big\}, \tag{2.2}$$

where the Heaviside function $H(\phi)$ equals 1 if $\phi \geq 0$ and equals 0 if $\phi < 0$. The regularization term $\int_\Omega |\nabla H(\phi)|\, dx$ measures the length of the interface [1] and favor

smooth curves. Minimizing $F(c_1, c_2, \phi)$ with respect to c_1 and c_2, we obtain: $c_1 = mean(u_0)$ if $\phi > 0$ and $c_2 = mean(u_0)$ if $\phi < 0$.

The active contour model without edges (2.2) can only divide Ω into two distinct regions. Later, Chan and Vese generalized this 2-phase active contour model to a multiphase level set formulation [5, 16]. By using a multiphase level set representation, it is possible to solve the "minimal" partition problem (2.1) for an arbitrary number of partitions. Below we show the energy functional for a 4-phase partition, involving 2 level set functions:

$$\begin{aligned} F(\mathbf{c}, \phi_1, \phi_2) = &\int_\Omega |c_1 - u_0|^2 H(\phi_1)H(\phi_2)\, dx + \int_\Omega |c_2 - u_0|^2 H(\phi_1)(1 - H(\phi_2))\, dx \\ &+ \int_\Omega |c_3 - u_0|^2 (1 - H(\phi_1))H(\phi_2)\, dx + \int_\Omega |c_4 - u_0|^2 (1 - H(\phi_1))(1 - H(\phi_2))\, dx \\ &+ \nu \int_\Omega |\nabla H(\phi_1)|\, dx + \nu \int_\Omega |\nabla H(\phi_2)|\, dx. \end{aligned} \tag{2.3}$$

Once the constant values $\{c_i\}_{i=1}^4$ and the level set functions ϕ_1, ϕ_2 are identified, the piecewise constant function u can be written as:

$$\begin{aligned} u = &c_1 H(\phi_1)H(\phi_2) + c_2 H(\phi_1)(1 - H(\phi_2)) + c_3 (1 - H(\phi_1))H(\phi_2) \\ &+ c_4 (1 - H(\phi_1))(1 - H(\phi_2)). \end{aligned} \tag{2.4}$$

Given u on this form, an alternative formulation of (2.3) was suggested in [4, 7]. Here the authors tried to minimize the following energy functional directly with u given as in (2.4):

$$F(\mathbf{c}, \phi_1, \phi_2) = \int_\Omega |u - u_0|^2 dx + \nu \int_\Omega |\nabla \phi_1| dx + \nu \int_\Omega |\nabla \phi_2| dx. \tag{2.5}$$

The above functional is essentially an application of the functional given in [4] for the image segmentation problem. Omitting the regularization term, functional (2.3) equals functional (2.5) except from one scalar. However, the gradients are not equal, producing different numerical schemes. Tests in [7] indicate somewhat better performance using the gradient associated with functional (2.5). Using the approaches above, two level set functions must be utilized to find four disjunct regions covering Ω. Correspondingly, we need to identify n functions and 2^n constants in order to identify the regions for 2^n phases.

Using our new approach as shown in (1.3), we just need one function to identify all the phases. Associated with ϕ, we define the basis functions ψ_i as:

$$\psi_i = \frac{1}{\alpha_i} \prod_{\substack{j=1 \\ j \neq i}}^{n} (\phi - j) \quad \text{and} \quad \alpha_i = \prod_{\substack{k=1 \\ k \neq i}}^{n} (i - k). \tag{2.6}$$

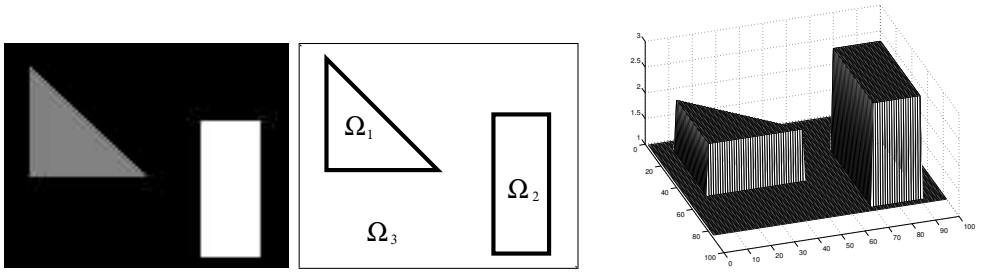
It is clear that a function u given by:

$$u = \sum_{i=1}^{n} c_i \psi_i \tag{2.7}$$

is a piecewise constant function and $u = c_i$ in Ω_i if ϕ is as given in (1.3). The sum in u involves basis functions of polynomial functions of order $n-1$ in ϕ and the unknown coefficients c_i, $i = 1, 2, \cdots n$. Each ψ_i is expressed as a product of linear factors of the form $(\phi - j)$, with the ith factor omitted. Thereupon $\psi_i(\mathbf{x})$=1 for $\mathbf{x} \in \Omega_i$, and $\psi_i(\mathbf{x})$ equals zeros elsewhere as long as (1.3) holds. The basis functions can further be utilized to express arc length of $\partial\Omega_i$ and the area Ω_i of each phase, i.e.

$$|\partial\Omega_i| = \int_\Omega |\nabla\psi_i| \, dx \text{ and } |\Omega_i| = \int_\Omega \psi_i \, dx. \tag{2.8}$$

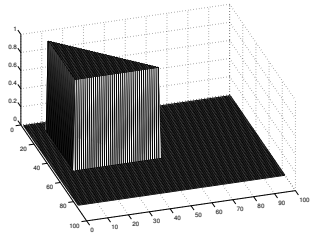
In Fig. 2.1 we demonstrate the relationship between an image function u, the partition of Ω associated with this image, the piecewise constant level set function ϕ and the corresponding basis functions ψ_i, $i = 1, 2, 3$. To ensure that equation (2.7)



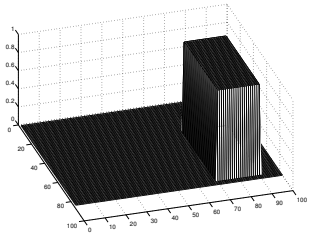
(a) Image u with 3 objects.

(b) Partition $\Omega = \cup_{i=1}^{3}\Omega_i$.

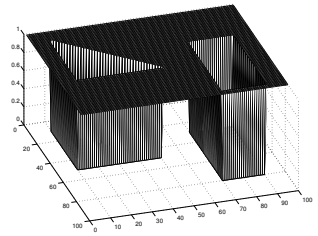
(c) Level set function $\phi = 1 \vee 2 \vee 3$.



(d) Basis function ψ_1. $\psi_1 = 0$ or 1.



(e) Basis function ψ_2. $\psi_2 = 0$ or 1.



(f) Basis function ψ_3. $\psi_3 = 0$ or 1.

FIG. 2.1. *The relationship between an image function $u \in BV(\Omega)$, the partition of Ω associated with the image, the piecewise constant level set function ϕ and the corresponding basis functions ψ_i, $i = 1, 2, 3$.*

gives us an unique representation of u, i.e. at convergence different values of ϕ should correspond to different function values $u(\phi)$ in (2.7), we introduce:

$$K(\phi) = (\phi - 1)(\phi - 2)\cdots(\phi - n) = \prod_{i=1}^{n}(\phi - i). \tag{2.9}$$

If a given function $\phi : \Omega \mapsto R$ satisfies

$$K(\phi) = 0, \tag{2.10}$$

there exists a unique $i \in \{1, 2, \cdots, n\}$ for every $x \in \Omega$ such that $\phi(x) = i$. Thus, each point $x \in \Omega$ can belong to one and only one phase if $K(\phi) = 0$. Thus constraint (2.10) is used to guarantee that there is no vacuum and overlap between the different phases. In [17] some other constraints for the classical level set methods was used to avoid vacuum and overlap.

Based on the above observations, we propose to solve the following constrained minimization problem for segmenting an image u_0:

$$\min_{\substack{\mathbf{c},\,\phi \\ K(\phi)=0}} \Big\{ F(\mathbf{c}, \phi) = \int_\Omega |u - u_0|^2 dx + \beta \sum_{i=1}^{n} \int_\Omega |\nabla \psi_i| dx \Big\}. \tag{2.11}$$

We see that large approximation errors will be penalized by the fidelity term $\int_\Omega |u - u_0|^2$. From (2.8), it is clear that the penalization term is just the sum of the length of the subdomain boundaries and the penalization parameter $\beta > 0$ controls the effect of these terms. Another approach could be to regularize ϕ directly in (2.11). This would make the algorithm simpler, but the total variation norm of ϕ is related to both the jumps and the length of the subdomain boundaries. Such a penalization will depend on the jumps which is not preferable for our case, see Fig. 2.1(c). We just want to penalize the length of the curves. If the image u_0 is a piecewise constant function and we take $\beta = 0$, then any minimizer of (2.11) will give a function u such that $u = u_0$ where u is related to the minimizers $\mathbf{c}$ and ϕ as in (2.7).

3. Minimization using augmented Lagrangian Method. We shall use the augmented Lagrangian method to solve the constrained minimization problem (2.11). The augmented Lagrangian functional for this minimization problem is defined as:

$$L(\mathbf{c}, \phi, \lambda) = F(\mathbf{c}, \phi) + \int_\Omega \lambda K(\phi)\, dx + \frac{r}{2} \int_\Omega |K(\phi)|^2 dx, \tag{3.1}$$

where $\lambda \in L^2(\Omega)$ is the multiplier and $r > 0$ is a penalty parameter which needs to be chosen properly. To find a minimizer for (2.11), we need to find the saddle points for L. We use the following Uzawa type algorithm to find a saddle point for $L(\mathbf{c}, \phi, \lambda)$:

Algorithm 1. Choose initial values for ϕ^0 and λ^0. For $k = 1, 2, \cdots$, do:

1. Find $\mathbf{c}^k$ from

$$L(\mathbf{c}^k, \phi^{k-1}, \lambda^{k-1}) = \min_{\mathbf{c}} L(\mathbf{c}, \phi^{k-1}, \lambda^{k-1}). \tag{3.2}$$

2. Use (2.7) to update $u = \sum_{i=1}^{n} c_i^k \psi_i(\phi^{k-1})$.

3. Find ϕ^k from

$$L(\mathbf{c}^k, \phi^k, \lambda^{k-1}) = \min_{\phi} L(\mathbf{c}^k, \phi, \lambda^{k-1}). \tag{3.3}$$

4. Use (2.7) to update $u = \sum_{i=1}^{n} c_i^k \psi_i(\phi^k)$.

5. Update the Lagrange-multiplier by

$$\lambda^k = \lambda^{k-1} + rK(\phi^k). \tag{3.4}$$

This algorithm has a linear convergence and its convergence has been analyzed in Kunisch and Tai [8] under a slightly different context. This algorithm has also been used in Chan and Tai [4, 3] for a level set method for elliptic inverse problems.

