Journal of Scientific Computing (2021) 87:101
https://doi.org/10.1007/s10915-021-01458-3

A Multigrid Algorithm for Maxflow and Min-Cut Problems with Applications to Multiphase Image Segmentation

Xue-Cheng Tai[1] · **Liang-Jian Deng**[2] · **Ke Yin**[3]

Received: 7 May 2019 / Revised: 11 January 2021 / Accepted: 28 February 2021 /
Published online: 13 May 2021

Abstract
In this paper, we propose a multiphase image segmentation method via solving the min-cut minimization problem under the multigrid method framework. At each level of the multigrid method for the min-cut problem, we first transfer it to the equivalent form, e.g., max-flow problem, then actually solve the dual of the max-flow problem. Particularly, a classical multigrid method is used to solve the sub-minimization problems. Several outer iterations are used for the multigrid method. The proposed idea can be used for general min-cut/max-flow minimization problems. We use multiphase image segmentation as an example in this work. Extensive experiments on simulated and real images demonstrate the efficiency and effectiveness of the proposed method.

Keywords Continuous min-cut and max-flow · Multiphase image segmentation · Multigrid method

1 Introduction

Image segmentation is a fundamental task in image processing that divides an image into several disjoint regions such that each region shares similar features, *e.g.*, texture, and intensity. By the results of image segmentation, many critical subsequent image applications can be well addressed, such as recognition, detection, and searching. Compared with two-phase image segmentation that only needs to divide an image into two disjoint parts, multiphase

✉ Liang-Jian Deng
liangjian.deng@uestc.edu.cn

Xue-Cheng Tai
Xuechengtai@hkbu.edu.hk

Ke Yin
kyin@hust.edu.cn

[1] Department of Mathematics, Hong Kong Baptist University, Kowloon Tsai, Hong Kong

[2] School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, China

[3] Center for Mathematical Sciences, Huazhong University of Science and Technology, Wuhan, China

image segmentation is more practical and challenging. In this work, we mainly focus on the task of *multiphase image segmentation*.

There exist many image segmentation approaches from different perspectives, *e.g.,* conventional discrete optimization-based methods, learning-based methods, and variational methods. Since our method in this work belongs to the category of variational methods, in what follows, we will mainly introduce recent development and trend of variational methods.

During the last two decades, variational methods have become a meaningful way to solve problems related to image segmentation because of their flexibility in model formulation and algorithm design. In particular, if we want to develop a competitive segmentation model, two issues have to be considered. One is how to depict the desired segmented region, and the other is how to model the characteristics and noise of each region [1–9].

The well-known Mumford–Shah model [10] penalizes the ℓ_2 error between the observed image and an unknown piecewise smooth function, as well as the total length of the segmentation boundaries. However, the Mumford–Shah model is challenging to solve since the discretization of an unknown boundary is quite complicated. Therefore, Zhu and Yuille in [11] proposed an explicit active contour approach represent the segmentation boundaries such that the discretization of Mumford–Shah becomes easy and practical. Furthermore, a particular case of the Mumford–Shah model with piecewise constant approximation, called the Chan–Vese model, was proposed by Chan and Vese [12]. In particular, we may solve the Chan–Vese model in [12] efficiently by the level set method [13] Different from the active contour approach, the level set method uses an implicit representation of boundaries such that it can take some advantages, *e.g.*, automatical dealing with the topological change of zero level sets [14–17]. The above mentioned explicit active contour and implicit level set methods both assume that each pixel belongs to a unique region. Different from the two methods, a representative method using a fuzzy membership function considers that each pixel can simultaneously belong to several regions with probability in [0, 1], see [2,18–20]. This type of method has distinct advantages, such as the convex energy functional that guarantees the convergence and stability of the solution, and larger feasible set to find better segmentation results. In [2], based on fuzzy membership functions and ℓ_1 norm fidelity, Li et al. proposed a variational model for multiphase image segmentation. An alternating direction method of multipliers (ADMM) is employed to solve the proposed model efficiently. Moreover, Houhou et al. [19] presented a fast texture segmentation model based on the specific shape operator and active contour. The existence of a solution to the proposed segmentation model is proven, and a simple yet fast algorithm is presented to solve the model.

There exist another explanation for fuzzy membership functions. They are, in fact, the convex relaxation of binary representation, c.f [21]. In the seminal work of Chan–Esedougla–Nikolova [22], it was observed that binary segmentation models could be relaxed to get convex global minimization models and then get a binary label by a threshold. This convex relaxation model is essentially the same as the fuzzy membership function approach. Chambolle, Cremers, and Pock [23] extended this idea to more general problems by function-lifting, i.e., transfer the non-convex minimization problem to a higher dimensional convex minimization problem. There are interesting extensions using graph-cut algorithms and global minimization in [24–29]. One interesting observation in [24–28] is that the convex relaxation models of [22,23] are in fact the dual problem of a continuous min-cut problem which is also equivalent to a continuous max-flow problem. The dual problem and the max-flow problem are convex and thus have global minimizers. Using this interpretation, we could also cast the lifting technique of [23] in the framework of continuous max-flow and min-cut as in [26–28].

One of the purposes of this work is to use multigrid methods to get some fast solvers for this kind of continuous max-flow and min-cut problems.

Since our work is about the multigrid (MG) method, it is necessary to introduce the related works of the multigrid method. The multigrid method [30] is an efficient and recursive approach for large scale systems. It projects the large problems on the finest level to smaller problems on the next level and does this process until the problems can be solved suitably. Multigrid method has been considered in many image applications, such as image restoration [31–36], image registration [37], image segmentation [38,39]. This method is also an ideal way to solve the signal processing problem that will also generate ill-posed linear systems [40]. In [31], Donatelli applied the multigrid method to the task of image deblurring with Tikhonov regularization for the case of shift-invariant point spread function with the periodic boundary condition. In [32], Chen and Tai designed a multigrid method to solve the total variation minimization by using piecewise linear function spanned subspace correction. Also, Badshah and Chen [39] proposed two related multigrid methods for solving the Chan–Vese model to address the problem of multiphase image segmentation.

In this paper, we employ the multigrid method to deal with the min-cut model and its equivalent form as max-flow problem. The golden section algorithm is employed to efficiently solve the dual of the max-flow (DMF) problem on each level of the multigrid method. Besides, due to the non-smoothness of the max-flow problem, the golden section algorithm is implemented to the smoothed version of the sub-minimization problem. In particular, to improve the computation efficiency, a classical Backslash-cycle type of multigrid method is selected to solve the coarser level problems. Furthermore, some outer iterations are applied to the Backslash-cycle multigrid method, aiming to enhance convergence speed. Due to the high efficiency of the multigrid method, the new technique could get competitively fast speed for multiphase image segmentation, competitive with and sometimes much faster than the max-flow based approach [24,25]. Experiments on simulated and real examples show the competitive results of multiphase image segmentation by the given method.

The organization of this paper is as follows. In Sect. 2, we introduce continuous min-cut and max-flow problems which are related to the problem we intend to solve. Section 3 will present the related notations and scheme of the multigrid method, and the details of how to apply the multigrid method to our model for two-phase image segmentation. In Sect. 4, we will first introduce the parallelization of the multigrid method based on four-color subdivision, then extend the two-phase image segmentation to multiphase cases. In Sect. 6, we report the segmentation results by the given method and compare it with a state-of-the-art approach. We also give some experimental analyses of our strategy. Finally, Sect. 7 will draw some conclusions.

2 Continuous Min-Cut and Max-Flow Problems

Given an image f, Mumford and Shah in [10] minimize the following functional:

$$\mathbf{E}(u, \Sigma) = \int_{\Omega\setminus\Sigma} |\nabla u|^2 dx + \lambda \int_{\Omega} (u - f)^2 dx + \mu \text{Length}(\Sigma), \tag{1}$$

with respect to the function u defined on Ω which has discontinuities on Σ. Above, λ, μ are two positive parameters. After [10], the Chan–Vese model was proposed for image segmentation in [12], which minimizes the following energy functional:

$$\mathbf{E}_{CV}(\phi, c_1, c_2) = \int_\Omega (f - c_1)^2 H(\phi) + (f - c_2)^2 (1 - H(\phi)) + \eta \int_\Omega |\nabla H(\phi)|, \quad (2)$$

where ϕ represents a level set function whose zero level curve is the segmentation boundary, c_1, c_2 are two scalars that will be updated in the computing procedure, $H(\cdot)$ is the Heaviside function, and η is a positive parameter. In particular, if the minimizer of the objective functional of Mumford–Shah's model is in the form of $u = c_1 H(\phi) + c_2(1 - H(\phi))$, i.e. a "binary image", then one can easily deduce the Chan–Vese's model. Furthermore, the minimization problem of (2) can be solved by a simple gradient flow.

