INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 8, Number 1, Pages 137–155

DOMAIN DECOMPOSITION METHODS WITH GRAPH CUTS ALGORITHMS FOR IMAGE SEGMENTATION

XUE-CHENG TAI AND YUPING DUAN

Abstract. Recently, it is shown that graph cuts algorithms can be used to solve some variational image restoration problems, especially connected with noise removal and segmentation. For very large size images, the usage for memory and computation increases dramatically. We propose a domain decomposition method with graph cuts algorithms. We show that the new approach costs effective both for memory and computation. Experiments with large size 2D and 3D data are supplied to show the efficiency of the algorithms.

Key words. Multiphase Mumford-Shah, Graph cuts, Image segmentation, Domain decomposition

1. Introduction

Segmentation is one of the fundamental problems in image processing and computer vision tasks. The result of image segmentation is a set of contours extracted from the image, or a set of regions that collectively cover the entire image. Mumford and Shah model [26] is an effective tool for region based image segmentation. This model is robust to noise and can segment objects without edges. However, the minimization problem is difficult to solve numerically.

The level set method [13, 27] was first introduced to solve the Mumford-Shah functional by Chan and Vese in [7, 33]. Chan and Vese model achieves great success in image segmentation due to its advantages in obtaining large convergence range and handling the topological changes. A lot further works of Chan and Vese model were done in [23, 28]. Some variants of the level set method, so-called "Piecewise Constant Level Set Method" (PCLSM), were proposed in [24, 25, 29]. This PCLSM can identify several interfaces by one single level set function, which makes it easier to solve the Mumford-Shah model.

Traditionally, methods based on gradient descent are often used for solving the Mumford-Shah models, see [24, 25, 29]. These methods are normally slow and difficult to find global minimizers. Recently, many works have been done on applying graph cuts algorithms for image segmentation [5, 3, 21, 12, 34]. They are proven to be more efficient for solving this kind of energy minimization problems. The connection between graph cuts and variational problems has been established in [2, 18, 10, 4]. For Mumford-Shah segmentation, some work using graph cuts optimization for two-phase model has been done in [9] and [14]. For multiphase Mumford-Shah model, the methods of [5, 22, 18] can be adopted for solving the corresponding energy problems. In this work, we shall follow the approach given in [1]. In [1], the authors used the level set formulation of Mumford-Shah model [27]

Received by the editors May 5, 2010.
2000 *Mathematics Subject Classification.* 65N55, 65F10, 68U10.
The authors would like to thank Professor Wenbing Tao for valuable discussions and constructive suggestions. The 3*D* CT scans are courtesy of Beijing Normal University. This work has been supported by MOE (Ministry of Education) Tier II project T207N2202 and IDM project NRF2007IDM-IDM002-010.

and adopted the graph construction method in [18, 19] to multiphase Mumford-Shah model. However, when the images become large and the number of phases increases, especially for 3D segmentation cases, both computational cost and memory usage increase greatly. In this work we try to find some remedies for these difficulties and show that we could get some algorithms which have quite high efficiency as well as low memory usage. We propose a method combining the domain decomposition method with graph cuts algorithms.

The paper is organized as follows. In section 2, we review the PCLSM and its applications to the Mumford-Shah model. In section 3, we review the graph cuts algorithm of [1] to the multiphase Mumford-Shah model. In Section 4, we combine the domain decomposition methods with the graph cuts idea to solve the Mumford-Shah model. Some implementation details are supplied in Section 5. Finally, in Section 6, we carry out some experiments by our algorithms and compare the results with the original graph cuts algorithm.

2. Mumford-Shah model with PCLSM

2.1. Mumford-Shah model. The Mumford-Shah model is a well known model for image segmentation problem [26]. In the model, Ω is a bounded domain and $u^0(x)$ is the input image. We search for n interfaces Γ_i and an approximation image u by minimizing the following energy functional

$$E(u,\Gamma_i) = \int_\Omega (u-u^0)^2 dx + \mu \int_{\Omega\setminus\bigcup_i \Gamma_i} |\nabla u|^2 dx + \sum_{i=1}^{n} \gamma \int_{\Gamma_i} ds. \tag{1}$$

where μ and γ are nonnegative constants and $\int_{\Gamma_i} ds$ is the length of the boundary of interfaces Γ_i. The most popular way to solve this minimization problem is applying the level set method [7], especially the piecewise constant level set Mumford-Shah model. For such case, the second term vanishes in the minimization functional.