There are different methods to solve minimization problem (3.3). We have used the gradient method with a fixed step size to solve it. For a given old value of ϕ, we get a new ϕ by:

$$\phi^{new} = \phi^{old} - \Delta t \frac{\partial L}{\partial \phi}(\mathbf{c}^k, \phi^{old}, \lambda^{k-1}). \tag{3.5}$$

The step size Δt is chosen by a trial and error approach and it is fixed during the whole iterative procedure. It is not necessary to solve the minimization problem (3.3) exactly. We terminate the gradient iteration (3.5) when

$$\left\| \frac{\partial L}{\partial \phi}(\mathbf{c}^k, \phi^{new}, \lambda^{k-1}) \right\|_{L^2} \leq \frac{1}{10} \left\| \frac{\partial L}{\partial \phi}(\mathbf{c}^k, \phi^{k-1}, \lambda^{k-1}) \right\|_{L^2} \tag{3.6}$$

is reached or else after each 400 iterations, whereupon we take $\phi^k = \phi^{new}$. To compute $\frac{dL}{d\phi}$ we utilize the chain rule to get:

$$\frac{\partial L}{\partial \phi} = (u - u_0)\frac{\partial u}{\partial \phi} - \beta \sum_{i=1}^{n} \nabla \cdot \Big(\frac{\nabla \psi_i}{|\nabla \psi_i|} \Big) \frac{\partial \psi_i}{\partial \phi} + \lambda \frac{\partial K}{\partial \phi} + rK\frac{\partial K}{\partial \phi}. \tag{3.7}$$

It is easy to get $\partial u/\partial \phi$, $\partial \psi_i/\partial \phi$ and $\partial K/\partial \phi$ from (2.6), (2.7) and (2.9).

As u is linear with respect to the c_i values, we see that L is quadratic with respect to c_i. Thus the minimization problem (3.2) can be solved exactly. Note that

$$\frac{\partial L}{\partial c_i} = \int_\Omega \frac{\partial L}{\partial u}\frac{\partial u}{\partial c_i} = \int_\Omega (u - u_0)\psi_i \, dx, \qquad \text{for } \; i = 1, 2, \cdots n. \tag{3.8}$$

Therefore, the minimizer of (3.2) satisfies a linear system of equations $A\mathbf{c}^k = b$:

$$\sum_{j=1}^{n} \int_\Omega (\psi_j \psi_i) c_i^k \, dx = \int_\Omega u_0 \psi_i \, dx, \qquad \text{for } \; i = 1, 2, \cdots n. \tag{3.9}$$

In the above $\psi_j = \psi_j(\phi^k)$, $\psi_i = \psi_i(\phi^k)$ and thus, $\mathbf{c}^k = \{c_i^k\}_{i=1}^n$ depends on ϕ^k. In our simulations, we form the matrix A and vector b and solve the equation $A\mathbf{c}^k = b$ by an exact solver.

Remark 1. *The updating for the constant values in (3.2) is very ill-posed. A small perturbation of the ϕ function produces a large perturbation for the $\mathbf{c}$ values. Due to this reason, we have tried out a variant of Algorithm 1. In each iteration we alternate between (3.3) and (3.4), while (3.2) is only carried out if $\|K(\phi^{new})\|_{L^2} < \frac{1}{10}\|K(\phi^{old})\|_{L^2}$. Here, ϕ^{old} denotes the value of ϕ when (3.2) was carried out the last time and ϕ^{new} denotes the current value of ϕ. If we use such a strategy, we can do just one or a few iterations for the gradient scheme (3.5) and Algorithm 1 is still convergent. This strategy is particular efficient when the amount of noisy is high.*

Remark 2. *In Algorithm 1, we give initial values for ϕ and λ. We first minimize with the constant values, and then minimize with the level set function. The multiplier is updated in the end of each iteration. In situations where good initial values for $\mathbf{c}$ are available, an alternative variant of Algorithm 1 may be used, i.e. we first minimize with the level set function followed by an minimization for the constant values and then update the multiplier.*

4. Strength and Weakness of our Level Set Formulation. In this section we will point out the main differences between a classical level set formulation and the variant we have proposed here. First, while multiple level set methods [5, 16] utilize n level set functions to represent 2^n distinct phases, our alternative level set formulation let us represent the same number of phases using only one level set function. This gains the storage capacity.

Further, when minimizing (2.3), the associated Euler-Lagrange equations are [5, 16]:

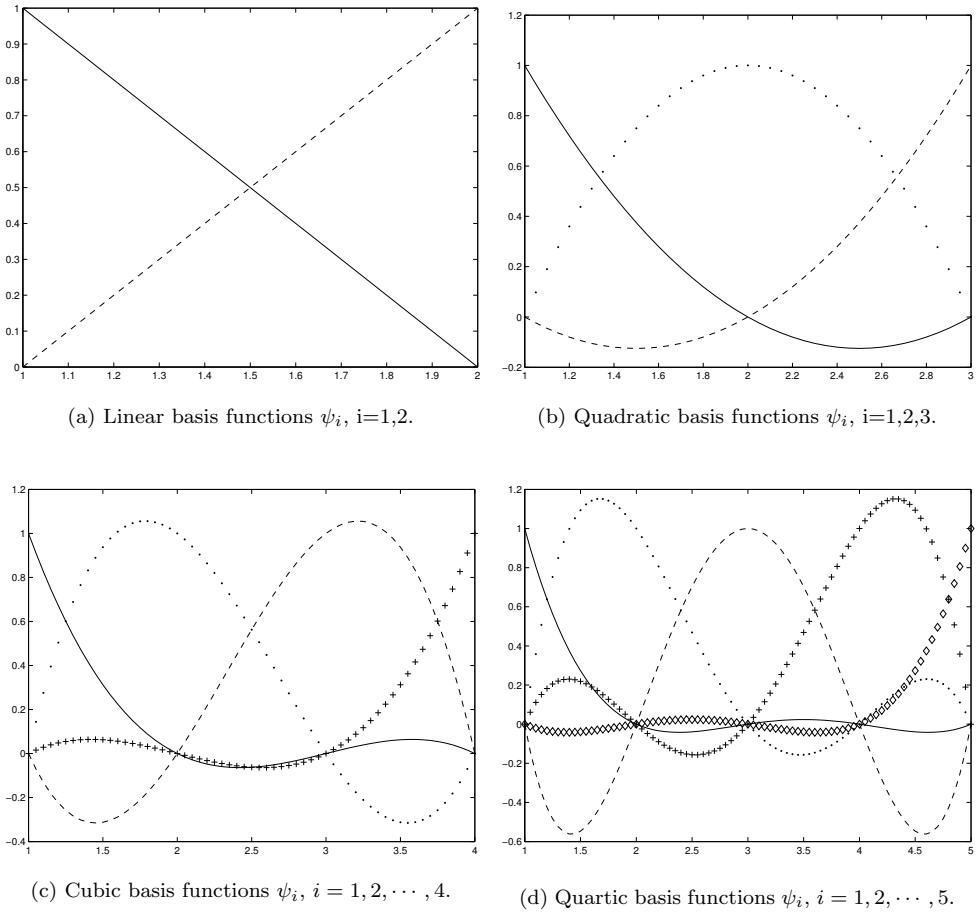
$$\begin{aligned}
\frac{\partial \phi_1}{\partial t} &= \delta(\phi_1)\Big\{\beta\ \mathrm{div}\Big(\frac{\nabla \phi_1}{|\nabla \phi_1|}\Big) \\
&\quad - \Big[\Big((u_0-c_1)^2-(u_0-c_3)^2\Big)H(\phi_2)+\Big((u_0-c_2)^2-(u_0-c_4)^2\Big)(1-H(\phi_2))\Big]\Big\}, \\
\frac{\partial \phi_2}{\partial t} &= \delta(\phi_2)\Big\{\beta\ \mathrm{div}\Big(\frac{\nabla \phi_2}{|\nabla \phi_2|}\Big) \\
&\quad - \Big[\Big((u_0-c_1)^2-(u_0-c_2)^2\Big)H(\phi_1)+\Big((u_0-c_3)^2-(u_0-c_4)^2\Big)(1-H(\phi_1))\Big]\Big\},
\end{aligned} \tag{4.1}$$

where both the Heaviside function $H(\phi)$ and the Delta function $\delta(\phi)$ are involved. Normally, these functions have to be regularized, see for example [2] for different approximations for $H_\epsilon(\phi)$ and $\delta_\epsilon(\phi)$. Without these approximations the level set functions ϕ_i, i=1,2, do not evolve properly, and converge to a local minimum. However, some numerical error do occur when approximations are used for $H_\epsilon(\phi)$ and $\delta_\epsilon(\phi)$. For small ϵ, the Delta function $\delta_\epsilon(\phi)$ is a function with sharp singular layers around the zero level curve of ϕ. We do not use such approximations in our formulation. The energy function for our approach is smooth and convex. The convergence for the Uzawa algorithm can be analyzed and it has a linear convergence rate.

Also, ϕ is normally initialized as the signed distance function like in (1.1). However, as the interface evolves, ϕ will generally drift away from its initialized value. Moreover, ϕ may develop noisy features and steep gradients that are not amenable to finite-difference approximations. For this reason, it is advisable to reinitialize the level set function, i.e. force ϕ to satisfy the partial differential equation: $|\nabla \phi| = 1$. The reinitializing technique is superfluous in our formulation.

Advantages using classical level set methods are the ability to express geometrical quantities and motions. The length of the zero level curve Γ and the area A of the region inside Γ are measured by $|\Gamma| = \int_\Omega |\nabla H(\phi)|\ dx$ and $|A| = \int_\Omega |H(\phi)|\ dx$. Further, the unit normal $\mathbf{N}$ and the mean curvature κ to Γ are defined by $\mathbf{N}=\frac{\nabla \phi}{|\nabla \phi|}$ and $\kappa = \nabla \cdot \mathbf{N}$. Fo our approach, there are some difficulty to find a suitable way to represent the unit normal and the mean curvature for the curves.

Another shortcoming of our method is that higher order polynomials are used as basis functions to represent u in (2.7). Further, the same basis functions are also used to find $\mathbf{c}$ in (3.9). To illustrate this weakness, we depict the linear, quadratic, cubic and quartic basis functions in Fig. 4.1. In Fig. 4.1 we see that one, and only one basis function is different from zero at each root when $K(\phi) = 0$ is fulfilled. However, by inspecting the higher order basis functions ψ_i in Fig. 4.1(c) and (d) we see that even a small perturbation away from each root cause a significant change in the contribution from several of the basis functions. This sensitivity for perturbations makes the coefficient matrix in (3.9) ill-conditioned. To cure this problem we could search for



(a) Linear basis functions ψ_i, i=1,2.

(b) Quadratic basis functions ψ_i, i=1,2,3.

(c) Cubic basis functions ψ_i, $i = 1, 2, \cdots, 4$.

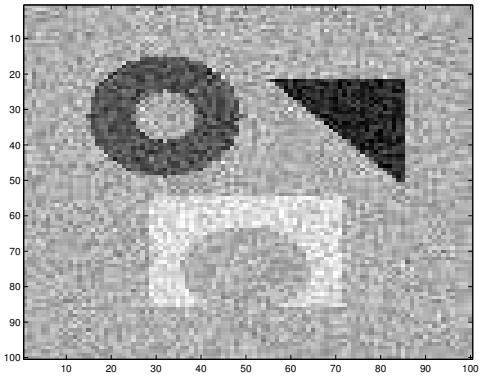
(d) Quartic basis functions ψ_i, $i = 1, 2, \cdots, 5$.

FIG. 4.1. *Different basis functions defined in (2.6).*

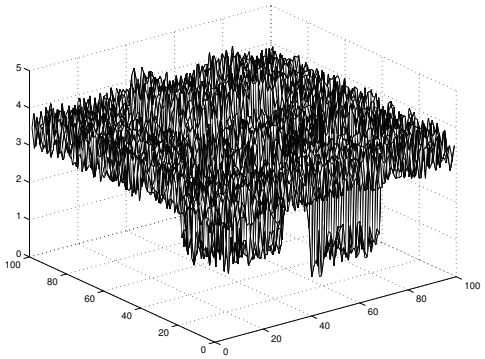
other basis functions with better orthogonality properties, but that is outside the scope of this paper.