Let $u = H(\phi)$, it is easy to get a solution of (2) by the following minimization problem

$$\min_{v \in \{0,1\}} \mathbf{E}_{CV}(u, c_1, c_2) = \int_\Omega (f - c_1)^2 v + (f - c_2)^2 (1 - v) + \eta \int_\Omega |\nabla v|, \quad (3)$$

which was proposed in [21] and referred as *binary level set* approach. More generally, consider

$$\min_{v \in \{0,1\}} \mathbf{E}_{potts}(v) = \int_\Omega f_1(x) v(x) + f_2(x)(1 - v(x)) + \int_\Omega g(x) |\nabla v(x)|, \quad (4)$$

where f_i are some given functions indicating the possibility of a point belonging to phase 0 or 1. The function $g(x)$ can be just a constant or an edge detector function. Especially, the Chan–Vese model in [12] is actually a special case of this model if c_i are known. Recently, it was observed that this is a *min-cut problem* [24,27] which is equivalent to a *max-flow problem* that is written as

$$\begin{aligned} &\max_{p_s, p_t, q} \int_\Omega p_s dx \text{ subject to:} \\ &p_s(x) \le f_1(x), \ p_t(x) \le f_2(x), \ |q(x)| \le g(x), \forall x \in \Omega, \\ &\text{div} q(x) - p_s(x) + p_t(x) = 0, \forall x \in \Omega, \ \ q \cdot n = 0 \text{ on } \partial\Omega, \end{aligned} \quad (5)$$

where n is the unit out normal vector of $\partial\Omega$, p_s is a scalar function and is the amount of flow from a "source" to x, p_t is a scalar function representing the amount of flow to the "sink" and $q(x) = (q_1(x), q_2(x))$ is the flow inside the domain. It is necessary to emphasis that $|q| = \sqrt{q_1^2 + q_2^2}$. In particular, the method also works if using $|q| = |q_1| + |q_2|$. In such a case, its discrete version is actually a max-flow problem that can be easily solved by traditional graph cut method. Note that the functions f_i and g in (5) are the same as in (4). The goal is to maximize the total amount of flow in the system (which is the domain connected with a "source" and a "sink") with flow conservation governed by the last equation of (5). Note that we also need to assume that there is no "flow" in and out from the domain boundary $\partial\Omega$. The dual of the max-flow problem (5) can be written as follows (c.f. [27, Prop. 3.1]):

$$\min_{u(x) \in [0,1]} \int_\Omega (f_1 u + f_2(1 - u) + g(x)|\nabla u|) dx. \quad (6)$$

As explained in [27], the function u in (6) is actually the Lagrangian multiplier for the flow conservation equation in (5).

Note that the above three problems are truly equivalent:

$$(4) \Leftrightarrow (5) \Leftrightarrow (6).$$

It is necessary to emphasize that the equivalence between (4) and (6) was first observed in the seminal work of Chan–Esedouglu–Nikolova [22] by a different derivation. Connections between these models with the well-known graph-cut approaches are also explained in some

recent works, see [24–27,41]. There are two advantages of continuous min-cut and max-flow models: (i) Both (5) and (6) are convex minimization problems; thus they are guaranteed to have global minimizers; (ii) A number of the well developed convex minimization algorithms, especially some operator splitting and augmented Lagrangian methods, can be employed to solve them efficiently.

These algorithms can classify the domain into two phases. There are three different ways to extend it to multiphase segmentation with the same kind of advantages as outlined below, and a brief survey of these approaches can also be found in [42].

Approach I: The first approach for multiphase min-cut/max-flow with a given graph was explained in [25]. There exists a max-flow and min-cut problem on the graph that are dual to each other and guaranteed to have global minimizers. This graph is given in the continuous setting. When it is discretized, we get the commonly used discrete graph with vertexes. The details of the graph and the model can be found in [25, Prop. 1].

Approach II: Another multiphase image segmentation model using graphs is the Ishikawa graph [43–45]. The interpretation of this graph model as a min-cut and max-flow problem can be found in [41,46]. The interpretation of this model as convex relaxation can be found in [23].

Approach III: A third approach for multiphase image segmentation is to use the product of labels as in [28,29,47], which essentially extends the multiphase Chan–Vese approach [48]. The interpretation of the product of labels as min-cut and max-flow problems over a graph can be found in [28,29].

Later in this work, we shall use multigrid methods to solve the multiphase min-cut problem related to Approach I. For this approach, the corresponding graph is shown in Figure 1 in [25]. We need to copy the image domain K times and properly connect them, see [25, p. 385]. The min-cut on this graph solves the multiphase Potts model:

$$\min_{\Omega_i} \sum_{i=1}^{K} \int_{\Omega_i} f_i(x)dx + \sum_{i=1}^{K} \int_{\partial\Omega_i} g(x)ds \text{ s.t. } \cup_{i=1}^{M} \Omega_i = \Omega, \quad \Omega_k \cap \Omega_l = \varnothing, \forall k \neq l.$$

The corresponding max-flow problem needs to maximize an energy functional with functions:

1. A scalar function $p_s(x)$ which is the amount of flow from the source to x;
2. A vectorial function $h(x) = (h_1(x), h_2(x), \cdots h_K(x))$ and each $h_i(x)$ is a scalar function which is the amount of flow from a point x of the ith copy of the image domain to the "sink";
3. A vectorial function $q(x) = (\mathbf{q}_1(x), \mathbf{q}_2(x), \cdots, \mathbf{q}_K(x))$ and each $\mathbf{q}_i(x) = (q_i^1(x), q_i^2(x)$ is the flow vector function on each copy of the image domain.

The max-flow problem needs to solve

$$\max_{p_s,h,q} \int_{\Omega} p_s(x)dx, \tag{7}$$

under constraints:

$$h_i(x) \leq f_i(x), \quad |q_i(x)| \leq g(x), \quad (\text{div} q_i - p_s + h_i)(x) = 0, \forall x \in \Omega, \ i = 1, 2 \cdots K. \tag{8}$$

The dual problem of the above max-flow problem is the following convex min-cut problem (c.f [25, p. 386]):

$$\min_{u} \sum_{i=1}^{K} \left(\int_{\Omega} u_i(x) f_i(x)dx + \int_{\Omega} g(x)|\nabla u_i(x)|dx \right), \text{ s.t. } \sum_{i=1}^{k} u_i(x) = 1, \quad u_i(x) \geq 0. \tag{9}$$

Table 1 Main notations used in this work

Notation	Meaning	Remark
j	Level # of MG method	$j = 0, 1, \ldots, J$
n_j	Element # on jth level	$n_j = 4^j$
i	Index for elements on jth level	$i = 1, 2, \ldots, n_j$
$\tau_{i,j}$	ith element on jth level (Fig. 1)	$i = 1, 2, \cdots, n_j;\ j = 0, 1, \ldots, J$
N	A gray image with size $N \times N$	$N \in \mathbb{N}^+$
$P_{i,j}$	Center points of elements over the finest mesh that are inside $\tau_{i,j}$	$\frac{N^2}{4^j}$ center points, (Figs. 1, 2)
$B_{i,j}$	Center points of elements over the finest mesh that are inside $\tau_{i,j}$ and adjacent to $\partial\tau_{i,j}$	Fig. 2
$\tilde{B}_{i,j}$	Center points of elements over the finest mesh that are outside $\tau_{i,j}$ and adjacent to $B_{i,j}$	Fig. 2
p	phase #	$p = 1, 2, \ldots, K$
$\mathcal{N}(x)$	Center points of the down and right neighbor elements of the element centered at x over the finest mesh	$N(x) = \{N_d(x), N_r(x)\}$

3 Multigrid Algorithm for Piecewise Constant Approximations for Two-Phase Min-Cut Minimization Problems

In the following, we try to use the multigrid method to solve the dual problem of the max-flow problem (6), i.e., we will consider

$$\min_{u(x)\in[0,1]} F(u), \qquad F(u) = \int_\Omega (f_1 u + f_2(1-u) + g(x)|\nabla u|)dx.$$

We assume that Ω is a rectangular domain, and there is a coarse mesh partition that divides the domain into some coarse rectangular elements. Then, we refine each rectangular element into four equal rectangular sub-mesh elements to get the next level of fine mesh. We continue this refinement J times, and the finest mesh is the one that we shall use for processing an image. In practical applications for image processing, the pixels of the given image already defined the finest mesh, and it is always possible to produce the mesh given by the pixels from the above-outlined refinement process.

To explain the details in the derivation of the multigrid methods, we need to introduce some nations. We use $\tau_{i,j}$ to denote the ith element on jth level with $j = 0$ be the coarsest mesh. $P_{i,j}$ denotes the center points of the element $\tau_{i,j}$ *over the finest mesh*. n_j represents the number of elements on level j. The other notations are summarized in Table 1 and explained in Figs. 1 and 2.

We regard an image to be a piecewise constant function over the finest mesh. The multigrid method we shall use is to minimize the energy for the min-cut problem not only on the finest mesh but also over all the coarser level meshes. The algorithm is given in Algorithm 1, and we shall explain the details in the following. The updating is shown as a diagram in Fig. 3. The interpolation and prolongation between the levels are implicitly handled in the subproblem updating.