2.2. Piecewise constant level set method. In [24, 25, 29], the piecewise constant level set method (PCLSM) was proposed and applied to the Mumford-Shah model. The main idea of PCLSM is to seek a partition of the domain Ω into n subdomains Ω_i, $i = 1, 2, \cdots, n$. The essential idea is to use a piecewise constant level set function ϕ to identify the subdomains

$$\phi = i \quad in \quad \Omega_i. \tag{2}$$

Once the function ϕ is identified, we can construct the corresponding characteristic functions for each subdomain Ω_i as

$$\psi_i = \frac{1}{\alpha_i} \prod_{j=1, j\neq i}^{n} (\phi - j), \quad \text{with} \quad \alpha_i = \prod_{k=1, k\neq i}^{n} (i-k). \tag{3}$$

If ϕ is defined as in (2), we have $\psi_i(x) = 1$ for $x \in \Omega_i$, otherwise we have $\psi_i(x) = 0$. Based on these characteristic functions, we can extract the geometrical information of the boundaries of the subdomains $\{\Omega_i\}_{i=1}^n$. For example, the length of the interfaces surrounding each subdomain Ω_i, $i = 1, 2, \cdots, n$, should be

$$Length(\partial\Omega_i) = \int_\Omega |\nabla(\psi_i)| dx. \tag{4}$$

For some given values $c_i, i = 1, 2, \cdots n$, define

$$u = \sum_{i=1}^{n} c_i \psi_i. \tag{5}$$

We have $u = c_i$ in the corresponding subdomain Ω_i, if ϕ satisfies (2). In the next subsection, we shall use this idea for image segmentation with the Mumford-Shah model.

2.3. The minimization problem. We assume u is a piecewise constant function as given in (5) and ϕ is the corresponding level set function (2). The multiphase piecewise constant Mumford-Shah model is to solve the following minimization problem

$$\min_{\mathbf{c}\in\mathbb{R}^n,\phi\in\{1,2,\cdots,n\}} E(\mathbf{c},\phi), \qquad E(\mathbf{c},\phi) = \int_\Omega (u-u^0)^2 dx + \gamma \sum_{i=1}^n \int_\Omega |\nabla\psi_i| dx. \tag{6}$$

We use total variation (TV) of the characteristic function to replace the last term of the Mumford-Shah functional, measuring the length of the interfaces. Such an approach has also been used in other segmentation models in [8, 20]. It is easy to see that

$$\phi = \sum_{i=1}^n i\psi_i(\phi), \quad \nabla\psi_i = \psi_i'(\phi)\nabla\phi.$$

Thus, there exist two constants $\alpha_1(n) > 0, \alpha_2(n) > 0$, such that

$$\alpha_1(n)\int_\Omega |\nabla\phi| dx \le \sum_{i=1}^n \int_\Omega |\psi_i(\phi)| dx \le \alpha_2(n)\int_\Omega |\nabla\phi| dx. \tag{7}$$

Unless "symmetry" is a crucial issue for the segmentation problem, we replace the regularization term in $E(\mathbf{c},\phi)$ by an equivalent functional and solve the following minimization problem

$$\min_{\mathbf{c}\in\mathbb{R}^n,\phi\in\{1,2,\cdots,n\}} E(\mathbf{c},\phi), \qquad E(\mathbf{c},\phi) = \int_\Omega (u-u^0)^2 dx + \gamma \int_\Omega |\nabla\phi| dx. \tag{8}$$

This functional is the Mumford-Shah model we used in the paper. In [24, 25, 29], the constrained optimization problem (8) was solved by finding the saddle point of the corresponding augmented Lagrangian functional. In these methods, some iterative numerical methods are used to solve the corresponding Euler-Lagrange equations, such as gradient decent time marching scheme. In the next section, we shall construct a graph and solve the minimization problem (8) by the graph cuts algorithms as in [18, 1].

3. Graph cuts for multiphase Mumford-Shah Model

Instead of solving the Euler-Larange equation, graph cuts algorithms have been proposed to solve the minimization problem (8). We give a review of this algorithm in the following.

3.1. Background on graph cuts. The graph cuts algorithm is an established powerful method to minimize certain kinds of energy functionals. A directed capacitated graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a set of vertices $\mathcal{V}$ and directed edges $\mathcal{E}$. There are two special vertices in the graph, i.e., the source s and the sink t. A cut on graph $\mathcal{G}$ partitions the vertices into two disjoint groups S and T such that $s \in S$ and $t \in T$. The cost of the cut is the sum of capacities of all edges that go from S to T

$$c(S,T) = \sum_{u\in S, v\in T, (u,v)\in\mathcal{E}} c(u,v). \tag{9}$$

We focus on finding a cut with the smallest cost $c(S,T)$, namely the minimal cut. To solve the minimal cut problem, there are mainly two groups of algorithms:

Goldberg-Tarjan style "push-relabel" methods [17] and Ford-Fulkerson style "augmenting paths" [16]. In our paper, we use the augmenting paths method [3].