To the end, we emphasize that numerical tests have shown that proper initial guess are needed using our approach. Fortunately we can use a simple strategy to approximate the constant coefficients c_i. Basically these constants are given by (3.9) if $\psi_i(\phi)$ is approximately known. Assume now that an observed signal u_0 should be segmented into n phases. From the constraint $K(\phi) = 0$ it follows that ϕ must take values in the set $\{1, 2, \cdots, n\}$. As an approximation for ϕ we let $\phi = u_0$ and thereafter rescale ϕ such that $1 \leq \phi(x, y) \leq n$ for $(x, y) \in \Omega$. Using this approach we have a way to initialize ϕ and find $\psi_i(\phi)$ to get good initial guess for the coefficients c_i, $i = 1, 2, \cdots, n$.

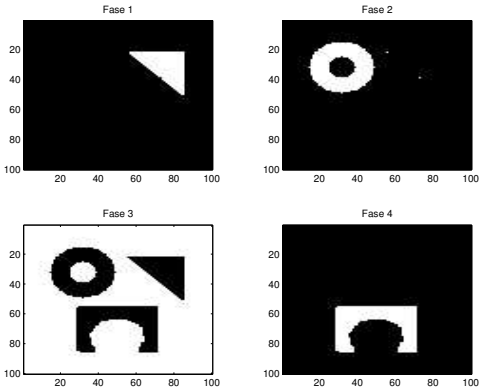
5. Numerical Results. In this section we validate the piecewise constant level set method (PCLSM) with numerical examples. We consider only two-dimensional cases and restrict ourself to gray-scale images, but the model can handle any dimension



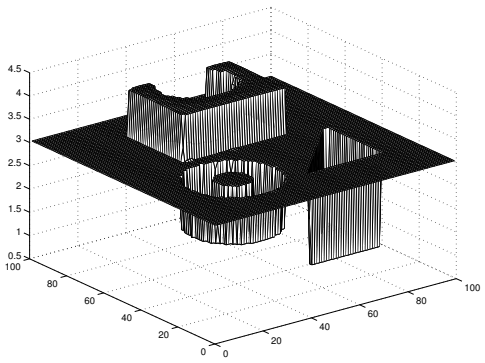
(a) Observed image u_0 (SNR$\approx$ 5.2).



(b) Initial level set function ϕ.



(c) Different phases using PCLSM.

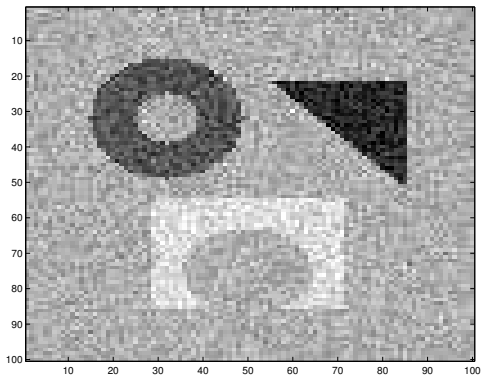


(d) At convergence ϕ approaches 4 constant values.

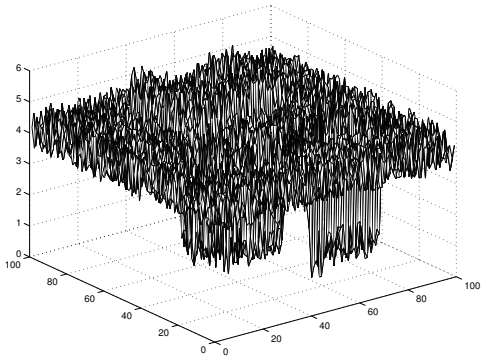
FIG. 5.1. *Each separate phase* $\phi = 1 \vee 2 \vee 3 \vee 4$ *are depicted as a bright region in Fig. 5.1(c).*

and can be extended to vector-valued images as well. Both synthesized images, natural scenes and a MR image are evaluated in this section. Our results will be compared with other related works [5, 16, 15]. The original image is known for some cases we evaluate here. Thereupon it is trivial to find the perfect segmentation result. To complicate such a segmentation process we typically expose the original image with Gaussian distributed noise and use the polluted image as the observation data u_0. To indicate the amount of noise that appears in the observation data, we report the signal-to-noise-ratio: SNR=$\frac{\text{variance of data}}{\text{variance of noise.}}$

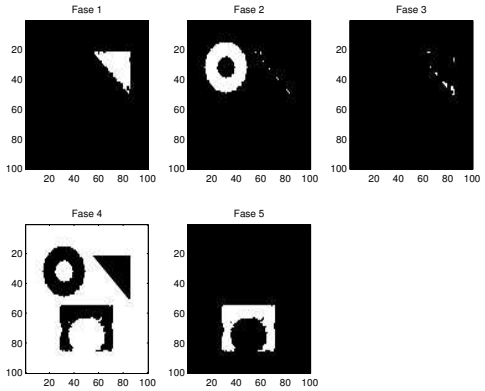
To demonstrate a 4-phase segmentation we begin with a noisy synthetic image containing 3 objects (and a background) as shown in Fig. 5.1(a). This is the same image as Chan and Vese used to examine their multiphase algorithm [5, 16]. A careful evaluation of our algorithm is reported below. The observation data u_0 is given in Fig. 5.1(a) and the only assumption we make is that a 4-phase model should be utilized to find the segments. In Fig. 5.1(d) the ϕ function is depicted at convergence (less



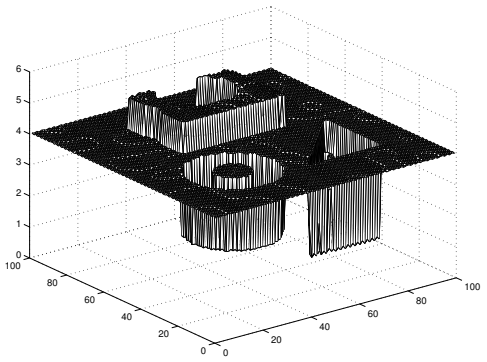
(a) Observed image u_0 (SNR$\approx$ 5.2).



(b) Initial level set function ϕ.



(c) Different phases using PCLSM.

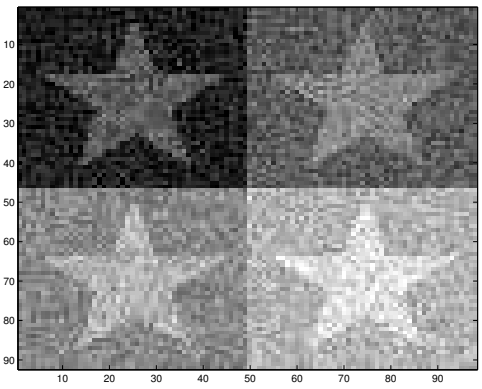


(d) At convergence ϕ approaches 4 constant values.

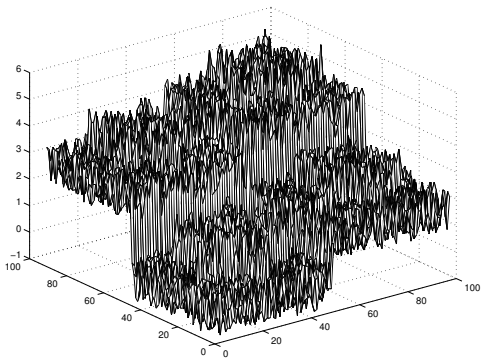
FIG. 5.2. *Each separate phase* $\phi = 1 \vee 2 \vee 3 \vee 4 \vee 5$ *are depicted as a bright region in Fig. 5.2(c).*

than 1000 iteration). ϕ approaches the predetermined constants $\phi = 1 \vee 2 \vee 3 \vee 4$. Each of these constants represents one unique phase as seen in Fig. 5.1(c). Our result is in accordance with what Chan and Vese report in [5, 16].

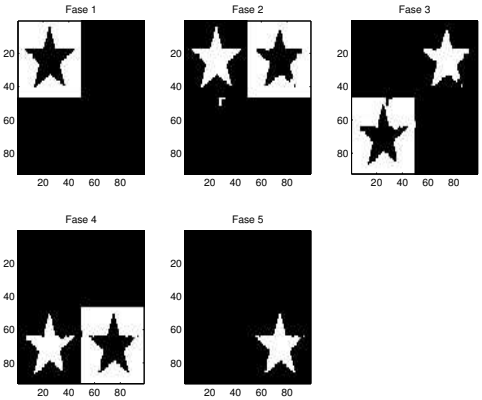
In many applications the number of objects to detect are not known a priori. A robust and reliable algorithm should find the correct segmentation even when the exact number of phases are not known. By introducing a model with more phases than one actually needs, we can find the correct segmentation if all superfluous phases are empty when the algorithm has converged. To see if our algorithm can handle such a case we use Fig. 5.2(a) as the observation image and utilize a 5-phase model. Our results are reported below. One of the 5 phases must be empty if a 5-phase model is used to find a 4-phase segmentation. Due to the high noise level some pixels can easily be misclassified and contribute to the phase that should be empty. The level set function shown in Fig. 5.2(d) approaches the constants $\phi = 1 \vee 2 \vee 4 \vee 5$, except from the few misclassified pixels where $\phi = 3$ as seen in Fig. 5.2(c). By comparing



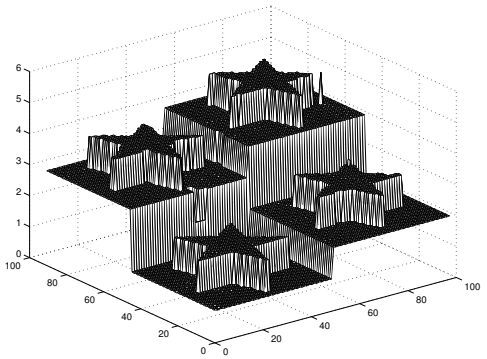
(a) Observed image u_0 (SNR$\approx$ 10.6).



(b) Initial level set function ϕ.



(c) Different phases using PCLSM.

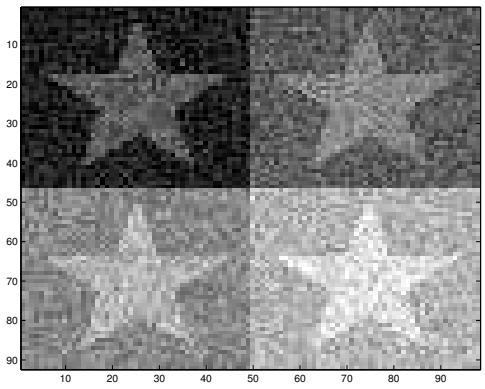


(d) At convergence ϕ approaches 5 constant values.

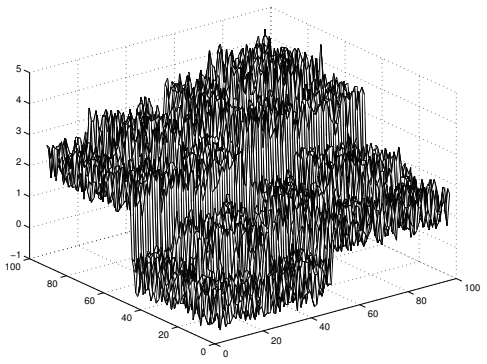
FIG. 5.3. *Each separate phase* $\phi = 1 \vee 2 \vee 3 \vee 4 \vee 5$ *are depicted as a bright region in Fig. 5.3(c).*

Fig. 5.1(c) (where a 4-phase model is used) and Fig. 5.2(c) (where a 5-phase model is used), we observe only small changes in the segmented phases, except from the extra nonempty phase $\phi = 3$ in Fig. 5.2(c). For a test like this, we can not control which phase that ends up empty.