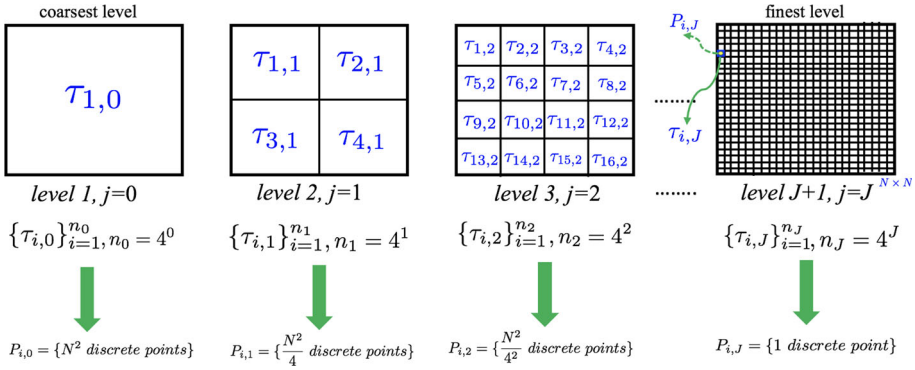


Fig. 1 An illustration of refinement and the finest mesh and notations of $\tau_{i,j}$ and $P_{i,j}$. $P_{i,j}$ is the set containing the center points for all the elements over the finest mesh that are inside $\tau_{i,j}$. If $j < J$, $P_{i,j}$ contains multiple center points, e.g., $P_{i,0}$ containing N^2 center points of $\tau_{i,0}$ over the finest mesh

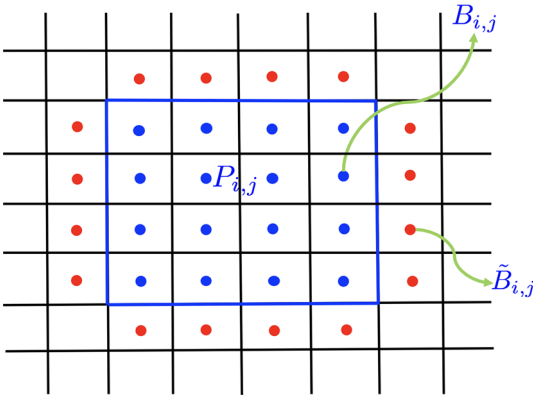


Fig. 2 An illustration of $P_{i,j}$, $B_{i,j}$ and $\tilde{B}_{i,j}$. $P_{i,j}$ (blue points) represents the *center points* for all elements over the finest mesh that are inside $\tau_{i,j}$ (the region enclosed by the blue line), $B_{i,j}$ contains all the center points for element over the finest mesh that are inside $\tau_{i,j}$ and adjacent to $\partial\tau_{i,j}$. $\tilde{B}_{i,j}$ contains all the center points for elements over the finest mesh that outside $\tau_{i,j}$ and adjacent to $B_{i,j}$ (Color figure online)

Algorithm 1 Multigrid algorithm for the dual problem of the max-flow (6)

Input: Input function f, the maximum iterations *maxIter*, the converged threshold tol
Output: u_{out}
Initialization: $E_{old} = 0$, an initial value $u \in [0, 1]$ for u
for $k = 1 : maxIter$ **do**
  for $j = 0 : J$ **do**
    for $i = 1 : n_j$ **do**
      $e_{i,j} = \arg\min_{c \in \mathcal{C}_{i,j}} F(u + c\phi_{i,j})$ by (11)–(12)
      $u \leftarrow u + e_{i,j}\phi_{i,j}$
    end for
  end for
  Compute total energy $E = E(u)$ via (6)
  if $|E - E_{old}|/|E| < tol$ **then** break;
  else set $E_{old} = E$
  end if
end for
Output $u_{out} = u$

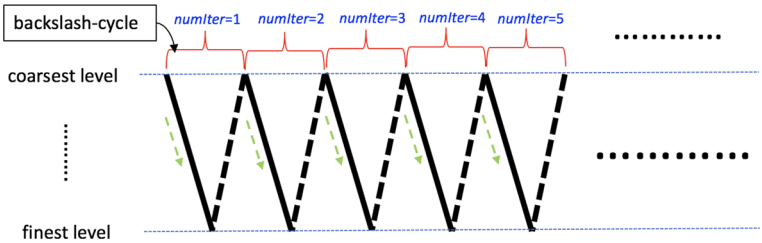


Fig. 3 An illustration of Backslash-cycle multigrid method with several outer iterations

This multigrid algorithm is trying to minimize the min-cut energy functional in (6) successively over all the elements from all the mesh levels. In the following, we explain the details in solving the minimization subproblems and also give the definition of the constraint $\mathcal{C}_{i,j}$.

First, we note that the piecewise constant finite element functions space over the different mesh levels can be written as:

$$V_j = \{v|\ v|_{\tau_{i,j}} \in \mathbf{R}, \quad \forall \tau_{i,j}\},\ j = 0, 1, 2 \cdots J.$$

The basis functions for these spaces are:

$$\phi_{i,j}(x) = \begin{cases} 1, & x \in \tau_{i,j}, \\ 0, & \text{else.} \end{cases} \tag{10}$$

We have that

$$V_j = \text{span}(\{\phi_{i,j}\}_{i=1}^{n_j}),\ j = 0, 1, 2 \cdots J.$$

In our multigrid method, all the integrations will be done over the finest mesh. The function $u \in V_J$ is always updated over the finest mesh. For $c \in \mathbf{R}$, we for simplicity define

$$v_c(x) = u(x) + c\phi_{i,j}(x) = \begin{cases} u(x) + c, & \forall x \in \tau_{i,j}, \\ u(x), & \text{else} \end{cases}$$

This is also a finite element function over the finest mesh due to the fact that $\phi_{i,j} \in V_j \subset V_J,\ \forall j \le J$.

The minimization subproblem in Step 7 in Algorithm 1 in the discrete setting is to solve:

$$\begin{aligned} e_{i,j} &= \arg\min_{c \in \mathcal{C}_{i,j}} F(v_c) = \arg\min_{c \in \mathcal{C}_{i,j}} F(u + c\phi_{i,j}) \\ &= \arg\min_{c \in \mathcal{C}_{i,j}} \Bigg(\sum_{x \in P_{i,j}} [f_1(x)v_c(x) + f_2(x)(1 - v_c(x))] \\ &\quad + \sum_{x \in P_{i,j} \setminus B_{i,j}} g(x) \sqrt{\sum_{y \in \mathcal{N}(x)} |v_c(y) - v_c(x)|^2} \\ &\quad + \sum_{x \in B_{i,j}} g(x) \sqrt{\sum_{y \in \mathcal{N}(x) \bigcap B_{i,j}} |v_c(y) - v_c(x)|^2 + \sum_{y \in \mathcal{N}(x) \bigcap \tilde{B}_{i,j}} |u(y) - v_c(x)|^2} \\ &\quad + \sum_{y \in \tilde{B}_{i,j}} g(y) \sqrt{\sum_{x \in \mathcal{N}(y) \bigcap B_{i,j}} |v_c(x) - u(y)|^2 + \sum_{z \in \mathcal{N}(y) \bigcap \tilde{B}_{i,j}} |u(z) - u(y)|^2} \Bigg) \end{aligned}$$

$$= \arg\min_{c\in\mathcal{C}_{i,j}} \Bigg(\sum_{x\in P_{i,j}} [f_1(x) - f_2(x)]\, c + \sum_{x\in B_{i,j}} g(x) \sqrt{v(x) + \sum_{y\in\mathcal{N}(x)\bigcap \tilde{B}_{i,j}} |u(x) - u(y) + c|^2} + \sum_{y\in \tilde{B}_{i,j}} g(y) \sqrt{\sum_{x\in\mathcal{N}(y)\bigcap B_{i,j}} |u(x) - u(y) + c|^2 + \tilde{v}(y)}, \Bigg) \tag{11}$$

where $\mathcal{N}(x)$ is as defined in Table 1 and it represents the down and right neighbor elements center points of the element centered at x, $v(x) = \sum_{y\in\mathcal{N}(x)\bigcap B_{i,j}} |u(x) - u(y)|^2$ with $x \in B_{i,j}$, and $\tilde{v}(y) = \sum_{z\in\mathcal{N}(y)\bigcap \tilde{B}_{i,j}} |u(z) - u(y)|^2$ with $y \in \tilde{B}_{i,j}$. We have cast the terms that are independent of c in getting equalities in the above formulas. The summations in the above formula are done over the finest mesh as all the center points from $P_{i,j}$, $B_{i,j}$, $\tilde{B}_{i,j}$ are centers of elements from the finest level.

For the minimization subproblem in Step 7 in Algorithm 1, we need to guarantee that all the updated values for u satisfy $u \in [0, 1]$. This gives the constraint set $\mathcal{C}_{i,j}$ for c in Step 7 as follows:

$$\begin{aligned} u + c\phi_{i,j} \in [0,1] &\Leftrightarrow -\frac{u(x)}{\phi_{i,j}(x)} \le c \le \frac{1-u(x)}{\phi_{i,j}(x)},\ x \in \tau_{i,j} \\ &\Leftrightarrow \mathcal{C}_{i,j} = [-\min_{x\in\tau_{i,j}} u(x),\ 1 - \max_{x\in\tau_{i,j}} u(x)]. \end{aligned} \tag{12}$$

Here are some remarks about this algorithm:

- Note that we always have $u \in [0, 1]$ during the iterations, thus the constraint in (12) is always non-empty.
- Function $u(x) \in V_J$ is regarded as a finite element function defined over the finest mesh, thus the min-max values of $u(x)$ over $\tau_{i,j}$ in (12) is evaluated over the finest elements that are inside $\tau_{i,j}$.
- The updating in Step 8 is always done over the finest mesh, i.e. over mesh at level J, and so there is a piecewise constant interpolation from the j level to the finest mesh at level J here.