3.2. Discretization of energy functional. Assume we want to segment an $M \times N$ image into $n(n \geq 2)$ phases. Let $\mathcal{P}$ denotes the index set of the pixels, i.e. ,

$$\mathcal{P} = \{(i,j) \quad | \quad i \in 1,\ldots,M, j \in 1,\ldots,N\}. \tag{10}$$

There are two different ways to discretize the TV term of the functional (8), i.e., isotropic and anisotropic. Since the isotropic TV is not graph representable, we consider anisotropic discretization of the TV term. The anisotropic discretization depends on the neighbor pixels adopted to represent the TV term. In this paper, we consider 4 and 8 neighbors for 2D images, c.f. [2, 14, 11]

$$TV^4(\phi) = \sum_{i,j} |\phi_{i+1,j} - \phi_{i,j}| + |\phi_{i,j+1} - \phi_{i,j}|, \tag{11}$$

$$TV^8(\phi) = TV^4(\phi) + \frac{1}{\sqrt{2}} \sum_{i,j} (|\phi_{i+1,j+1} - \phi_{i,j}| + |\phi_{i+1,j-1} - \phi_{i,j}|). \tag{12}$$

The data fidelity term can be discretized directly. For a given $p = (i,j) \in \mathcal{P}$, define

$$\mathcal{N}_4(p) = \{(i \pm 1, j), (i, j \pm 1)\} \cap \mathcal{P}, \tag{13}$$

$$\mathcal{N}_8(p) = \{(i \pm 1, j), (i, j \pm 1), (i \pm 1, j \pm 1)\} \cap \mathcal{P}. \tag{14}$$

Using these notations, the discrete version of (8) can be written as

$$E_d(\mathbf{c}, \phi) = \sum_{p \in \mathcal{P}} |u_p - u_p^0|^2 + \gamma \sum_{p \in \mathcal{P}, q \in \mathcal{N}_k(p)} w_{pq} |\phi_p - \phi_q|. \tag{15}$$

Above, $\mathcal{N}_k(p), k = 4, 8$, is defined as in (13)-(14) and w_{pq} is the corresponding weight for the discretized TV-term as in (11) and (12), see also [1]. u_p^0 is the intensity value of u^0 at $p \in \mathcal{P}$ and u_p is related to ϕ_p as in (5) and (3). We assume that the value of $c_i, i = 1, 2, \cdots n$ are known. For boundary points p, $\mathcal{N}_k(p)$ has less neighboring points.

By doing so, the minimization problem is transformed into discrete form which is graph representable. We can get the minimizer of (15) using the max-flow / min-cut algorithm.

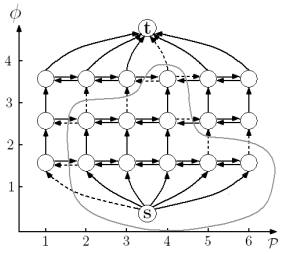
It is easy to extend the model to 3D problems. For example, we can use the neighborhood involved 6 neighbors for 3D cases and use the following term as the regularization term

$$TV^{3D,6}(\phi) = \sum_{i,j,k} (|\phi_{i+1,j,k} - \phi_{i,j,k}| + |\phi_{i,j+1,k} - \phi_{i,j,k}| + |\phi_{i,j,k+1} - \phi_{i,j,k}|).$$

Later, we shall also test on 3D segmentation problems and this regularization term has been used there. We can also add more neighboring points to approximate the length better.

3.3. Graph construction. Recently, a special graph has been constructed in [18, 1] to solve the multiphase Mumford-Shah model. In this subsection, we briefly review the essential ideas followed the instruction in [1].

To use graph cuts algorithm for the multiphase segmentation problems, we have to introduce an extra dimension, i.e., we construct graph with one dimension higher



(a)

1	3	3	4	2	2

(b)

FIGURE 1. (a) The graph corresponds to 1D signal of 6 grid points. We construct a 3 level grids for this 4 phase segmentation problem. The gray curve denotes the cut. (b) shows the values of the level set function ϕ at each grid point corresponding to the cut in (a).

than the original image. For a 2D image of size $M \times N$, we construct a graph in 3D containing $M \times N \times (n-1)$ vertices. Specifically, we have $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and

$$\mathcal{V} = \left\{ v_{p,l} \mid (p,l) \in \mathbb{R}^2 \times \mathbb{R} |\ p \in \mathcal{P},\ l \in \{1, \ldots, n-1\} \right\}. \tag{16}$$