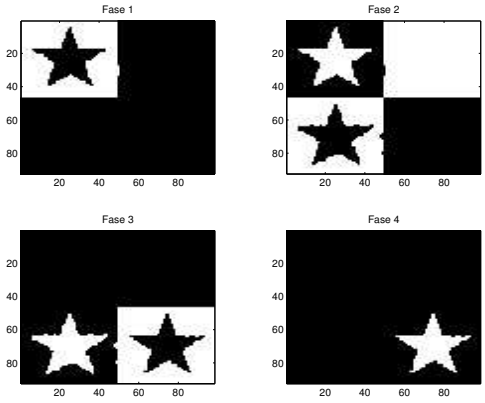
In the example below we use PCLSM to find a 5-phase segmentation. We use a noisy synthetic image containing 4 stars on 4 different backgrounds, as shown in Fig. 5.3(a). There are only 5 different phases in this image. The star in the upper left corner belongs to same phase as the background in the upper right corner. Further, the star in the upper right corner belongs to the same phase as the background in the lower left corner, and the star in this corner belongs to the same phase as the background in the lower right corner. In total that gives us 5 different phases. Almost no misclassification was observed in this tests. By using one level set function, we manage to find all 5 objects, as seen from Fig. 5.3(c). This is in contrast to classi-



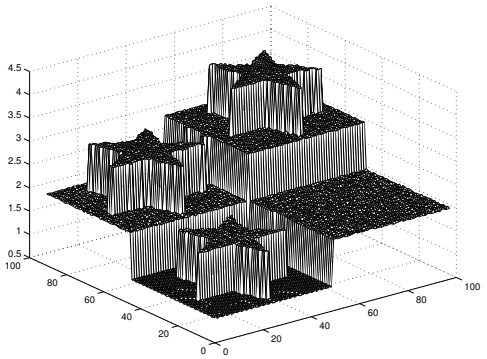
(a) Observed image u_0 (SNR$\approx$ 10.6).



(b) Initial level set function ϕ.



(c) Different phases using PCLSM.



(d) At convergence ϕ approaches 4 constant values.

FIG. 5.4. *Each separate phase* $\phi = 1 \vee 2 \vee 3 \vee 4$ *are depicted as a bright region in Fig. 5.4(c).*

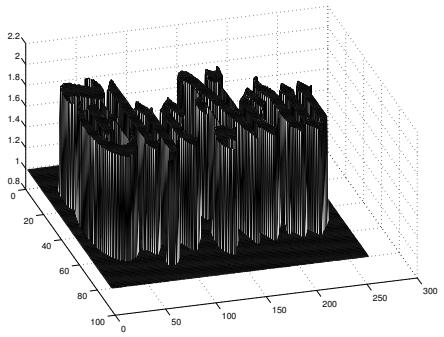
cal level set methods where 3 level set functions must be utilized to get the same result.

If an image actually consists of 5 objects, but a 4-phase model is utilized in the segmentation procedure, one may hope that 2 of the 5 objects merge into 1 new object without affecting the segmentation result for the other objects. Below we depict our result using Fig. 5.4(a) as the observation data in a 4-phase segmentation. By comparing Fig. 5.3(c) and Fig. 5.4(c) we see that phase $\phi = 2$ and phase $\phi = 3$ in Fig. 5.3(c) have merged into one phase $\phi = 2$ in Fig. 5.4(c). All other phases seem to be unaffected of this.

All the examples above indicate that our algorithm is an interesting alternative to the multiphase algorithm [5, 16] where standard level set formulation is utilized. We have shown that our algorithm is robust with respect to noise. As in [5, 16] we have shown that the exact number of phases do not have to be known a priori. Redundant phases can disappear in the sense that they will end up empty. Below we proceed with



(a) Old newspaper.



(b) A small partition of ϕ



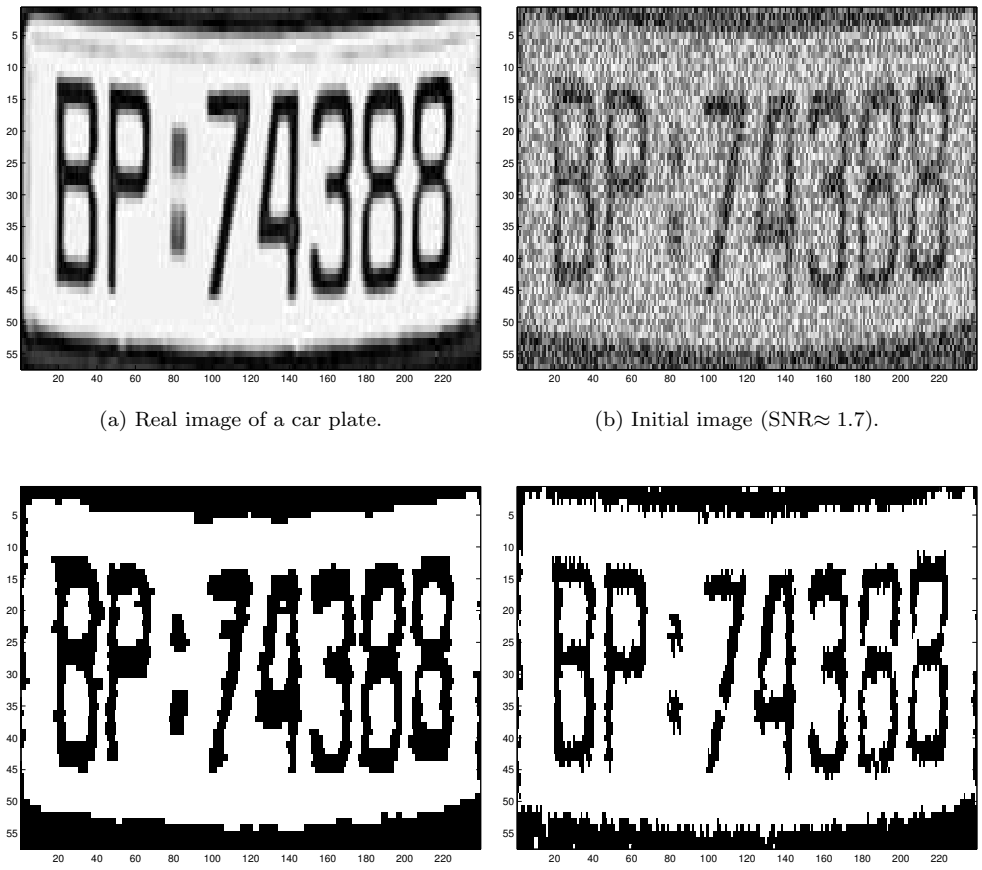
(c) Result using PCLSM.



(d) Result using CVM.

FIG. 5.5. *For the purpose of illustration only a small partition of the level set function ϕ is depicted in the upper right corner. We see that ϕ takes two constant values, as it should.*

some examples using real pictures (some natural scene and one MR image). First we demonstrate that PCLSM can be use to extract characters or numbers form images. We begin with an image of an old newspaper where only two phases are needed; one phase to represent the characters and one phase to represent the remaining. To evaluate the segmentation process, the Chan/Vese method [5, 16], for short (CVM), is examined using the same input image. The result achieved with the segmentation techniques PCLSM and CVM are shown in Fig. 5.5(c) and Fig. 5.5(d) respectively. Few differences between the segmented images were observed, but a careful inspection of all characters in Fig. 5.5(c) and Fig. 5.5(d) reveal that PCLSM preserve small details somewhat better than CVM. In this paper we will mainly focus on the segmentation process, but let us briefly describe a way to identify the characters separated by phase $\phi = 2$ given in Fig. 5.5(b). First one can isolate each character in this phase by a *Peak-and-valley method* [13]. Thereafter, to identify the isolated characters, the center of mass for each character is aligned to the center of mass for all character templates in a database. Thereafter each isolated character is registered to all tem-



(a) Real image of a car plate.

(b) Initial image (SNR$\approx$ 1.7).

(c) Segmented with PCLSM.

(d) Segmented with CVM.

FIG. 5.6. *Character and number segmentation from a car plate.*

plates. Finally the difference between each registered character and all templates are measured. The template corresponding to the smallest difference in L_2-norm is chosen to be the solution.

In the next example we illustrate a 2-phase segmentation on a real car plate image. Locating and reading car plates is a well known problem, and there are a number of commercial software available. Below we demonstrate that our level set method can be used in such an application. Since character and number recognitions are the main target in this example, we assume the location of the car plate to be known. A perfect segmentation was found with both PCLSM and CVM if Fig. 5.6(a) was used as the observation data. We challenge these two segmentation techniques by adding Gaussian distributed noise to the real image and use the polluted image in Fig. 5.6(b) as the observation data. With this amount of noise both PCLSM and CVM miss some details along the edges for the characters and numbers. By increasing the regularization term for CVM we obtain smoother edges but then each character or number

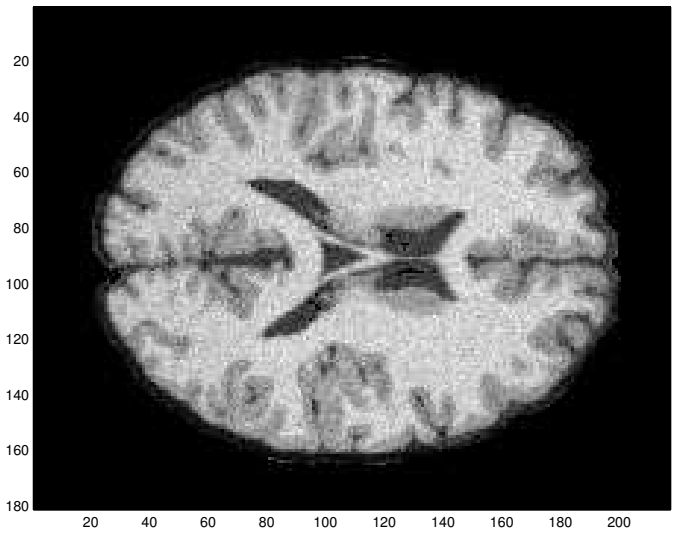
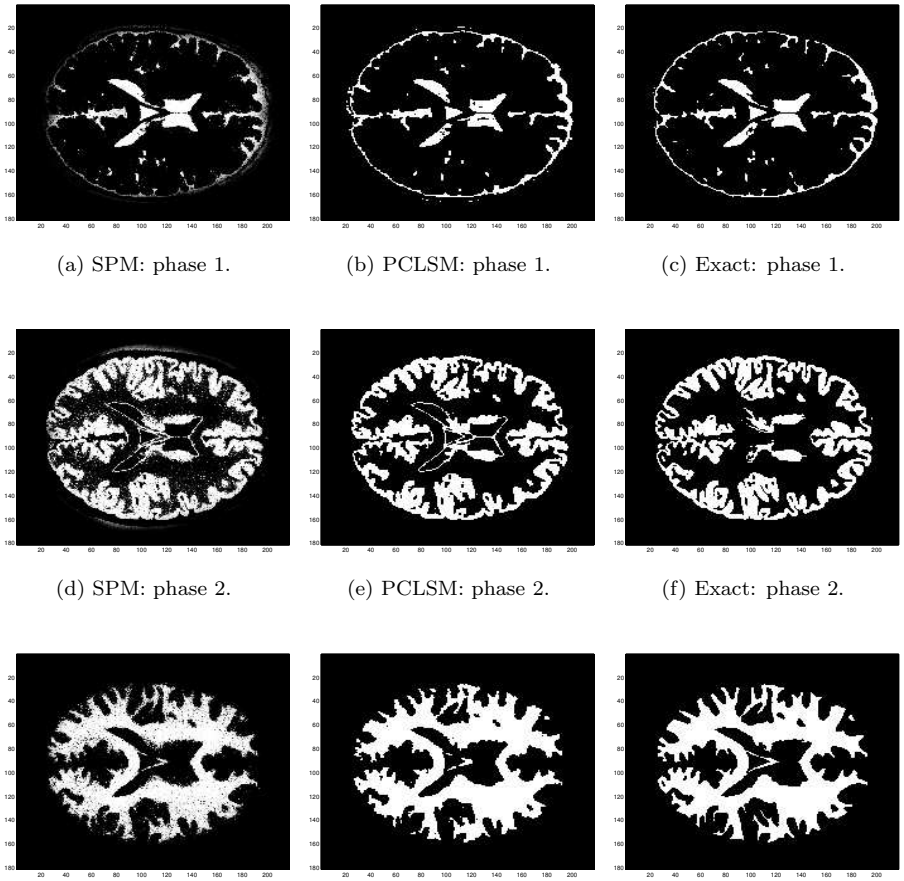


FIG. 5.7. *MRI image with a change in the intensity values going from left to right caused by the non-uniformity RF-puls.*

may be broken into several pieces. From Fig. 5.6(c) we see that each character and number appear as unbroken.