The problem (11) is a minimization problem with real number c, we may denote the objection function as $h(c)$, thus the minimization problem can be written as:

$$\min_{a\le c\le b} h(c), \tag{13}$$

which can be solved by golden section method, c.f. [49].

4 Parallelization of Mutligrid Method

Parallelization is crucial for many applications with the min-cut/max-flow approach. Here, we shall partition the elements over each level onto four colors and do the updating with the elements of the same color in parallel. As illustrated in Fig. 4, the element over each level j can be partition into 4 groups that are marked with 4 colors. The elements of the same color do not intersect each other. We use $\tau_{i,j}^{col}$ to denote the ith element over the jth level with color numbered as $col \in \{1, 2, 3, 4\}$. We assume that the number of elements with the color

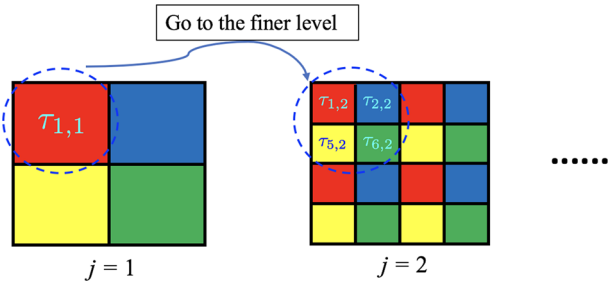


Fig. 4 An illustration to partition the elements over each level into 4 groups marked with 4 colors

Table 2 Main notations used in this work

Notation	Meaning	Remark
col	Color index with the 4-color partition	$col = 1, 2, 3, 4$
τ_j^{col}	Union of elements on jth level with colth color	See (14) and Fig. 4
n_j^{col}	Number of elements in τ_j^{col}	–
P_j^{col}	Center points for finest mesh elements inside τ_j^{col}	See (14)
B_j^{col}	Center points for finest mesh elements inside τ_j^{col} and adjacent to $\partial\tau_j^{col}$	See (14)
$\tilde{B}_j^{col}$	Center points for finest mesh elements outside τ_j^{col} and adjacent to B_j^{col}	See (14)

col at the jth level is n_j^{col}. Then we know that $n_j^{col} = 4^{j-1}$ and $n_j = 4^j$. Correspondingly, we define

$$\begin{aligned} \tau_j^{col} &= \bigcup_{i=1}^{n_j^{col}} \tau_{i,j}^{col}, \quad P_j^{col} = \bigcup_{i=1}^{n_j^{col}} P_{i,j}^{col}, \\ B_j^{col} &= \bigcup_{i=1}^{n_j^{col}} B_{i,j}^{col}, \quad \tilde{B}_j^{col} = \bigcup_{i=1}^{n_j^{col}} \tilde{B}_{i,j}^{col}, \end{aligned} \quad j = 0, 1, 2, \cdots, J; \quad col = 1, 2, 3, 4, \tag{14}$$

We emphasise again that *all the center points are defined over the finest mesh*. For convenience, we use $\mathcal{I}_j^{col}$ to denote the indexes of the elements of τ_j^{col}, i.e

$$\mathcal{I}_j^{col} = \{i | \ \tau_{i,j} \text{ has the color } col\}, \quad j = 0, 1, 2, \cdots, J; \quad col = 1, 2, 3, 4,$$

A summary with the notations with the 4-color partition is given in Table 2. With these notations, the colored parallel multigrid algorithm for the max-flow/min-cut problem (6) is written in detail in Algorithm 2. Next we explain the needed details for the minimization

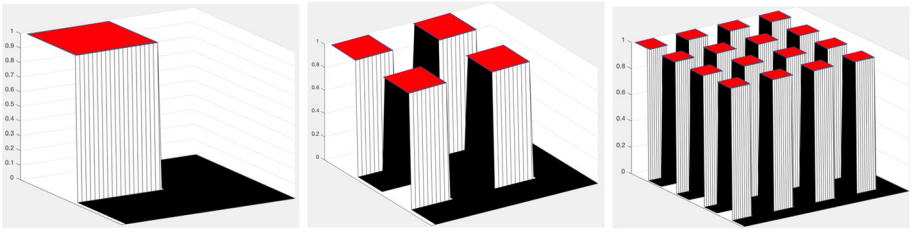


Fig. 5 An illustration of $\phi_{i,j}$ for the red color, i.e. $col = 1$. Take the level 1 to the level 3 as example (Color figure online)

problem in Step 7 of Algorithm 2 in the discrete setting. For notation simplicity, for a given vector $\mathbf{c} = (c_1, c_2, \cdots, c_{n_j^{col}}) \in \mathbf{R}^n_{\mathbf{j}^{col}}$, let us define

$$s(x) = \sum_{i \in \mathcal{I}_j^{col}} c_i \phi_{i,j}(x).$$

This function is a piecewise constant function taking the constant value c_i in the elements over the jth level with the color col, see Fig. 5 for an illustration of $\phi_{i,j}$ with the red color as an example. The minimization problem in Step 7 of Algorithm 2 in the discrete setting is to solve the following problem:

$$\begin{aligned} \mathbf{e}_{col,j} = \arg\min_{\mathbf{c} \in \mathcal{C}_j^{col}} \Bigg(& \sum_{x \in P_j^{col}} [f_1(x) - f_2(x)]\, s(x) \\ & + \sum_{x \in B_j^{col}} g(x) \sqrt{v(x) + \sum_{y \in \mathcal{N}(x) \bigcap \tilde{B}_j^{col}} |u(x) - u(y) + s(y)|^2} \\ & + \sum_{y \in \tilde{B}_j^{col}} g(x) \sqrt{\tilde{v}(y) + \sum_{x \in \mathcal{N}(y) \bigcap B_j^{col}} |u(x) - u(y) + s(x)|^2} \Bigg). \end{aligned} \tag{15}$$

The minimizer $\mathbf{e_{col,j}}$ of the above problem is a vector in $\mathbf{R}^{n_j^{col}}$. Again, we emphasize that the summation in the above formula is done over the finest mesh. The good point of this algorithm is that the values of the $\mathbf{e}_{col,j}$ can be computed in parallel, i.e. minimization problem in Step 7 of Algorithm 2 can be computed in parallel over elements for a fixed level j and a fixed color $col \in \{1, 2, 3, 4\}$. The constraint set $\mathcal{C}_j^{col}$ can be deduced in the same way as for Algorithm 1, which is:

$$\mathcal{C}_j^{col} = \{(c_1, c_2, \cdots c_{n_j^{col}}) |\ \ c_i \in [-\min_{x \in \tau_{i,j}} u(x), 1 - \max_{x \in \tau_{i,j}} u(x)], \ \ i = 1, 2, \cdots, n_j^{col}\}. \tag{16}$$

In the following, we give some detailed explanations about Algorithm 2. For the implementation, u will be a matrix on the finest mesh, and it will always be stored and updated on the finest mesh.

In the Step 8 of Algorithm 2, $(\mathbf{e_{col,j}})_i$ are the values of the ith component of the minimizer in Step 7 which is a vector in $\mathbf{R}^{n_j^{col}}$. We have the following remarks about Algorithm 2:

Algorithm 2 Parallel multigrid algorithm with four color partitions

1: **Input:** Input function f, the maximum iterations *maxIter*, the converged threshold tol, a smoothing constant δ.
2: **Output:** u_{out}.
3: **Initialization:** $E_{old} = 0$, an initial value $u_0 \in [0, 1]$ for u.
4: **for** k = 1 : *maxIter* **do**
5: **for** j = 0 : J **do**
6: **for** col = 1 : 4 **do**
7: $\mathbf{e}_{col,j} = \arg\min_{(c_1,c_2,\cdots c_{n_j^{col}})\in C_j^{col}} F(u + \sum_{i\in\mathcal{I}_j^{col}} c_i\phi_{i,j})$
8: $u \leftarrow u + \sum_{i\in\mathcal{I}_j^{col}} (\mathbf{e}_{col,j})_i\phi_{i,j}$
9: **end for**
10: **end for**
11: Compute total energy $E = E(u)$ via (26)
12: **if** $|E - E_{old}|/|E| < tol$ **then break**
13: **else** set $E_{old} = E$
14: **end if**
15: **end for**
16: Output $u_{out} = u$

– The minimization problem in Step 7 can be done in parallel over the elements in τ_j^{col}, i.e. $i \in \mathcal{I}_j^{col}$. Our code is implemented in Matlab. By implementing the minimization over the elements with the same color in parallel, we observe huge computing time improvement.
– The function $u(x)$ is a finite element function over the finest mesh. So the updating in Step 8 is always done over the finest mesh, and there is a piecewise constant interpolation from the jth level to the finest mesh at level J. Even more, they can be updated in parallel over the elements of the same color. The coloring of the elements is done over each level. The updating for u is done over the finest mesh elements that are inside each $\tau_{i,j} \subset \tau_j^{col}$, i.e. we add value $(\mathbf{e}_{col,j})_i$ to all elements over the finest mesh that are inside $\tau_{i,j} \subset \tau_j^{col}$. Element $\tau_{i,j}$ is on the jth level, and it can contain many elements over the finest mesh on level J.