The edges $\mathcal{E}$ are divided into two groups: $\mathcal{E}_D$ coresponds to the data fidelity term in (15) and $\mathcal{E}_R$ corresponds to the TV term in (15). They are defined, respectively, as

$$\mathcal{E}_D = \cup_{p \in \mathcal{P}} \left\{ (s, v_{p,1}) \cup_{l=1}^{n-2} (v_{p,l}, v_{p,l+1}) \cup (v_{p,n-1}, t) \right\}, \tag{17}$$

$$\mathcal{E}_R = \{ (v_{p,l}, v_{q,l}) | p \in \mathcal{P}, q \in \mathcal{N}_k(p), l \in 1, \ldots, n-1 \}. \tag{18}$$

In Fig.1, the graph for a $1D$ signal with 4-phase segmentation is shown. The edges in $\mathcal{E}_D$ are illustrated as the vertical arrows while the edges in $\mathcal{E}_R$ are illustrated as the horizontal arrows in Fig.1. A cut is called admissible if it only serves one vertical edge for each $p \in \mathcal{P}$, c.f., [1]. In order to exclude non-adimissible cut, we introduce an artificial constant $\sigma > 0$ and define the capacity of the edges as

$$c(s, v_{p,1}) = |u_p^0 - c_1|^2 + \frac{\sigma}{MN}, \quad \forall p \in \mathcal{P}, \tag{19}$$

$$c(v_{p,l}, v_{p,l+1}) = |u_p^0 - c_{l+1}|^2 + \frac{\sigma}{MN}, \forall p \in \mathcal{P}, \quad \forall l \in 1, \ldots n-2, \tag{20}$$

$$c(v_{p,n}, t) = |u_p^0 - c_n|^2 + \frac{\sigma}{MN}, \quad \forall p \in \mathcal{P}, \tag{21}$$

$$c(v_{p,l}, v_{q,l}) = \gamma \cdot w_{pq}, \qquad \forall p \in \mathcal{P}, \forall q \in \mathcal{N}_k(p), \forall l \in 1, \ldots n-1. \tag{22}$$

In the above, γ is the regularization parameter, w_{pq} is the weight for the discretization of the TV-norm and $\mathcal{N}_k(p)$ is the set containing the neighbors of $p \in \mathcal{P}$ used in the discretization.

After adding all edges to the graph, we can solve the minimization problem (15) by using the max-flow / min-cut algorithm. We emphasize that the segmentation problem is transferred from the size of $M \times N$ to the size of $M \times N \times (n-1)$.

3.4. An iterative segmentation scheme. In the last section, we show that graph cuts algorithms can be used to solve the Mumford-Shah minimization problem when the value of **c** is known. For minimization problem (8), we also need to estimate the **c** value and the following algorithm – Algorithm 1, is rather robust and converges fast.

Algorithm 1 (Graph cuts segmentation algorithm)
Choose initial values for $\mathbf{c^0}$, set $l = 0$.
while $(\|\mathbf{c}^l - \mathbf{c}^{l-1}\| > tol)$

(1) Use graph cuts to estimate ϕ^{l+1} from

$$\phi^{l+1} = arg \min_{\tilde{\phi}} E_d(\mathbf{c}^l, \tilde{\phi}), \tag{23}$$

(2) Compute the characteristic functions $\{\psi_k^{l+1}\}_{k=1}^n$ from ϕ^{l+1}, c.f., (3).
(3) Update $\mathbf{c}^{l+1}$ by

$$c_k{}^{l+1} = \frac{\sum_{p\in\mathcal{P}} u_p^0 \psi_{k,p}}{\sum_{p\in\mathcal{P}} \psi_{k,p}}, \qquad k = 1, \ldots, n. \tag{24}$$

(4) Update $l \leftarrow l + 1$.

The initial values $\mathbf{c}^0$ are computed very efficiently by the isodata algorithm, see [32]. For segmentation problems, the above iterative procedure normally converges in about 5-6 iterations. Compared with traditional gradient decent methods, it is normally 500 times faster for relatively large size 2D images we have tested, see [1]. However, when the image size is very large, the memory requirement and computational cost become a challenge problem.

4. Graph cuts algorithms with domain decomposition

As we discussed, when the images become large, the computational and memory cost of the multiphase graph cuts algorithm increases greatly. This causes problems for some data set with very large size, especially for 3D applications. Therefore, we consider to use a domain decomposition method to overcome these difficulties.