In our next example segmentation of a MR image is demonstrated. The image in Fig. 5.7 is available to the public at *http://www.bic.mni.mcgill.ca/brainweb/*. These realistic MRI data are used by the neuroimaging community to evaluate the performance of various image analysis methods in a setting where the truth is known. For the image used in this test the noise level is 7% and the non-uniformity intensity level of the RF-puls is 20%, see *http://www.bic.mni.mcgill.ca/brainweb/* for details concerning the noise level percent and the intensity level of the RF-puls. Both PCLSM and SPM [15] are used to segment the above MRI phantom and the results are depicted in Fig. 5.8. SPM is a statistical segmentation software program to process MRI data for research and clinical use. In Fig. 5.8 there are three tissue classes that should be identified; phase 1: cerebrospinal fluid, phase 2: gray matter, phase 3: white matter. Because of this, four phases were used, but we do not depict the background phase here. The main difference between PCLSM and SPM is their ability to handle noisy data. SPM is adjusted for MR data, but each phase obtained with SPM is perforated with misclassified pixels. The same problem do not occur with PCLSM.

As a final example we consider a scattered data problem. Here the goal is to connect single observation points and form continuous objects. The lights in a satellite image of Europe by night act as scattered data points. The intensity from each light source are the same, but the density of data points fluctuate within the satellite image. Generally it is hard to find features by connecting single observation points. Models based on edge stopping functions will fail completely for applications like this. We do not reckon our model as well suited for scattered data problems, but by following the proposition from remark 1 in section 3 we managed to get the result in Fig. 5.9(c). This result is comparable with what Chan and Vese report in [2] for the same satellite image. In Fig. 5.9(b)-(d) we demonstrate the effect of the total variation norm for ψ_i



(a) SPM: phase 1. (b) PCLSM: phase 1. (c) Exact: phase 1.

(d) SPM: phase 2. (e) PCLSM: phase 2. (f) Exact: phase 2.

(g) SPM: phase 3. (h) PCLSM: phase 3. (i) Exact: phase 3.

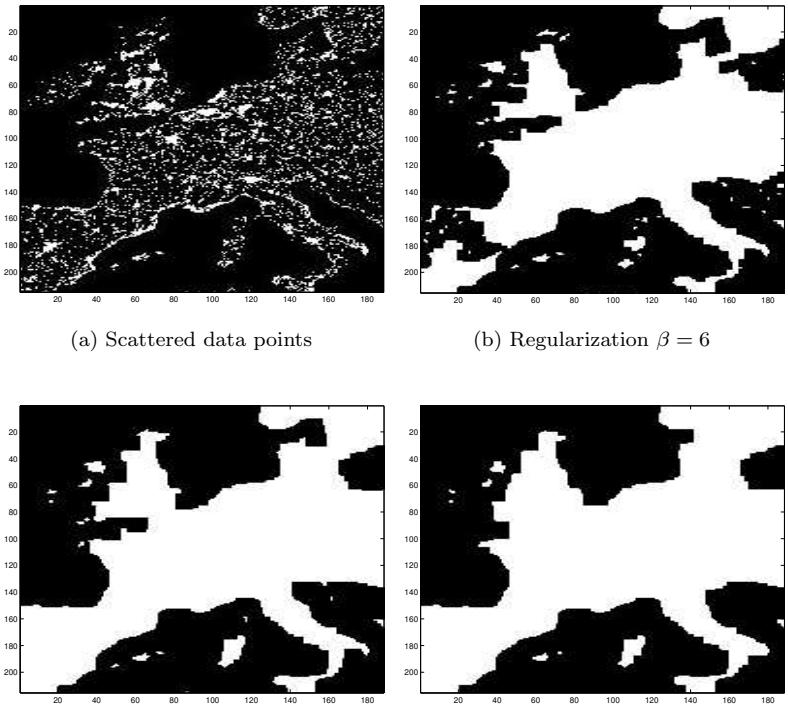
FIG. 5.8. *SPM is adjusted for MR data, but it seems hard for SPM to handle to noise properly. All phases found by PCLSM match the exact phases in a better way than SPM do.*

in (2.11). Weighting these terms too softly causes an image with perforated regions as seen in Fig. 5.9(b). On the other hand, if these terms are weighted too hard all sparse regions disappear, as seen in Fig. 5.9(d).

Acknowledgment. The authors thank Stanley Osher for valuable comments on a draft of this paper and for bring our attention to references [9, 6] before the submission of this work.

REFERENCES

[1] E. L. C. AND G. R. F., *Measure therory and fine properties of functions*, 1992.
[2] T. CHAN AND L. A. VESE, *Active contours without edges*, IEEE Image Proc., 10 (2001), pp. 266–277.
[3] T. F. CHAN AND X.-C. TAI, *Identification of discontinuous coefficients in elliptic problems using total variation regularization*, SIAM J. Sci. Comput., (Accepted and to appear). Available online athttp://www.mi.uib.no/˜tai/.



(a) Scattered data points (b) Regularization $\beta = 6$

(c) Regularization $\beta = 7$ (d) Regularization $\beta = 9$

FIG. 5.9. *Scattered data are connected to form an object.*

[4] T. F. CHAN AND X.-C. TAI, *Level set and total variation regularization for elliptic inverse problems with discontinuous coefficients*, J. Comput. Physics, (to appear).

[5] T. F. CHAN AND L. A. VESE, *Image segmentation using level sets and the piecewise constant mumford-shah model*, tech. rep., CAM Report 00-14, UCLA, Math. Depart., April 2000. revised December 2000.

[6] S. CHEN, B. MERRIMAN, M. KANG, R. E. CAFILSCH, C. RATSCH, L.-T. CHENG, M. GYURE, R. P. FEDKIW, AND S. OSHER, *Level set method for thin film epitaxial growth*, tech. rep., CAM 00-03, UCLA Math. Depart., 2000.

[7] E. HODNELAND, *Segmentation of digital images*, cand. scient thesis, Department of Mathematics, University of Bergen, 2003. Available online at http://www.mi.uib.no/%7Etai.

[8] K. KUNISCH AND X.-C. TAI, *Sequential and parallel splitting methods for bilinear control problems in Hilbert spaces*, SIAM J. Numer. Anal., 34 (1997), pp. 91–118.

[9] B. MERRIMAN, R. CAFLISCH, S. OSHER, C. RATSCH, S. CHEN, M. KANG, AND M. GYURE, *Island dynamics and level set methods for continuum modeling of epitaxial growth*, in Applied and industrial mathematics, Venice–2, 1998, Kluwer Acad. Publ., Dordrecht, 2000, pp. 145–171.

[10] D. MUMFORD AND J. SHAH, *Optimal approximation by piecewise smooth functions and assosiated variational problems*, Comm. Pure Appl. Math, 42 (1989), pp. 577–685.

[11] S. OSHER AND J. A. SETHIAN, *Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations*, J. Comput. Phys., 79 (1988), pp. 12–49.

[12] A. G. SAMSON C., BLANC-FÉRAUD L. AND Z. J., *A level set model for image classification*, IJCV, 40 (2000), pp. 187–197.

[13] C. J. SETCHELL, *Applications of computer vision to road-traffic monitoring*, phd thesis, Department of Computer Science, University of Bristol, 1997. Available online at http://www.cs.bris.ac.uk/Tools/Reports/Abstracts/1997-setchell.html.

[14] B. SONG AND T. CHAN, *A fast algorithm for level set based optimization*, tech. rep., CAM 0268, UCLA Math. Depart., 2002.

[15] SPM, *http://afni.nimh.nih.gov/afni/*.
[16] L. A. Vese and T. F. Chan, *A new multiphase level set framework for image segmentation via the Mumford and Shah model*, International Journal of Computer Vision, 50 (2002.), pp. 271–293.
[17] H.-K. Zhao, T. Chan, B. Merriman, and S. Osher, *A variational level set approach to multiphase motion*, J. Comput. Phys., 127 (1996), pp. 179–195.

Appendix B

A Binary Level Set Model and Some Applications to Image Processing

Johan Lie, Marius Lysaker and Xue-Cheng Tai

December 8, 2003

Abstract

In this work we propose a variant of a PDE based level set method. Traditionally interfaces are represented by the zero level 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 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 convex. We show numerical results using the method for segmentation of digital images.

1 Introduction

The level set method introduced by Osher and Sethian [1] is a versatile tool for tracing interfaces separating a domain Ω into sub domains. Interfaces are treated as the zero level of level set 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, 3, 4, 5, 6]. For a recent survey on the level set methods see [7, 8, 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 §2. 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 type of iteration to get 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 sub domains 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]. See also [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 §2. Following the Mumford/Shah formulation for image segmentation, Chan and Vese solved the minimization problem by using level set functions [2, 10]. The interface Γ is traced by the level set functions. Motivated by the Chan-Vese approach [2, 10], we will in this article solve the segmentation problem by introducing a piecewise constant level set function. Instead of using the zero level of a level set function to represent the interface between sub domains, 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 sub domains, 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 to represent multiple phases.

The rest of this article is structured as follows. In §2 we give a brief review of the traditional level set method. Our new level set model is formulated in §3. In §4, we apply this model for digital image segmentation. The segmentation problem is formulated as a minimization 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 §5.

2 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 following properties:

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

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:

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

where $d(x,\Gamma)$ denotes Euclidian distance between x and Γ. 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:

$$|\nabla\phi| = 1. \tag{3}$$

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:

$$\phi_t + sgn(\tilde{\phi})(|\nabla\phi| - 1) = 0 \tag{4}$$

$$\psi(x,0) = \tilde{\phi}(x). \tag{5}$$

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, and is illustrated by a simple one dimensional signal in Figure 1.

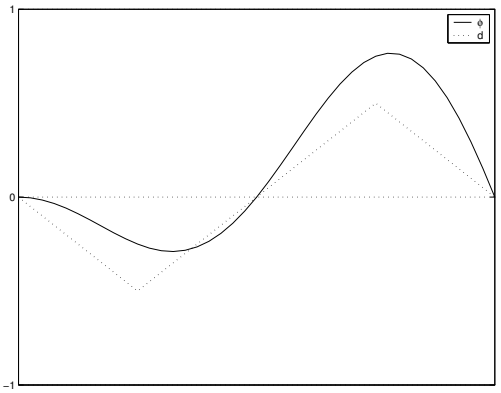


Figure 1: The reinitialization procedure replaces ϕ with a distance function d having the same zeros as ϕ.