5 Multigrid Method for Multiphase Min-Cut/Max-Flow

In this section, we intend to use multigrid for multiphase (K phases) min-cut problem (9). We will present the algorithm without the coloring of the elements. The parallel colored multigrid algorithm can be deduced in a similar way as for Algorithm 2. For the multiphase min-cut problem, we only need to replace the scalar function u in Algorithm 1 by a vector function

$$\mathbf{u}(x) = (u_1(x), u_2(x), \cdots, u_K(x)) \in S,$$

with

$$S = \{(v_1(x), v_2(x), \cdots, v_K(x))|\ \ v_k(x) \geq 0,\ k = 1, 2, \cdots K,\ \ \sum_{k=1}^{K} v_i(x) = 1\}.$$

We still try to update the $\mathbf{u}$ values over all elements $\tau_{i,j}$ with all i and j. Let $\mathbf{u}$ be a given values at a given iteration, we update by

$$\mathbf{u}(x) \leftarrow \mathbf{u}(x) + \mathbf{c}\phi_{i,j} = \big(u_1(x) + c_1\phi_{i,j}, u_2(x) + c_2\phi_{i,j}, \cdots, u_K(x) + c_K\phi_{i,j}\big),\ \mathbf{c} \in \mathbf{R}^K.$$

To guarantee that $\mathbf{u}(x) + \mathbf{c}\phi_{i,j} \in S$, we need

$$\begin{cases} \sum_{p=1}^{K}(u_p + c_p) = 1, \\ 0 \le u_p + c_p \le 1. \end{cases} \tag{17}$$

Here u_p represents the values of the vector function $\mathbf{u}$ of the last iteration on the pth phase, and thus it satisfies $\mathbf{u} \in S$. Especially, (17) can be simplified to the following

$$\sum_{p=1}^{K} c_p = 0, \quad -u_p \le c_p \le 1 - u_p. \tag{18}$$

Thus, to extend Algorithm 1 to the multiphase min-cut/max-flow problem (9), we just need to change the scalar label function $u(x)$ to a vector label function $\mathbf{u}(x) \in S$ and replace the minimization problem in Step 7 by solving an approximate minimizer for

$$\mathbf{e}_{i,j} = \arg\min_{\mathbf{c}\in\mathcal{C}_{i,j}} F(\mathbf{u} + \mathbf{c}\phi_{i,j}), \tag{19}$$

with F being the energy functional given in (9), i.e.

$$F(\mathbf{v}) = \sum_{p}^{K} \int_{\Omega} \Big(v_p(x) f_p(x) dx + g(x)|\nabla v_p(x)| \Big) dx, \quad \mathbf{v} = (v_1(x), v_2(x), \cdots v_K(x)). \tag{20}$$

From (18), the constraint set $\mathcal{C}_{i,j}$ for (19) is

$$\mathcal{C}_{i,j} = \Big\{ \mathbf{c} | \quad \sum_{p=1}^{K} c_p = 0, \text{ in } \tau_{i,j}, \quad -\min_{x\in\tau_{i,j}}(u_p)(x) \le c_p \le 1 - \max_{x\in\tau_{i,j}}(u_p)(x) \Big\}. \tag{21}$$

In our implementations, we first solve (19) without the constraint $\mathcal{C}_{i,j}$ and then project the minimizer to the simplex $\mathcal{C}_{i,j}$. We just do this once and take it as an approximate minimizer for (19). It is clear that F is separable, i.e.

$$F(v) = \sum_{p} F_p(v_p), \text{ with } F_p(v_p) = \int_{\Omega} v_p(x) f_p(x) dx + g(x)|\nabla v_p(x)| dx. \tag{22}$$

This means that the computation for minimization problem (19) without the constraint $\mathcal{C}_{i,j}$ can be done in parallel for each p.

If we let $e_{i,j,p}$ be the value of the pth component of an approximate minimizer of the (19), then the updating of the label functions $u_p(x)$ can be done in parallel for $p = 1, 2, \cdots K$ by

$$u_p \leftarrow u_p + e_{i,j,p}\phi_{i,j}, \quad p = 1, 2, \cdots K. \tag{23}$$

From these explanations, we can see that the extension from 2-phase min-cut/max-flow to multiphase is very easy. All the codes for the 2-phase min-cut problem can be used for each $u_p(x)$. The only extra task is to handle the extra constraint $\sum_{p=1}^{K} c_p = 0$. We have tested both on the algorithm of [50,51]. Both are rather fast. In the tests given later, we have used the algorithm of [50].

The algorithm is summarized in Algorithm 3.

Algorithm 3 Multigrid algorithm for the dual problem of the max-flow (9)

1: **Input:** Input function f_i and g, the maximum iterations *maxIter*, the converged threshold tol
2: **Output:** u_{out}
3: **Initialization:** $E_{old} = 0$, an initial value $u_p \in [0, 1]$ for $p = 1, 2, \cdots K$
4: **for** k = 1 : *maxIter* **do**
5: **for** $j = 0 : J$ **do**
6: **for** $i = 1 : n_j$ **do**
7: Find a minimizer in parallel for $p = 1, 2, \cdots K$ for
8: $\tilde{e}_{i,j,p} = \arg\min_{c_p \in \mathbf{R}} F_p(u_p + c_p \phi_{i,j})$.
9: Project $(\tilde{e}_{i,j,p})_{p=1}^{K}$ into the simplex $\mathcal{C}_{i,j}$ in (21) to get $(e_{i,j,p})_{p=1}^{K}$.
10: Update u_p in parallel for p as $u_p \leftarrow u_p + e_{i,j,p}\phi_{i,j}$
11: **end for**
12: **end for**
13: Compute total energy $E = E(u)$ via (9)
14: **if** $|E - E_{old}|/|E| < tol$ **then break**
15: **else** set $E_{old} = E$
16: **end if**
17: **end for**
18: Output $u_{out} = u$

The first minimization problem in Step 8 in Algorithm 3 is again solved by the Golden section with the lower and upper bound given by the second inequality in the definition of $\mathcal{C}_{i,j}$ in (21). It is easy to implement this algorithm with 4-color partition of the elements over each level $j = 0, 1, \cdots J$.

6 Numerical Results

In this section, we compare the proposed method with three state-of-the-art approaches, named FCM-L1 method [2], SLaT method [52] and max-flow method [24]. Note that the solved model in this work is the similar as that in the max-flow method; thus, we will do some discussions for the two approaches in this section. Especially, the model used in the FCM-L1 and SLaT methods are different from that of the max-flow method and the proposed method; thus, we only present the simple visual and quantitative comparisons for the FCM-L1 approach. All examples are mainly divided into two categories, one is for synthetic images that may be corrupted by random noise, and the other is for real images. Besides, all tests are implemented in MATLAB(R2017a) on a laptop of 8Gb RAM and Intel(R) Core(TM) i5 CPU: @3.10 GHz.

Parameter setting: In our experiments, it is reasonable to stop the iteration if the following relative total energy (ReEng) is smaller than a pre-defined positive threshold tol, i.e.,

$$\text{ReEng} = \frac{|E - E_{old}|}{|E|} < tol, \tag{24}$$

where tol is set as 1×10^{-5} in our experiments. The bigger tol will lead to the faster stopping of the iterative method. E and E_{old} are with the same definitions as in Algorithm 3. Also, we set 4 levels for the multigrid method, i.e., $J = 3$, which means that the multigrid method will start from $(J-3)$th level (viewed as the coarsest level) and end in Jth level (the finest level). Therefore, the pixel numbers for each $P_{i,j}^{col}$ on the coarsest and the finest levels are respectively 8×8 and 1×1 (denoted as $8 \times 8 \rightarrow 1 \times 1$). Note that we also implement thresholding (with a value of 0.5 for all experiments) to the outcomes of the max-flow and the proposed

methods, which makes sure the produced results being piecewise constant [22]. Moreover, the maximum number *maxIter* of outer iteration in Algorithm 3 is set as 150. Besides, the parameters in our method are easy to select because the proposed approach is not sensitive to the associated parameters. Actually, choosing suitable parameters is always a difficulty in many image algorithms. Empirical tuning is a popular way to determine parameters; thus, we take this way to obtain the parameters in our work.

Smoothing Implementation: Before the experiments, a smoothing implementation is taken to smooth the singularity of the model. Note that the dual form of max-flow problem (6) in the multigrid method is non-smoothness, since the corresponding Euler-Lagrange equation

$$\alpha \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) - (f_1 - f_2) = 0, \tag{25}$$

contains a singularity of $|\nabla u| = 0$. To address this issue, a common strategy that incorporates a small constant δ to eliminate the singularity is used, which makes the new minimization problem become

$$\min_{u \in [0,1]} F_\delta(u) = \int_\Omega f_1 u + f_2(1-u) + \alpha \left(\sqrt{|\nabla u|^2 + \delta^2} - \delta \right) dx. \tag{26}$$

Therefore, in practical implementation, the F function in Algorithm 1, 2 and 3 should be replaced by F_δ that is defined in (26). Additionally, the δ is fixed as 5×10^{-2} in the experiments.

6.1 Segmentation Results

In this section, we report the total energy changes for the max-flow method and the proposed method in Fig. 6, since they are all to solve the same model.[1] It indicates that both our method and the max-flow based method get similar converged total energy. Particularly, the reason why the final converged total energy of the multigrid method is slightly bigger than that of the max-flow approach is that the multigrid method uses the smoothed energy (26). In contrast, the max-flow approach is only applied to the original TV minimization problem.