Domain decomposition method is an efficient tool in large-scale computation and has been used to solve PDE problems [6, 15, 30, 31]. In [31], it is proven domain decomposition can be applied to general convex minimization problems.

As was done in [31], we decompose the image domain into four kinds of regions. There is an example of domain decomposition shown in Fig.2. Then we use similar iterative segmentation scheme to solve subproblems over the subdomains of each region.

Ω_1	Ω_2	Ω_1	Ω_2	Ω_1
Ω_3	Ω_4	Ω_3	Ω_4	Ω_3
Ω_1	Ω_2	Ω_1	Ω_2	Ω_1
Ω_3	Ω_4	Ω_3	Ω_4	Ω_3
Ω_1	Ω_2	Ω_1	Ω_2	Ω_1

FIGURE 2. An example of domain decomposition with 25 subdomains

4.1. Non-overlapping domain decomposition. First, we consider the non-overlapping domain decomposition method. We assume Ω has been decomposed into 4 kinds of non-overlapping subdomains. The subdomains intersect only on

their interfaces, see Fig.2. We denote $\mathcal{P}_i \subset \mathcal{P}$, $i = 1, 2, 3, 4$, the index sets for the grid points of the subdomains, c.f. (10). Corresponding to each subdomain, we define the energy functional

$$E_d^i(\mathbf{c}, \phi) = \sum_{p \in \mathcal{P}_i} |u_p - u_p^0|^2 + \gamma \sum_{p \in \mathcal{P}_i, q \in \mathcal{N}_k(p) \cap \mathcal{P}_i} w_{p,q} |\phi_p - \phi_q|. \tag{25}$$

The non-overlapping algorithm can be written as follows:

Algorithm 2(Non-overlapping domain decomposition)
Choose initial values for $\mathbf{c^0}$, set $l = 0$.
While $(\|\mathbf{c}^l - \mathbf{c}^{l-1}\| > tol)$

(1) For $i = 1, 2, 3, 4$, use a graph cuts algorithm to estimate $\phi_{|\Omega_i}^{l+1}$ from

$$\phi_{|\Omega_i}^{l+1} = arg \min_{\tilde{\phi}} E_d^i(\mathbf{c}^l, \tilde{\phi}). \tag{26}$$

(2) Compute the characteristic functions $\{\psi_k^{l+1}\}_{k=1}^n$ from ϕ^{l+1}, c.f., (3).
(3) Update $\mathbf{c}^{l+1}$ according to the following discrete formula for $\mathbf{c}$

$$c_k^{l+1} = \frac{\sum_{p \in \mathcal{P}} u_p^0 \psi_{k,p}^{l+1}}{\sum_{p \in \mathcal{P}} \psi_{k,p}^{l+1}}, \qquad k = 1, \ldots, n. \tag{27}$$

(4) Update $l \leftarrow l + 1$.

Here and later, we denote $\phi_{|\Omega_i}$ be the value of ϕ in Ω_i. For minimization problem (26), we only need to use graph cuts algorithms to find the values of ϕ^{l+1} in Ω_i. Each Ω_i contains many disjoint subdomains, i.e., $\Omega_i = \cap_j \Omega_{i,j}$. As the subproblems over $\Omega_{i,j}$ are independent of each other, we can use the graph cuts algorithms to solve the subdomain problems simultaneously. If we have parallel computers, these subdomain problems can be solved in parallel. In our implementations, we just solve the subproblems one by one. Even so, the computational cost is reduced compared to solving by the graph cuts algorithm in the whole domain.

For a given $p \in \mathcal{P}_i$ on the boundary $\partial\Omega_i$ of Ω_i, the subdomain energy functional E_d^i only includes regularization terms related to $q \in \mathcal{N}_k(p) \cap \mathcal{P}_i$, i.e., the subdomain problems only regularize with points inside the subdomain. Thus, this will cause some errors compared with Algorithm 1. Due to the reason that the $c_i, i = 1, 2, \cdots n$ are computed globally, it shows that the non-overlapping algorithm has always been able to find a good segmentation despite this error.

4.2. Overlapping domain decomposition. In overlapping domain decomposition, the subdomains overlap with each other. Fig.3 dispatches the subdomains in our overlapping domain decomposition approach corresponding to the domain decomposition method presented in Fig.2. The dashed line denotes the boundary of the subdomains. In overlapping domain decomposition, we use the overlapping parts to influence the cuts of the interior parts in each subdomain. Therefore, the subdomains are no longer independent and have intimate relation with their neighbor subdomains in the segmentation. The overlapping size influences the convergence rate of the iterate process as analysed in [31]. Large overlapping size gives faster convergence for the iteration. However, it also leads to increased cost in solving the subdomain problems. A proper choice of the overlapping size is needed in order to get the best convergence.