Note that the zero level set is unaffected by solving (4). For numerical computations the reinitialization procedure is crucial, and there exists many numerical finite difference schemes for computing the distance function. See [3, 7, 8, 15] for some details. The interface $\Gamma(t)$ is implicitly moved according to the nonlinear PDE:

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

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:

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

2.1 Level Set Method 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:

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

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 proposed the following minimization problem for a two phase segmentation:

$$\min_{c_1,\,c_2,\,\phi} \left\{ \int_\Omega |u_0 - c_1|^2 H(\phi) dx + \int_\Omega |u_0 - c_2|^2 (1 - H(\phi)) dx + \nu \int_\Omega H(\phi) dx + \mu \int_\Omega \delta(\phi) |\nabla \phi| dx \right\}. \tag{9}$$

Here ϕ is the level set function satisfying (1) and $H(\phi)$ is the Heaviside function:

$$H(\phi) = \begin{cases} 1, & \phi \geq 0, \\ 0, & \phi < 0. \end{cases} \tag{10}$$

Finding a minimum of (9) is done by introducing an artificial time variable, and moving ϕ in the steepest descent direction to steady state:

$$\phi_t = \delta_\epsilon(\phi) \left(\mu \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right) - \nu - (u_0 - c_1)^2 + (u_0 - c_2)^2 \right), \tag{11}$$

$$\phi(\boldsymbol{x}, 0) = \phi_0(\boldsymbol{x}). \tag{12}$$

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 (11) 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:

$$\begin{aligned} F(\boldsymbol{c}, \phi_1, \phi_2) &= \int_\Omega |c_1 - u_0|^2 H(\phi_1) H(\phi_2) dx \\ &+ \int_\Omega |c_2 - u_0|^2 H(\phi_1)(1 - H(\phi_2)) dx \\ &+ \int_\Omega |c_3 - u_0|^2 (1 - H(\phi_1)) H(\phi_2) dx \\ &+ \int_\Omega |c_4 - u_0|^2 (1 - H(\phi_1))(1 - H(\phi_2)) dx \end{aligned} \tag{13}$$

$$
\begin{aligned}
&+ \quad \nu \left(\int_{\Omega} H(\phi_1) dx + \int H(\phi_2) dx \right) \\
&+ \quad \mu \left(\int_{\Omega} \delta(\phi_1) |\nabla \phi_1| dx + \int \delta(\phi_2) |\nabla \phi_2| dx \right).
\end{aligned}
$$

Having determined $\boldsymbol{c} = \{c_i\}_{i=1}^4$, ϕ_1 and ϕ_2 by the minimization of (13), four different regions can be identified by the sign of the two level set functions such that:

$$
u(\boldsymbol{x}) = \begin{cases} c_1, & \text{if } \phi_1(\boldsymbol{x}) > 0 \ , \ \phi_2(\boldsymbol{x}) > 0, \\ c_2, & \text{if } \phi_1(\boldsymbol{x}) > 0 \ , \ \phi_2(\boldsymbol{x}) < 0, \\ c_3, & \text{if } \phi_1(\boldsymbol{x}) < 0 \ , \ \phi_2(\boldsymbol{x}) > 0, \\ c_4, & \text{if } \phi_1(\boldsymbol{x}) < 0 \ , \ \phi_2(\boldsymbol{x}) < 0. \end{cases} \tag{14}
$$

By utilizing (10), the recovered cartoon image u consisting of four phases can be written as:

$$
\begin{aligned}
u &= c_1 H(\phi_1) H(\phi_2) + c_2 H(\phi_1)(1 - H(\phi_2)) \\
&+ c_3 (1 - H(\phi_1)) H(\phi_2) + c_4 (1 - H(\phi_1))(1 - H(\phi_2)).
\end{aligned} \tag{15}
$$

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 ride 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 non-differentiability with the Heaviside and Delta functions is avoided.

3 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:

$$
\phi(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \ \text{int}(\Omega_1) \ , \\ -1 & \text{if } \boldsymbol{x} \ \text{ext}(\Omega_1) \ . \end{cases} \tag{16}
$$

Γ is implicitly defined as the discontinuity of ϕ. This representation can be used for different applications where sub domains need to be identified. In order to use this idea for image segmentation, we use (16) 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:

$$
u = \frac{c_1}{2}(\phi + 1) - \frac{c_2}{2}(\phi - 1). \tag{17}
$$

The formula (17) 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

$$\begin{aligned} u = &\frac{c_1}{4}(\phi_1+1)(\phi_2+1) - \frac{c_2}{4}(\phi_1+1)(\phi_2-1) \\ &-\frac{c_3}{4}(\phi_1-1)(\phi_2+1) + \frac{c_4}{4}(\phi_1-1)(\phi_2-1) \end{aligned} \tag{18}$$

is a piecewise constant function of the form:

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

Introducing basis functions ψ_i as in the following

$$u = c_1 \underbrace{\frac{1}{4}(\phi_1+1)(\phi_2+1)}_{\psi_1} + c_2 \underbrace{(-1)\frac{1}{4}(\phi_1+1)(\phi_2-1)}_{\psi_2} + \cdots \tag{19}$$

we see that u can be written as:

$$u = \sum_{i=1}^{4} c_i \psi_i. \tag{20}$$

For more general cases, we could use N level set functions and this will produce 2^N number of phases. To simplify notation, we form the vectors $\boldsymbol{\phi} = \{\phi_1, \phi_2, \cdots, \phi_N\}$ and $\boldsymbol{c} = \{c_1, c_2, \cdots, c_{2^N}\}$. Let $b(j)$ be the number at the j^{th} position of the binary representation of $i-1$. Furthermore, set

$$s(i) = \sum_{j=1}^{N} b(i-1), \tag{21}$$

and write ψ_i as the product:

$$\psi_i = \frac{(-1)^{s(i)}}{2^N} \prod_{j=1}^{N} (\phi_j + (1 - 2b(j))). \tag{22}$$

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

$$u = \sum_{i=1}^{2^N} c_i \psi_i. \tag{23}$$

If the level set functions ϕ_i satisfy $\phi_i^2 = 1$ and ψ_i are defined as in (20) or (22), 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.

$$|\partial\Omega_i| = \int_\Omega |\nabla \psi_i| dx, \qquad \text{and} \qquad |\Omega_i| = \int_\Omega \psi_i dx. \tag{24}$$

4 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:

$$F(\boldsymbol{\phi}, \boldsymbol{c}) = \frac{1}{2}\int_{\Omega} |\, u - u_0 \,|^2 dx + \beta \sum_{i=1}^{2^N} \int_{\Omega} |\nabla \psi_i| dx. \tag{25}$$

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 eqrefeq:cpsi. The first term of (25) 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:

$$\boxed{\min_{\boldsymbol{\phi}, \boldsymbol{c}} F(\boldsymbol{\phi}, \boldsymbol{c}), \quad \text{subject to} \quad \phi_i^2 = 1, \forall\, i.} \tag{26}$$

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:

$$\phi_i^2 = 1, \forall i \;\Leftrightarrow\; \boldsymbol{K}(\boldsymbol{\phi}) = \boldsymbol{0}. \tag{27}$$

We use two related methods to solve the minimization problem (26), 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.

4.1 A Projection Lagrangian Algorithm

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

$$L(\boldsymbol{\phi}, \boldsymbol{c}, \boldsymbol{\lambda}) = F(\boldsymbol{\phi}, \boldsymbol{c}) + \sum_{i=1}^{N} \int_{\Omega} \lambda_i K_i dx. \tag{28}$$

Here $\boldsymbol{\lambda} = \{\lambda_i\}_{i=1}^N$ is a vector of functions of the same dimension as $\boldsymbol{\phi}$, called the Lagrange multiplier. Note that when the constraint is fulfilled, the Lagrangian term vanishes, and $L = F$. We search a saddle point of (28), which in turn will give a minimizer of (26). At a saddle point of L we must have:

$$\frac{\partial L}{\partial \phi_i} = 0, \quad \frac{\partial L}{\partial c_i} = 0, \quad \frac{\partial L}{\partial \lambda_i} = 0, \quad \forall i. \tag{29}$$

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 (28) will not vanish. From the definition of L, we see that:

$$\frac{\partial L}{\partial \phi_i} = \frac{\partial F}{\partial \phi_i} + \sum_{j=1}^{2^N} \lambda_j \frac{\partial K_j}{\partial \phi_i} = (u - u_0) \sum_{j=1}^{2^N} c_j \frac{\partial \psi_j}{\partial \phi_i} - \beta \sum_{j=1}^{N} \nabla \cdot \left(\frac{\nabla \psi_j}{|\nabla \psi_j|} \right) \frac{\partial \psi_j}{\partial \phi_i} + 2\lambda_i \phi_i. \tag{30}$$

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:

$$\frac{\partial L}{\partial c_i} = \int_\Omega \frac{\partial L}{\partial u}\frac{\partial u}{\partial c_i}\,dx = \int_\Omega \frac{\partial F}{\partial u}\frac{\partial u}{\partial c_i}\,dx = \int_\Omega (u - u_0)\,\psi_i\,dx. \tag{31}$$

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

$$\frac{\partial L}{\partial \lambda_i} = K_i = \phi_i^2 - 1. \tag{32}$$

All the derivatives (30) (31) (32) 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 (30) (31) (32) to equal zero, we increase k until none of $\boldsymbol{\phi}$, $\boldsymbol{c}$, $\boldsymbol{\lambda}$ change 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:

$$\boldsymbol{\phi}_t = -\frac{\partial L}{\partial \boldsymbol{\phi}}, \tag{33}$$

to steady state. At a 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:

$$\boldsymbol{\phi_t} \approx \frac{\boldsymbol{\phi}^k - \boldsymbol{\phi}^{k-1}}{\Delta t}. \tag{34}$$

Here Δt is a small positive time step. Combining (33) with (34), and rearranging the terms gives an updating scheme for ϕ_i:

$$\phi_i^k = \phi_i^{k-1} - \Delta t \frac{\partial L}{\partial \phi_i}(\boldsymbol{\phi}^{k-1}, \boldsymbol{c}^{k-1}\,\boldsymbol{\lambda}^{k-1}). \tag{35}$$

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. If an infinite number of iterations were done, that is $t \to \infty$, we would end up with the exact minimizer of L w.r.t $\boldsymbol{\phi}$. However, we terminate the iterations after a finite number of iterations, and use the resulting $\boldsymbol{\phi}^k$ as an approximation to the exact minimizer for the k^{th} iteration.

Secondly, we consider the minimization of L w.r.t $\boldsymbol{c}$, which is done by using (23). 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 $\boldsymbol{b} = (u_0, \psi_i)_{L_2(\Omega)}$.

$$\sum_{i=1}^{2^N} \int_\Omega (\psi_i^k, \psi_j^k)\, c_i^k\,dx = \int_\Omega (u_0, \psi_i^k)\,dx, \quad i = 1, 2 \cdots 2^N. \tag{36}$$

Last, an updating scheme for $\boldsymbol{\lambda}$ is constructed by combining (29) with (30) and (32). A saddle point of L must satisfy

$$0 = \frac{\partial L}{\partial \phi_i} = \frac{\partial F}{\partial \phi_i} + 2\lambda_i \phi_i. \tag{37}$$

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 (37) to get:

$$\lambda_i = -\frac{1}{2}\phi_i \frac{\partial F}{\partial \phi_i}. \tag{38}$$

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

$$\lambda_i^k = -\frac{1}{2}\phi_i^k \frac{\partial F}{\partial \phi_i}(\boldsymbol{\phi}^k, \boldsymbol{c}^k). \tag{39}$$

Now the three updating formulas (35), (36) and (39) are combined to construct an algorithm using the Lagrangian approach. This scheme is essentially a projection Lagrange algorithm.

Algorithm 1 .