In Fig. 7, we test the performance of computational time for both the max-flow method and our method with the increased image size on a noisy synthetic image (see Fig. 7a). In Fig. 7b, the image size is increased from 100×100 to 1000×1000. When the image content is simple, and the image size is small (see e.g., smaller than 300×300), the max-flow method uses less computational time than the given approach, while our method will be significantly faster than the max-flow method when the image content is complex and the image size is bigger.

In Fig. 8, we take some simple synthetic and real images for the test of multiphase image segmentation. They include one synthetic image with the noise of unknown level (i.e., the first image) and five real images without any corruption (i.e., the second to the sixth image). In particular, these test images are first pre-processed by the region force method [42,53] (see the second column in Fig. 8). The max-flow and the proposed method are both implemented based on these pre-processed images for fair comparisons. From the last four columns of Fig. 8, it is easy to see that the four compared methods perform similarly well for the synthetic image that

[1] The models used in the FCM-L1 and SLaT methods are different, thus it is not meaningful to discuss the energy change.

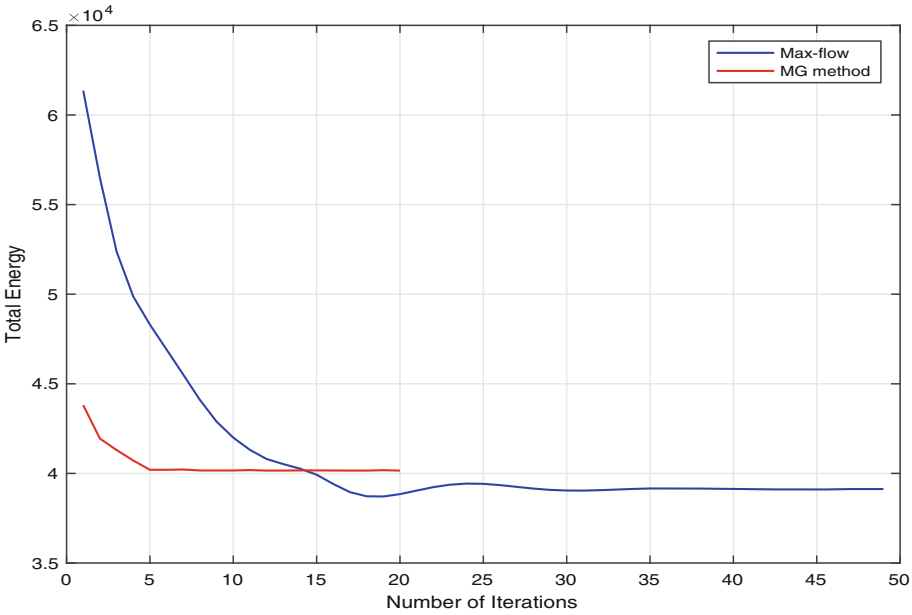
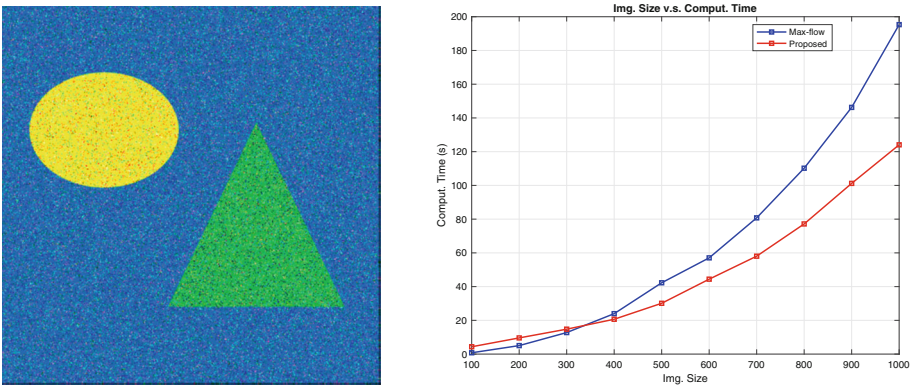


Fig. 6 The comparison of total energy for the max-flow method and the multigrid method implemented on a test image



(a) **(b)**

Fig. 7 The comparison of computational time with increased image sizes. Here we synthesize images with the sizes from 100×100 to 1000×1000. **a** An example with 3 phases (i.e., $K = 3$); **b** The comparison of computational time for both compared methods

only contains simple image content and some unknown noise. For the segmentation of real-world images (i.e., the last five examples), the proposed method produces competitive visual results compared with three other approaches. For the segmentation of small objects, the max-flow method and our method are slightly better. For instance, for the real image "zebra", the max-flow method and the proposed method could segment the black and white stripes well while other two methods are much less accurate. Another example is the "strawberry", in which the results by our method and the max-flow method show a better visual performance

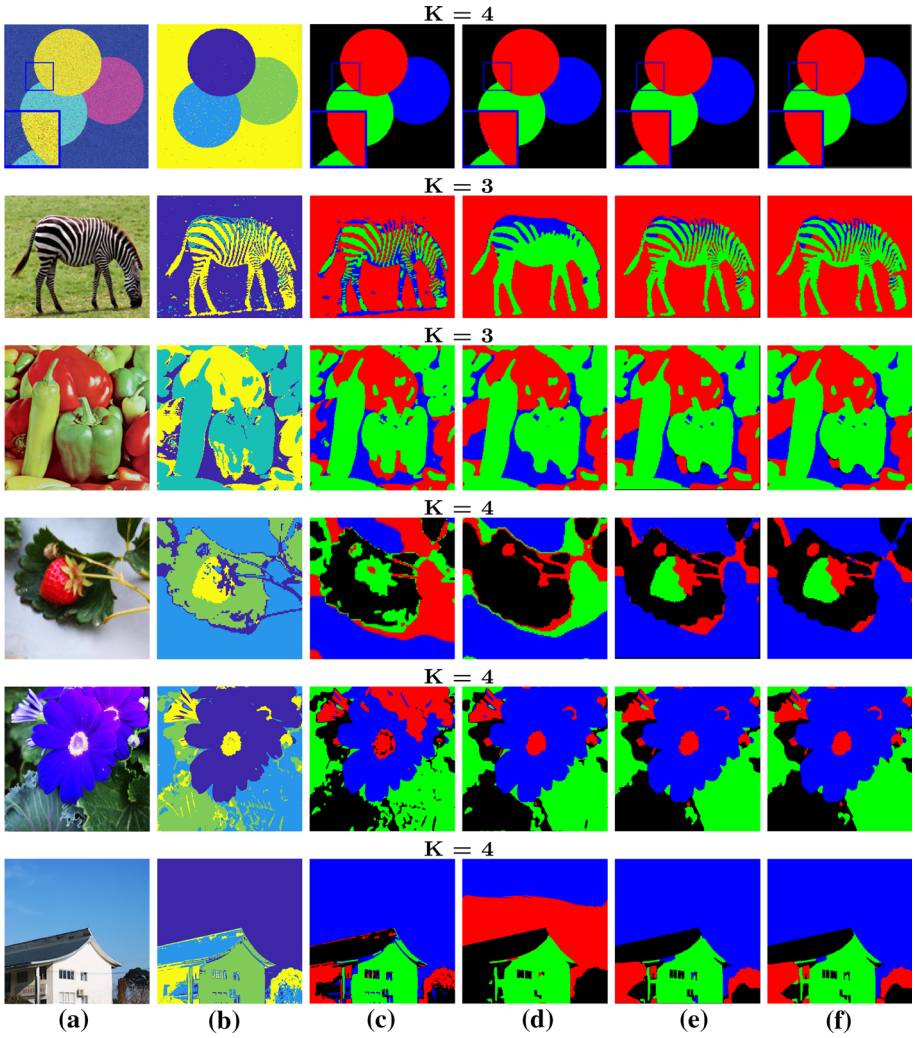


Fig. 8 The results of K-phase segmentation by various methods on the synthetic image "ball" (size of 256 × 256 × 3), five real-world images "zebra" (size of 195 × 290 × 3), "pepper" (size of 256 × 256 × 3), "strawberry" (size of 135 × 115 × 3), "flower" (size of 500 × 500 × 3) and "house" (size of 474 × 474 × 3). From left to right are **a** images to be segmented; **b** initial segmentation using the region force as the prior; **c** the results by the FCM-L1 method [2]; **d** the results by the SLaT method [52]; **e** The results by the max-flow method [24]; **f** The results by the multigrid method

than the other two approaches. The observation on other examples in comparison also confirm our conclusion.