As the subdomains have overlaps now, the corresponding index sets $\mathcal{P}_i$ also have overlaps. To explain the algorithm clearly, we need to introduce some notations.

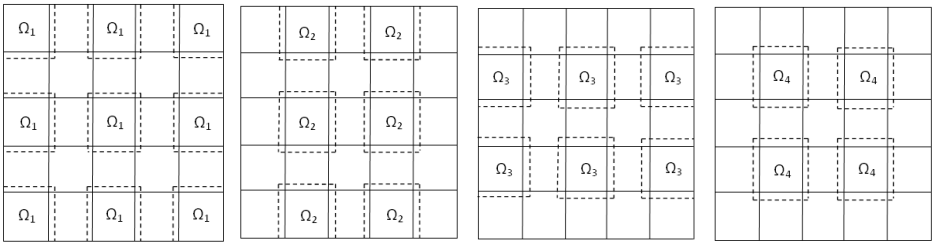


FIGURE 3. Four kinds of subdomains in the overlapping domain decomposition

We use Ω_i^0 to denote the interior grid points of Ω_i and $\partial\Omega_i$ to denote the boundary grid points of Ω_i. Correspondingly, $\mathcal{P}_i^0$ is the index set for Ω_i^0 and $\partial\mathcal{P}_i$ is the index set for $\partial\Omega_i$. Let

$$E_d^i(\mathbf{c},\phi) = \sum_{p\in\mathcal{P}_i^0} |u_p - u_p^0|^2 + \gamma \sum_{p\in\mathcal{P}_i^0, q\in\mathcal{N}_k(p)} w_{p,q}|\phi_p - \phi_q|. \tag{28}$$

The corresponding overlapping domain decomposition algorithm is applied in the following:

Algorithm 3 (Overlapping domain decomposition)
Choose initial values for $\mathbf{c^0}$ and ϕ^0. Set $l = 0$.
While ($\|\mathbf{c}^l - \mathbf{c}^{l-1}\| > tol$)

(1) For $i = 1, 2, 3, 4$, let $\phi^{l+\frac{i}{4}} = \phi^{l+\frac{i-1}{4}}$ in $\Omega\backslash\Omega_i^0$ and use a graph cuts algorithm to estimate $\phi^{l+\frac{i}{4}}_{|\Omega_i^0}$ from

$$\phi^{l+\frac{i}{4}}_{|\Omega_i^0} = arg\min_{\tilde{\phi}} E_d^i(\mathbf{c}^l, \tilde{\phi}). \tag{29}$$

(2) Compute the characteristic functions $\{\psi_k^{l+1}\}_{k=1}^n$ from ϕ^{l+1}, c.f., (3).
(3) Update $\mathbf{c}^{l+1}$ according to the following discrete formula for $\mathbf{c}$

$$c_k^{l+1} = \frac{\sum_{p\in\mathcal{P}} u_p^0 \psi_{k,p}^{l+1}}{\sum_{p\in\mathcal{P}} \psi_{k,p}^{l+1}}, \qquad k = 1, \ldots, n. \tag{30}$$

(4) Update $l \leftarrow l + 1$.

However, as the subdomains overlap with each other, solving (29) is quite different from solving (26). The value of $\phi^{l+\frac{i}{4}}$ is equal to $\phi^{l+\frac{i-1}{4}}$ in $\Omega\backslash\Omega_i^0$ and thus has no need for computation. The value of $\phi^{l+\frac{i}{4}}$ in Ω_i^0 needs to be solved through (29). For a point $p \in \mathcal{P}_i^0$, $\mathcal{N}_k(p)$ may be outside Ω_i^0. However, this does not cause any problem for solving (29) as the value outside Ω_i^0 is already known. This will take care of the regularization between the subdomains. We shall comment on the details for the implementation for (29) in Section 5.

For this algorithm, we have, c.f. [31]

$$\begin{aligned} E_d(\mathbf{c}^{l+1},\phi^{l+1}) \le E_d(\mathbf{c}^l,\phi^{l+1}) \le E_d(\mathbf{c}^l,\phi^{l+3/4}) &\le E_d(\mathbf{c}^l,\phi^{l+1/2}) \\ &\le E_d(\mathbf{c}^l,\phi^{l+1/4}) \le E_d(\mathbf{c}^l,\phi^l). \end{aligned}$$

This guarantees the monotonicity of the cost functional and thus gives a robust algorithm.