Initialize $\boldsymbol{c}^0$, $\boldsymbol{\phi}^0$, $\boldsymbol{\lambda}^0$.

1. Update $\boldsymbol{\phi}^k$ by (35), 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 (36), 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:

   $\lambda_i^k = -\frac{1}{2}\phi_i^k \frac{\partial F}{\partial \phi_i}(\boldsymbol{\phi}^k, \boldsymbol{c}^k), \quad \forall i = 1 \cdots 2^N$.

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 (36) becomes ill-conditioned.

Remark 2

The time step used in the gradient iteration in step 1 is essentially controlled by the size of $\nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|}\right)$, thus big β values require small time steps.

Remark 3

The number of gradient iterations performed in step 1 is usually set to a small number ≈ 10.

4.2 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:

$$L_\mu(\boldsymbol{\phi}, \boldsymbol{c}, \boldsymbol{\lambda}) = F(\boldsymbol{\phi}, \boldsymbol{c}) + \sum_{i=1}^{N} \int_\Omega \lambda_i K_i dx + \tfrac{1}{2}\mu \sum_{i=1}^{N} \int_\Omega K_i^2 dx \tag{40}$$

Here $\mu > 0$ is a penalization parameter, and the last term of (40) 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 (40). 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 $\boldsymbol{\lambda}^k$ and μ^k denote $\boldsymbol{\lambda}$ and μ at the k^{th} iteration. Following the approach in [18, 19], we use the following updating scheme:

$$\boldsymbol{\lambda}^k = \boldsymbol{\lambda}^{k-1} + \mu^{k-1}\boldsymbol{K}(\boldsymbol{\phi}^k). \tag{41}$$

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

$$\phi_i^k = \phi_i^{k-1} - \Delta t \frac{\partial L_\mu}{\partial \phi_i}(\boldsymbol{\phi}^{k-1}, \boldsymbol{c}^{k-1}, \lambda_i^{k-1}), \tag{42}$$

where

$$\frac{\partial L_\mu}{\partial \phi_i} = (u - u_0)\sum_{j=1}^{2^N} c_j \frac{\partial \psi_j}{\partial \phi_i} - \beta \sum_{j=1}^{2^N} \nabla \cdot \left(\frac{\nabla \psi_j}{|\nabla \psi_j|} \right) \frac{\partial \psi_j}{\partial \phi_i} + 2\lambda_i \phi_i + 2\mu(\phi_i^2 - 1)\phi_i. \tag{43}$$

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 (42), 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 (36), 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.

Remark 1

In most of our simulations we have set μ to be constant during the processing. Better convergence behavior can be expected if μ is increased during the process, but be aware of ill-conditioning if μ is increased to quickly. See [18, 19] for details.

Remark 2

As in the first algorithm, $\boldsymbol{c}$ should not be updated too early in the process.

Remark 3

In this algorithm, Δt in the gradient iteration is dependent of both β and μ.

5 Numerical Experiments

In this section numerical results are presented. Essentially our model has two parameters, β and μ. When using the 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 need to be chosen in the following range:

$$10^{-3} \leq \beta \leq 1 \quad \text{and} \quad 10^{-4} \leq \mu \leq 10^{-1}. \tag{44}$$

In each numerical test, we report the value we have 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:

$$u_0 = u + \eta, \tag{45}$$

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

$$SNR = \frac{\text{Variance of Data}}{\text{Variance of Noise}}. \tag{46}$$

If $SNR \approx 1$, the observation data is very noisy. A big SNR indicates a small amount of noise.

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 $\boldsymbol{c}_{opt}$. 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.

All the examples are processed using the augmented Lagrangian Algorithm 2, except in Example 1 and 3 where both algorithms are applied.

5.1 One Level Set Function

Example 1

The first example using one level set function is presented in Figure 2. With this example we want to show that regions where the boundary is not defined by a gradient can be detected. Note that gradient based methods will have trouble with evolving the interface and extract the object from the background in this case. There are no real edges, therefore edges observed in such images are often called cognitive contours.

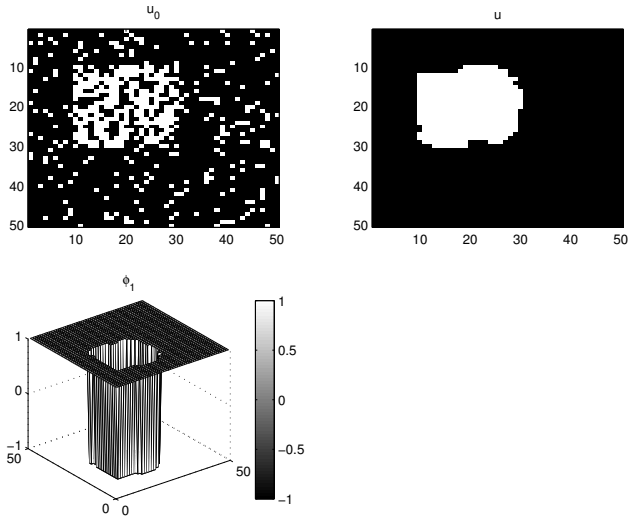


Figure 2: An image with a cognitive contour is processed with $\beta = 3 \cdot 10^{-3}$.

Both algorithms have been used for this example and they give almost identical results. The regularizer minimizes the length of the sub domain boundaries, and thus in some sense works against the constraint K. When using the Algorithm 1, the lack of a penalization term makes it difficult to get ϕ to converge to ± 1. Therefore the sign of ϕ has been used as an approximation to ϕ at convergence. The number of iterations is 100. When using algorithm 2, $\phi(\boldsymbol{x})$ convergence to ± 1 at every point after 300 iterations. We observe that the edges are reasonably well detected.

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, i.e. the level set function should convergence to the sign of some functions. Due to the use of the gradient method, the level set functions could already have the correct sign, but it may still need many iterations to get to ± 1. To overcome this problem, 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 algorithms again. By doing this, the step size Δt could be chosen by a line search when the sign of the iterative solution is used as an initial value.

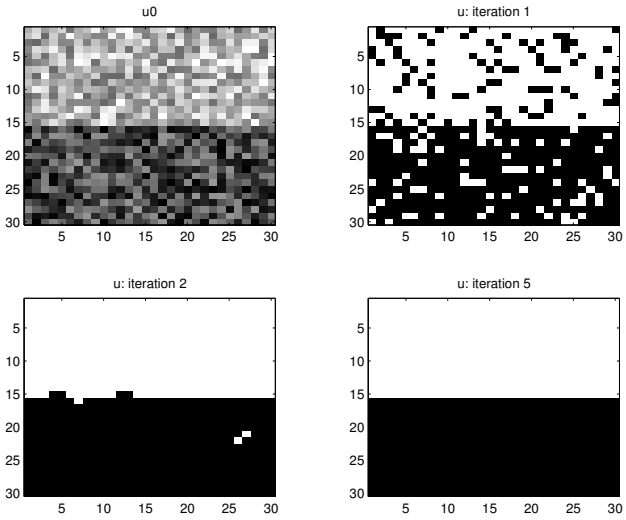


Figure 3: To increase the speed of the method, one can use sign(ϕ) as an approximation to ϕ after a small number of iterations. Here we have used $\beta = 1 \cdot 10^{-3}$ and $\mu = 5 \cdot 10^{-4}$.

We have tested this technique with a simple image imposed with a considerable amount of noise ($SNR \approx 5.5$), see upper left corner of Figure 3. The result after $1, 2$ and 5 iterations are illustrated in Figure 3. At each iteration, the sign of the iterative solution is used as the initial value for the next iteration. Note that 5 iterations is sufficient for recovering the true solution.

Example 3

What happens if we want to approximate more general functions with a piecewise constant function? We have processed two functions. One function is constant along the y direction, but linear along the x direction. Another function is constant along the y direction, but quadratic along the x direction. We want to use our level set methods to approximate it by a piecewise constant function and to see that we can find the correct discontinuity location or not. The calculated result is shown in Figures 4 and 5.

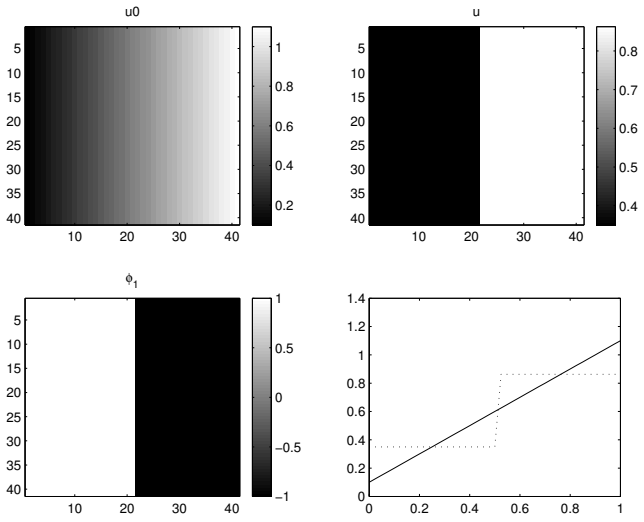


Figure 4: A linear function $\Omega \to \mathbb{R}$ is processed using Algorithm 2. The discontinuity is situated at the middle of the interval. We have used $\beta = 0$ and $\mu = 1 \cdot 10^{-3}$.

As expected by the formulation of the problem, a discontinuity is introduced in the solution. To point out where the discontinuity appears, we have plotted a strip of u. It seems that the discontinuity is introduced at the right place. For the linear function, we must expect the discontinuity of ϕ to be situated in the middle of the domain. For the quadratic function we must expect the discontinuity to be introduced somewhere in between the middle and the right end of the interval.

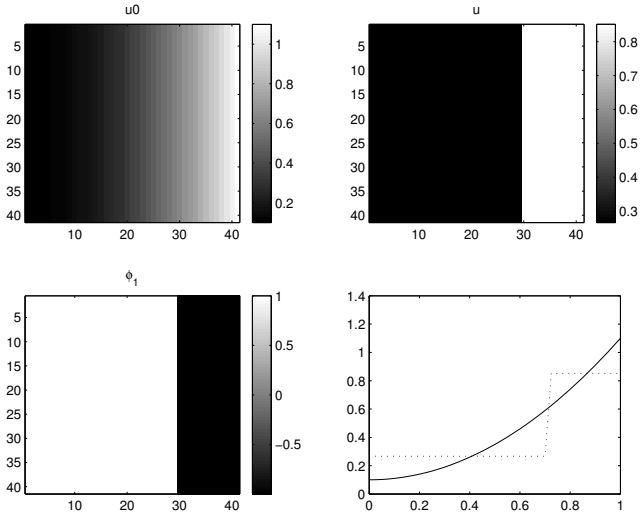


Figure 5: A second order polynomial with the same max and min values as in Figure 4 is processed. As expected, the discontinuity is now shifted towards a higher intensity than in the previous example. We have used $\beta = 0$ and $\mu = 1 \cdot 10^{-3}$.

Example 4

We want to test our algorithms for a real 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 need to 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 "continuous". 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. One of the computed result is shown in Figure 7. The recovered image is very similar to the results obtained in [2, 14].

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 to trace interfaces with sharp corners and with complicated geometries. In term of capturing sharp corners and complicated geometries, 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.