Table 3 reports the computational time of different approaches, which may be affected by the image size, image type, and image noise, etc. From Table 3, it is clear that the multigrid method is faster than the FCM-L1 method and the max-flow method for all compared examples. Note that the multigrid method could get a larger leading than the max-flow method with the bigger image size and the more complex real image structure (see the last two examples in Table 3). This conclusion also holds for the FCM-L1 approach. Especially for all the examples, the three stages' SLaT method's computational time is faster than the other methods. It

Table 3 The comparison of computational time for the compared methods (unit: second)

Example	FCM-L1 [2]	SLaT [52]	Max-flow [24]	Multigrid
Ball (K = 4)	9.12	2.05	9.65	8.24
Zebra (K = 3)	3.79	1.58	3.21	2.75
Pepper (K = 3)	14.67	3.40	9.99	9.60
Strawberry (K = 4)	3.58	1.19	5.71	5.16
Flower (K = 4)	52.50	16.46	110.23	51.37
House (K = 4)	80.87	10.40	95.57	47.56

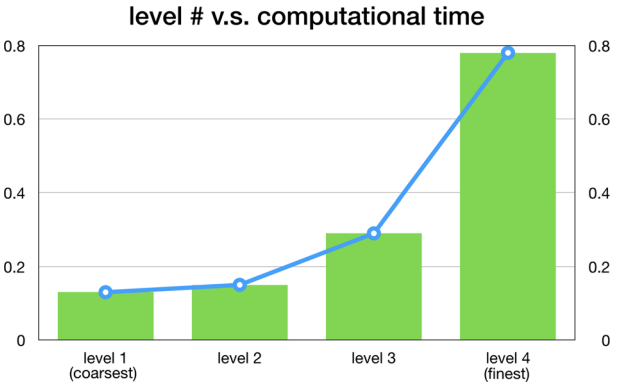


Fig. 9 The computational time with the increased level number for Backslash-cycle multigrid method on a test image "ball"

is mainly due to the fast algorithm (e.g., primal-dual) and the closed-form operations used in the method. However, we observe that SLaT has problems to segment real complex images, c.f. Fig. 8 2nd-6th examples. The simple smoothing and threshold procedures in SLaT make it fast to be solved, but this also leads to missing details for real complex images. In summary, we could conclude that the multigrid method could cost less computational time than the FCM-L1 method and the max-flow method, especially with a bigger image size and a complex real example. Although the SLaT approach gets less computational time than other methods, it performs unsatisfactorily on the real complex examples. In Fig. 9, we also show the computational time of each level of the multigrid method. From the result, it is easy to see that the computational time will increase with the increased level number.

6.2 More Discussions

It is observed that is often better to skip some of the coarser meshes for the multigrid method for image segmentation. In this section's tests, we choose the coarsest mesh with each element containing 8×8 finest elements. We denote this setting as $8 \times 8 \rightarrow 1 \times 1$). In Fig. 10, it shows the performance of computational time for our methods with the increased element size on the coarsest level. Actually, the level number can be computed by the coarsest element size. For instance, if it is $8 \times 8 \rightarrow 1 \times 1$ which means the level number is $\log_2 8 - \log_2 1 + 1 = 4$. From Fig. 10, the element size on coarsest level increases from 2×2 to 16×16 which indicates the level number should increase from 2 to 5. The computational time is decreased

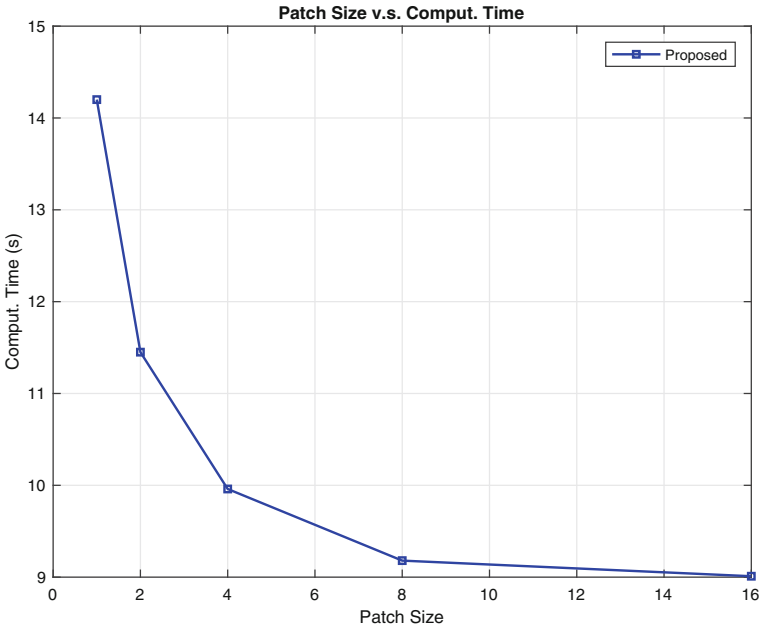


Fig. 10 The comparison of computational time with different element sizes on the coarsest level (from 1×1 to 16×16), i.e., $1 \times 1 \rightarrow 1 \times 1, 2 \times 2 \rightarrow 1 \times 1, \cdots, 16 \times 16 \rightarrow 1 \times 1$. Note that the default setting is $8 \times 8 \rightarrow 1 \times 1$ where 1×1 represents the finest level, and the level number is actually determined by element size of the corasest mesh. For instance, if it is $8 \times 8 \rightarrow 1 \times 1$ which means the level number is $\log_2 8 - \log_2 1 + 1 = 4$. The test is implemented on the third example in Fig. 8

from about 14 to 9 s, demonstrating that the multigrid method uses less computational time (or faster speed) than the direct method applied to the finest level. If we use even coarser mesh, the multigrid method's computational time will not reduce further, which indicates that it is not necessary to use very coarse mesh for our multigrid algorithm.

Our tests showed that we might also skip the computation on the finest mesh to get better computing time. For some real images, it is unnecessary to apply the multigrid method to the finest grid, i.e., the finest level with element size 1×1. We only need to use the multigrid algorithm on levels with element sizes 2×2 to the coarsest mesh. This strategy could significantly reduce computational time and get good segmentation results. Figure 11 shows the segmentation results with $K = 2$ by our method with the levels of element size ranging from $8 \times 8 \rightarrow 1 \times 1$ and $8 \times 8 \rightarrow 2 \times 2$, respectively. From the figure, it is clear that using $8 \times 8 \rightarrow 1 \times 1$ (Fig. 10b) will take more computational time than that of using $8 \times 8 \rightarrow 2 \times 2$ (see Fig. 10c). Actually, the segmentation result using $8 \times 8 \rightarrow 2 \times 2$ is just as good.

7 Conclusions

In this paper, a multiphase image segmentation method via the min-cut minimization problem was proposed under the framework of the multigrid (MG) method. We first transferred the min-cut on each level of the multigrid method to its max-flow problem equivalent form, then solved the equivalent form by the golden section method. A classical multigrid type of

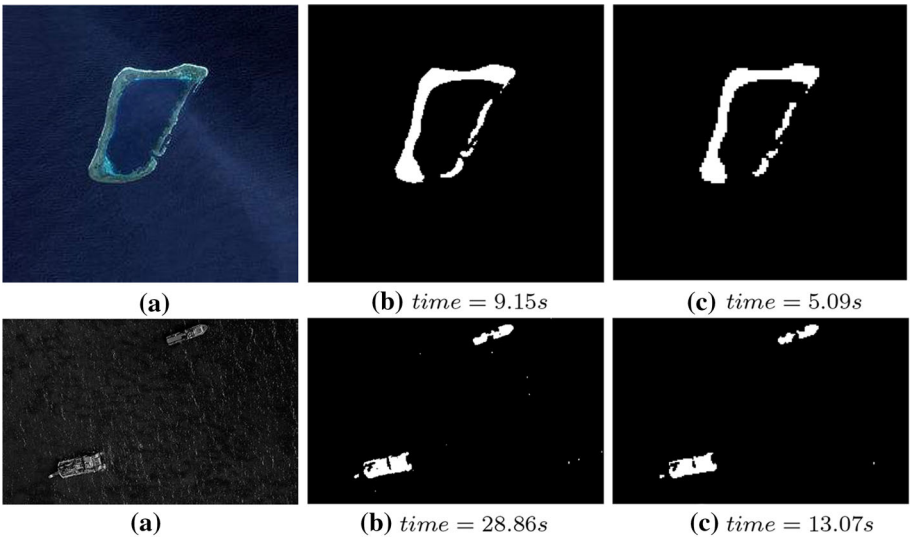


Fig. 11 The results of real images. **a** Real image; **b** results by our method with patch size $8 \times 8 \rightarrow 1 \times 1$; **c** results by our method with patch size $8 \times 8 \rightarrow 2 \times 2$

so-called Backslash-cycle was selected to address the sub-minimization problems. Extensive experiments demonstrated the effectiveness of the proposed method, e.g., the convergence and efficiency of the given approach, the competitive multiphase segmentation performance, especially the efficient multiphase segmentation of real images.

Acknowledgements The work of Xue-Cheng Tai was supported by RG(R)-RC/17-18/02-MATH, HKBU 12300819, NSF/RGC Grant N-HKBU214-19 and RC-FNRA-IG/19-20/SCI/01. The work of Liang-Jian Deng was partially supported by National Natural Science Foundation of China Grants 61702083, 61772003 and Key Projects of Applied Basic Research in Sichuan Province Grants 2020YJ0216. Besides, the work of Ke Yin was supported by National Natural Science Foundation of China Grant 11801200.