5. Implementation of the algorithms

For the implementation of Algorithm 1, we just need to construct the graph defined in (16) and (17)-(18) and then add the capacity (costs) as given in (19)-(22). Theoretically, any $\sigma > 0$ is enough to guarantee that any minimum cuts is admissible, see [1]. Once the graph is constructed, we use the augmenting path algorithm to find the minimum cut.

The implementation of Algorithm 2 is also easy. For each subproblem, we construct the graph as we have done for Algorithm 1 and use the augmenting path algorithm to solve (26). Nearly the same codes used for Algorithm 1 can be used for Algorithm 2. The only difference is that we need to construct and solve the graph cuts problem over each subdomain instead of on the whole domain Ω.

For Algorithm 3, due to the overlapping of the subdomains, some extra care need to be given in solving subdomain problem (29). For a given $p \in \mathcal{P}_i^0$, $\mathcal{N}_k(p)$ may be outside Ω_i^0 and these values are known and needed for E_d^i in (28). If we take $k = 4$ or $k = 6$ for $\mathcal{N}_k$, then $\mathcal{N}_k(p)$ is always within $\mathcal{P}_i = \mathcal{P}_i^0 \cup \partial\mathcal{P}_i$ for any $p \in \mathcal{P}_i^0$. Each Ω_i contains many disjoint subdomains, i.e., $\Omega_i = \cap_j \Omega_{i,j}$. As the subproblems over $\Omega_{i,j}$ are independent of each other, we can use graph cuts algorithms to solve the subdomain problems simultaneously or one by one. For each subdomain problem, we construct the graph for the subdomain $\Omega_{i,j}$ to include the interior and boundary grid points, i.e., the subdomain graph is

$$\begin{aligned}
\mathcal{V}^{i,j} &= \{v_{p,l} \mid (p,l) \in \mathbb{R}^2 \times \mathbb{R} |\ p \in \mathcal{P}_{i,j}, l \in \{1,\ldots,n-1\}\},\\
\mathcal{E}^{i,j} &= \mathcal{E}_D^{i,j} \cup \mathcal{E}_R^{i,j},\\
\mathcal{E}_D^{i,j} &= \cup_{p\in\mathcal{P}_{i,j}} \{(s, v_{p,1}) \cup_{l=1}^{n-2} (v_{p,l}, v_{p,l+1}) \cup (v_{p,n-1}, t)\},\\
\mathcal{E}_R^{i,j} &= \{(v_{p,l}, v_{q,l}) | p \in \mathcal{P}_{i,j}^0, q \in \mathcal{N}_k(p), l \in 1,\ldots,n-1\}.
\end{aligned}$$

In the above, notations $\mathcal{P}_{i,j}$ and $\mathcal{P}_{i,j}^0$ are self explainable. The capacity of the edges for the interior grid points are defined as in (19)-(22). The boundary value of $\phi^{l+\frac{i}{4}}$ is known as $\phi^{l+\frac{i}{4}} = \phi^{l+\frac{i-1}{4}}$ in $\Omega\backslash\Omega_i^0$. We only need to compute the value of $\phi^{l+\frac{i}{4}}$ in the interior of Ω_i which can be computed in parallel over the subdomains $\Omega_{i,j}$. To keep the boundary values unchanged, the capacity of the edges in $\mathcal{E}_D^{i,j}$ for any $p \in \partial\mathcal{P}_{i,j}$ should be defined as ∞ except one that indicates the value of the point p and the capacity for this edge should be given as 0. Compared with the implementation of Algorithms 1-2, we only need to set the capacity of the "vertical edges" to be ∞ or 0 for the grid points on the boundary of Ω_i. This is the only extra "care" that we need to take for the implementation of Algorithm 3.

In our implementations, we take $tol = 0.1$, the phase number $n = 4$ and $\sigma = k(n-1)\gamma$. The neighborhood k adopted is either $k = 4$ or $k = 8$ for 2D examples and $k = 6$ or $k = 26$ for 3D examples. The value of γ varies with the examples. For Algorithm 3, we alway take $\phi^0 = 0$ and use ISODATA algorithm of [32] to get the initial values for $\mathbf{c}$. Besides, the size of overlapping is one pixel unless specified otherwise.

6. Numerical experiments

In the following, we implement our domain decomposition algorithms on 2D synthetic and real data and 3D real data respectively. We develop our codes in C++ using the augmenting path algorithm proposed in [3]. All 2D numerical experiments were performed on a HP xw4600 Workstation with an Intel(R) Core(TM) 2 Duo CPU E6750 @ 2.66 GHz, 2.67 GHz and 2.00 GB of RAM and 3D experiments were developed in LINUX environment.