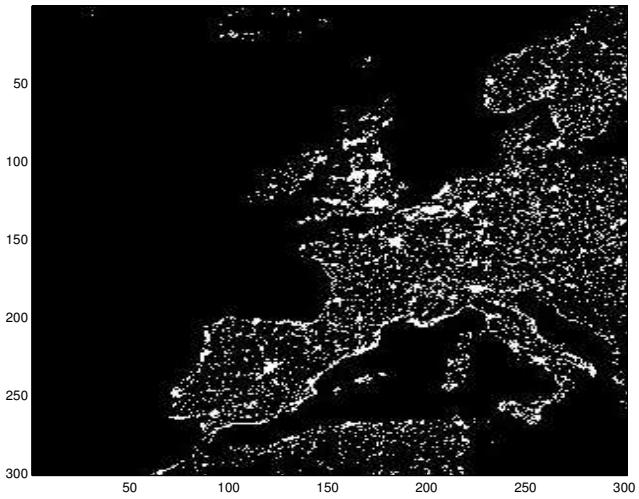


Figure 6: A satellite image of Europe captured at night.

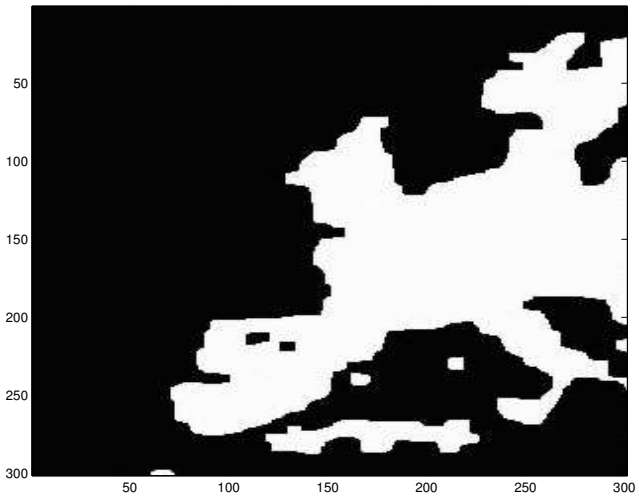


Figure 7: A piecewise constant approximation to a scattered data. We have used $\beta = 1 \cdot 10^{-3}$ and $\mu = 1 \cdot 10^{-4}$. The value of β can be chosen smaller to make Europe less "continuous".

5.2 Two Level Set Functions

Example 5

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. it shall move up or down at a given point. This makes it easier 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 this advantage with our methods, we have tested the algorithms using two level set functions for the image depicted in Figure 8. This same image has been tested by other level set methods in [10]. 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 Figure 8. The sharp corners, concave shapes and inside "holes" impose no problems for our algorithms.

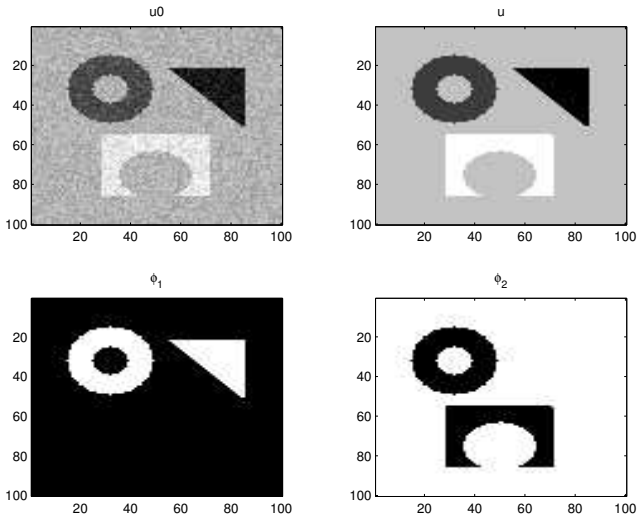


Figure 8: Two level set functions are utilized for detecting four regions having distinct intensities. The values for the parameters are $\mu = 1 \cdot 10^{-4}$ and $\beta = 1 \cdot 10^{-4}$.

Example 6

We would like to keep the image segmentation algorithm as general as possible. Therefore it is important that the algorithm can handle images having any number of phases. Our framework is most naturally adapted to images consisting of $M = 2^N$ regions, where N is the number of ϕ functions, but should also work for any number of phases. We have tested the method on an image containing three regions, using two level set functions.

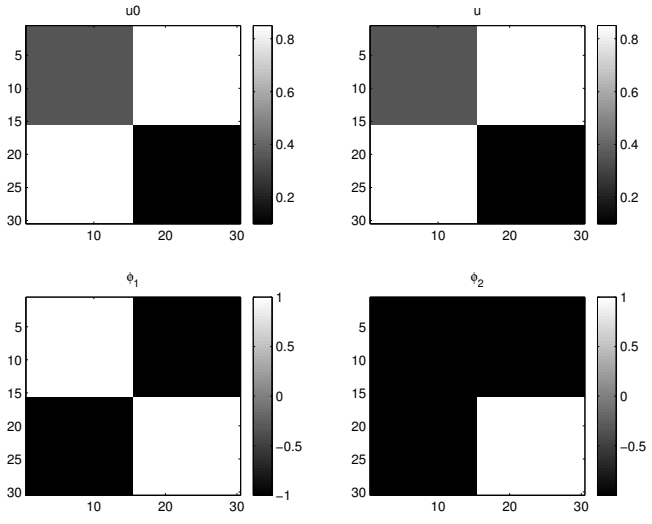


Figure 9: An image consisting of three regions is processed using two level set functions. $\beta = 0$ and $\mu = 1 \cdot 10^{-4}$.

The image and the recovered image and level set functions are shown in Figure 9. As expected, one of the basis functions becomes zero all over Ω. This means that the extra

phase will become empty during the iterative procedure if the level set functions produce more phases than what we need. However, if we know that there are only three phases in the image, we can use some techniques to make the algorithms more stable and fast. One technique is to set one of the c_i values equal to zero for all the iterations. Then it does not matter how the corresponding ψ_i evolves, because it will not give any contribution when constructing u. Another approach is to set two of the c_i values equal. This leads to the result shown in Figure 9, where one of the basis functions vanishes.

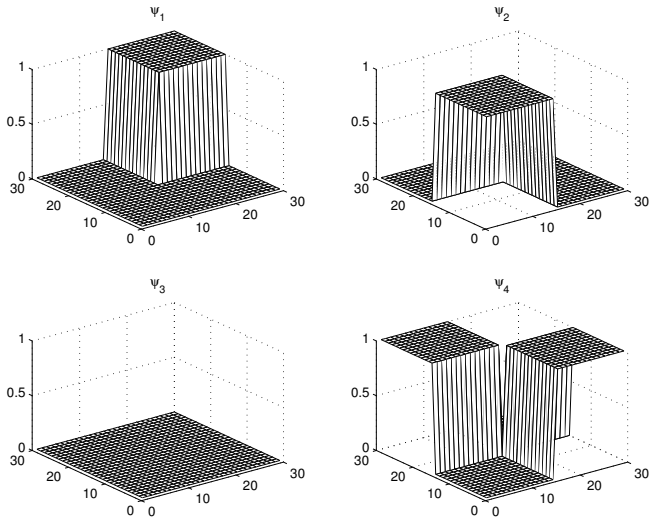


Figure 10: The corresponding ψ_i functions. Note that ψ_3 is empty.

Example 7

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 (11) may not be a problem. For our level set method, the level set functions are discontinuous at convergence, thus the curvature term in (30) 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.

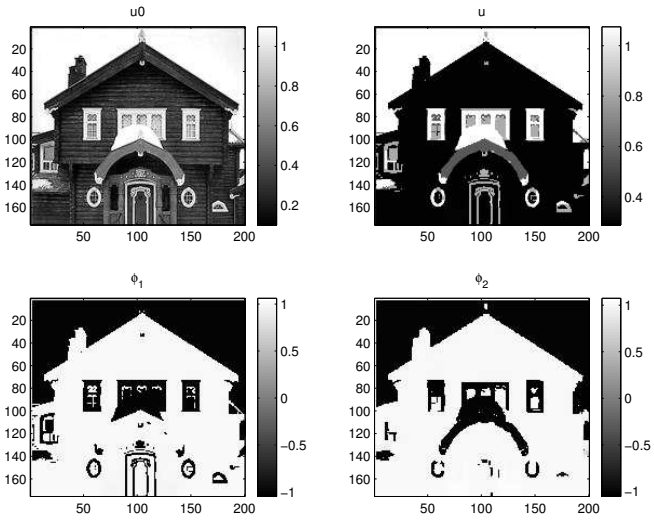


Figure 11: An image of a house is processed using two level set functions. The values for the parameters are $\mu = 1 \cdot 10^{-2}$ and $\beta = 5 \cdot 10^{-5}$.

Example 8

To conclude the numerical section we process a MR-image of a brain. We have picked out an image from the Brainweb database (www.bic.mni.mcgill.ca/brainweb). This is an online database from where synthetic MR-images of the human brain can be downloaded. The input image u_0 in Figure 12 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 Figure 12. 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 $\mu^0 = 5 \cdot 10^{-3}$, and update μ after 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 other traditional level set methods.

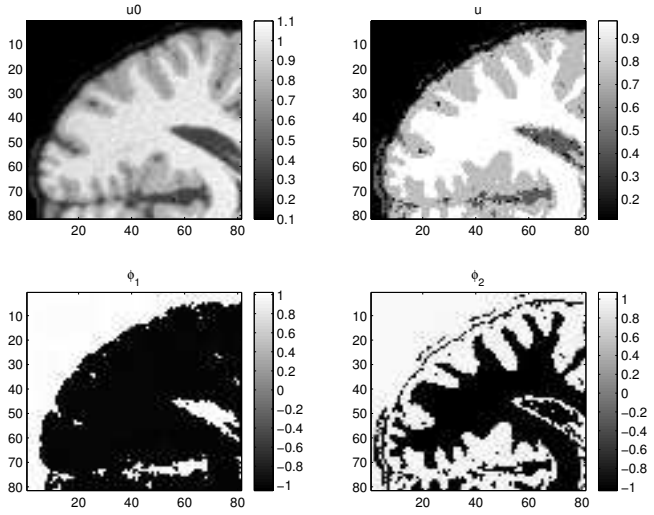


Figure 12: Two level set functions are used to find four regions in a MR-image of a brain. This is a synthetically produced image, downloaded from www.bic.mni.mcgill.ca/brainweb. The value for $\beta = 3 \cdot 10^{-4}$.

6 Conclusion

In this work and also in [14], we have proposed some discontinuous level set methods to capture 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 convex. In this work, we have only tested these methods for image segmentation. We have only tested some simple gradient methods for the iterative algorithms. Due to the fact that the functionals are smooth, we shall be able design other fast iterative algorithms to solve the minimization problems. The numerical results indicate that our methods give as good results as the continuous level set functions. Since our methods are not moving the curves during the iterative procedure, it has some advantages in treating geometries, for example, in a situation where inside "holes" need to be identified.

References

[1] S. Osher and J. Sethian, "Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulation." *J. Comput. Phys.*, vol. 79, 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," *Journal of Computational Physics*, no. 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," *Written on the occasion of Stanley Osher's 60th birthday*, 2001.

[6] M. Burger, B. Hackl, and W. Ring, "Incorporating topological derivatives into level set methods," *CAM-UCLA*, no. 46, 2003.

[7] 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.

[8] ——, *Level Set Methods and Dynamic Implicit Surfaces*, ser. Applied Mathematical Sciences. Springer, 2003, vol. 153.

[9] 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.

[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 discrete approximations," *SIAM Journal on Applied Mathematics*, vol. 55, no. 3, pp. 827–863, Jun. 1995.

[14] J. Lie, M. Lysaker, and X.-C. Tai, "A variant of the levelset method and applications to image segmentation," *UCLA,CAM-03-50*, 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. Catte, T. Coll, and F. Dibos, "A geometric model for active contours in image processing," *Numerische Mathematik*, vol. 66, pp. 1–31, 1993.

[18] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. Belmont Massachusetts: Athena Scientific, 1996.

[19] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 1999.