References

1. Pock, T., Antonin, C., Cremers, E., Bischof, H.: A convex relaxation approach for computing minimal partitions. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
2. Li, F., Osher, S., Qin, J., Yan, M.: A multiphase image segmentation based on fuzzy membership functions and L1-norm fidelity. J. Sci. Comput. **69**, 82–106 (2016)
3. Chan, R., Yang, H., Zeng, T.: A two-stage image segmentation method for blurry images with Poisson or multiplicative gamma noise. SIAM J. Imaging Sci. **7**(1), 98–127 (2014)
4. Chan, R., Lanza, A., Morigi, S., Sgallari, F.: Convex non-convex image segmentation. Numerische Mathematik **138**(3), 635–680 (2018)
5. Cai, X., Chan, R., Zeng, T.: A two-stage image segmentation method using a convex variant of the Mumford–Shah model and thresholding. SIAM J. Imaging Sci. **6**(1), 368–390 (2013)
6. Cai, X., Chan, R., Morigi, S., Sgallazi, F.: Vessel segmentation in medical imaging using a tight-frame based algorithm. SIAM J. Imaging Sci. **6**(1), 464–486 (2013)
7. Cai, X., Chan, R., Schonlieb, C.-B., Steidl, G., Zeng, T.: Linkage between piecewise constant Mumford–Shah model and Rudin–Osher–Fatemi model and its virtue in image segmentation. SIAM J. Sci. Comput. **41**(6), B1310–B1340 (2019)
8. Tan, L., Pan, Z., Liu, W., Duan, J., Wei, W., Wang, G.: Image segmentation with depth information via simplified variational level set formulation. J. Math. Imaging Vis. **60**, 1–17 (2018)
9. Spencer, J., Chen, K., Duan, J.: Parameter-free selective segmentation with convex variational methods. IEEE Trans. Image Process. **28**(5), 2163–2172 (2019)
10. Mumford, D., Shah, J.: Optimal approximations by piecewise smooth functions and associated variational problems. Commun. Pure Appl. Math. **42**, 577–685 (1989)
11. Zhu, S., Yuille, A.: Region competition: unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **18**, 884–900 (1996)
12. Chan, T.F., Vese, L.A.: Active contours without edges. IEEE Trans. Image Process. **10**, 266–277 (2001)
13. Osher, S., Sethian, J.: Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations. J. Comput. Phys. **79**, 12–49 (1988)
14. Aubert, G., Kornprobst, P.: Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations, Vol. 147. Springer (2006)
15. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. Int. J. Comput. Vis. **22**, 61–79 (1997)
16. Guo, W., Qin, J., Tari, S.: Automatic prior shape selection for image segmentation. Res. Shape Model. (2015)
17. Tan, L., Pan, Z., Liu, W., Duan, J., Wei, W., Wang, G.: Image segmentation with depth information via simplified variational level set formulation. J. Math. Imaging Vis. **60**(1), 1–17 (2018)
18. Bresson, X., Esedoglu, S., Vandergheynst, P., Thiran, J.-P., Osher, S.: Fast global minimization of the active contour/snake model. J. Math. Imaging Vis. **28**, 151–167 (2007)
19. Houhou, N., Thiran, J., Bresson, X.: Fast texture segmentation model based on the shape operator and active contour. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–8 (2008)
20. Mory, B., Ardon, R.: Fuzzy region competition: a convex two-phase segmentation framework. In: Scale Space and Variational Methods in Computer Vision, pp. 214–226. Springer (2007)
21. Lie, J., Lysaker, M., Tai, X.-C.: A binary level set model and some applications to Mumford–Shah image segmentation. IEEE Trans. Image Process. **15**, 1171–1181 (2006)
22. Chan, T., Esedoglu, S., Nikolova, M.: Algorithms for finding global minimizers of image segmentation and denoising models. SIAM J. Appl. Math. **66**, 1632–1648 (2006)
23. Chambolle, Antonin, Cremers, D., Pock, T.: A convex approach to minimal partitions. SIAM J. Imaging Sci. **5**(4), 1113–1158 (2012)
24. Yuan, J., Bae, E., Tai, X.-C.: A study on continuous max-flow and min-cut approaches. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2010)
25. Yuan, J., Bae, E., Tai, X.-C., Boykov, Y.: A continuous max-flow approach to Potts model. In: European Conference on Computer Vision (ECCV), pp. 379–392 (2010)

26. Bae, E., Yuan, J., Tai, X.-C., Boykov, T.: A fast continuous max-flow approach to non-convex multilabeling problems. In: Efficient Global Minimization Methods for Variational Problems in Imaging and Vision (2011)
27. Yuan, J., Bae, E., Tai, X.-C., Boykov, Y.: A spatially continuous max-flow and min-cut framework for binary labeling problems. Numerische Mathematik **66**, 1–29 (2013)
28. Bae, E., Lellmann, J., Tai, X.-C.: Convex relaxations for a generalized Chan–Vese model. In: Energy Minimization Methods in Computer Vision and Pattern Recognition, pp. 223–236. Springer (2013)
29. Bae, E., Tai, X.-C.: Efficient global minimization methods for image segmentation models with four regions. J. Math. Imaging Vis. **51**, 71–97 (2015)
30. Briggs, W., Henson, V., McCormick, S.: A Multigrid Tutorial. SIAM, Philadelphia (2000)
31. Donatelli, M.: A multigrid for image deblurring with Tikhonov regularization. Numer. Linear Algebra Appl. **12**, 715–729 (2005)
32. Chen, K., Tai, X.-C.: A nonlinear multigrid method for total variation minimization from image restoration. J. Sci. Comput. **33**, 115–138 (2007)
33. Español, M.: Multilevel Methods for Discrete Ill-Posed Problems: Application to Deblurring, PhD thesis, Department of Mathematicsm, Tufts University (2009)
34. Español, M., Kilmer, M.: Multilevel approach for signal restoration problems with Toeplitz matrices. SIAM J. Sci. Comput. **32**, 299–319 (2010)
35. Chen, K., Dong, Y., Hintermüller, M.: A nonlinear multigrid solver with line Gauss–Seidel–Semismooth–Newton smoother for the Fenchel predual in total variation based image restoration. Inverse Probl. Imaging **5**, 323–339 (2011)
36. Deng, L.-J., Huang, T.-Z., Zhao, X.-L.: Wavelet-based two-level methods for image restoration. Commun. Nonlinear Sci. Numer. Simul. **17**, 5079–5087 (2012)
37. Chumchob, N., Chen, K.: A robust multigrid approach for variational image registration models. J. Comput. Appl. Math. **236**, 653–674 (2011)
38. Badshah, N., Chen, K.: Multigrid method for the Chan–Vese model in variational segmentation. Commun. Comput. Phys **4**, 294–316 (2008)
39. Badshah, N., Chen, K.: On two multigrid algorithms for modeling variational multiphase image segmentation. IEEE Trans. Image Process. **18**, 1097–1106 (2009)
40. Deng, L.-J., Huang, T.-Z., Zhao, X.-L., Zhao, L., Wang, S.: Signal restoration combining Tikhonov regularization and multilevel method with thresholding strategy. J. Opt. Soc. Am. Opt. Image Sci. Vis. **30**, 948–955 (2013)
41. Yuan, J., Bae, E., Tai, X.-C., Boykov, T.: "A study on continuous max-flow and min-cut approaches", Technical report CAM10-61. UCLA, CAM (2010)
42. Wei, K., Yin, K., Tai, X.-C., Chan, T.F.: New region force for variational models in image segmentation and high dimensional data clustering. Ann. Math. Sci. Appl. **3**(1), 255–286 (2018)
43. Ishikawa, Hiroshi: Exact optimization for Markov random fields with convex priors. IEEE Trans. Pattern Anal. Mach. Intell. **25**, 1333–1336 (2003)
44. Darbon, J., Sigelle, M.: Image restoration with constrained total variation. Part II: levelable functions, convex priors and non-convex cases. J. Math. Imaging Vis. **26**(3), 277–292 (2006)
45. Darbon, J., Sigelle, M.: Image restoration with discrete constrained total variation part I: fast and exact optimization. J. Math. Imaging Vis. **26**(3), 261–276 (2006)
46. Bae, E., Yuan, J., Tai, X.-C., Boykov, Y.: A fast continuous max-flow approach to non-convex multilabeling problems. In: Efficient Algorithms for Global Optimization Methods in Computer Vision, pp. 134–154. Springer (2014)
47. Goldluecke, Bastian, Cremers, D.: Convex relaxation for multilabel problems with product label spaces. Comput. Vis. ECCV **2010**, 225–238 (2010)
48. Vese, L.A., Chan, T.F.: A new multiphase level set framework for image segmentation via the Mumford and Shah model. Int. J. Comput. Vis. **50**, 271–293 (2002)
49. Kiefer, J.: Sequential minimax search for a maximum. Proc. Am. Math. Soc. **4**, 502–506 (1953)
50. Condat, Laurent: Fast projection onto the simplex and the l1 ball. Math. Program. **158**(1–2), 575–585 (2016)
51. Chen, Y., Ye, X.: Projection onto a simplex. arXiv preprint arXiv:1101.6081 (2011)
52. Cai, X., Chan, R., Nikolova, M., Zeng, T.: A three-stage approach for segmenting degraded color images: smoothing, lifting and thresholding (SLaT). J. Sci. Comput. **72**(3), 1313–1332 (2017)
53. Yin, K., Tai, X.-C.: An effective region force for some variational models for learning and clustering. J. Sci. Comput. **74**, 175–196 (2018)