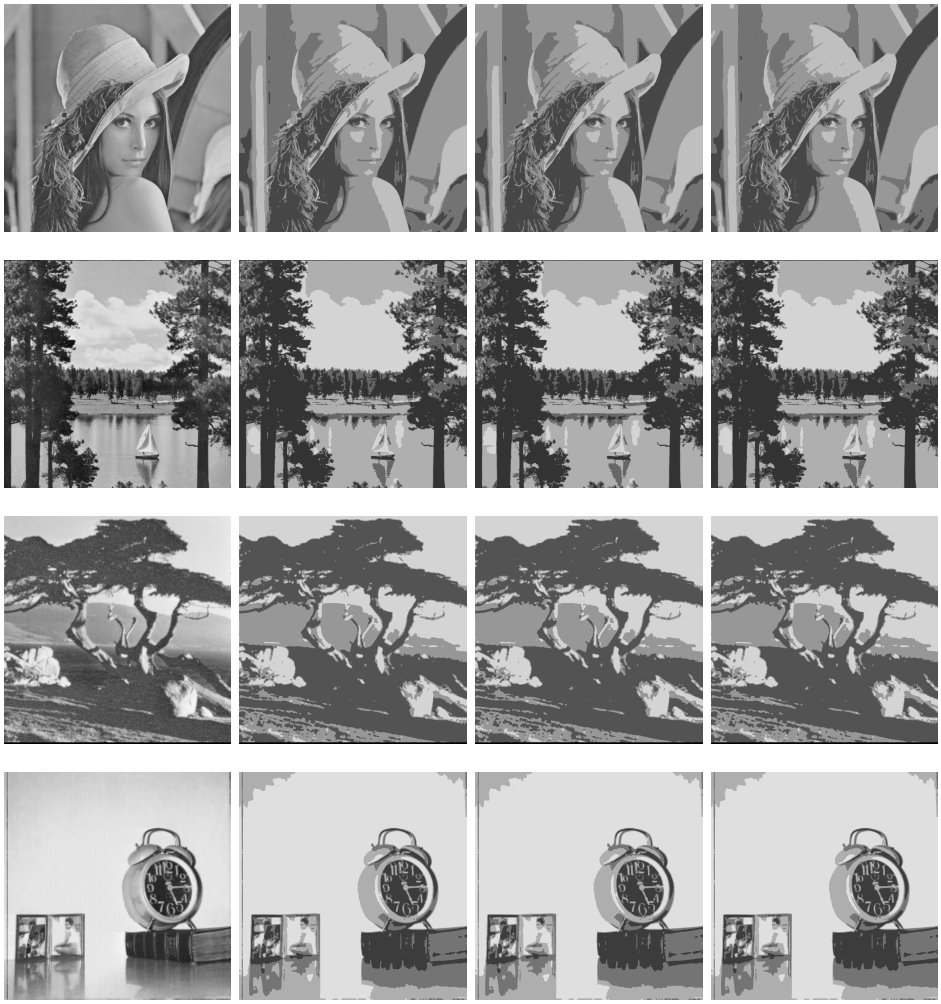
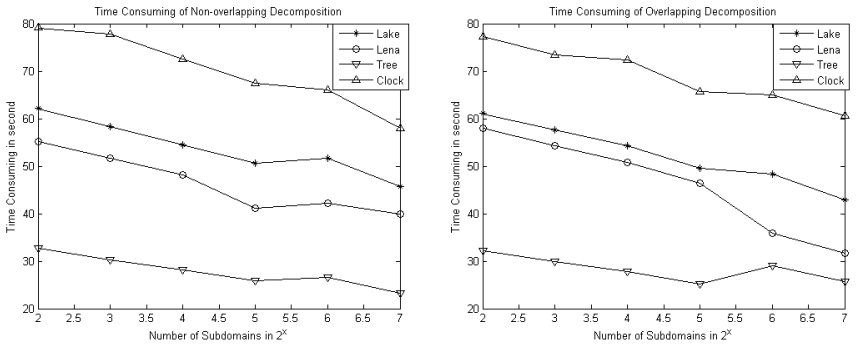


FIGURE 4. The comparison results of lena, lake, tree and clock with 16×16 subdomains. From left to right: given image, restored image by Algorithm 1, restored image by non-overlapping Algorithm 2 and restored image by overlapping Algorithm 3.

6.1. Tests of domain decomposition algorithm and 2D experiments. First of all, we set up a series of experiments to demonstrate the effectiveness of our domain decomposition (denoted by DD below) based algorithms: Algorithm 2 and Algorithm 3. These experiemnts are developed on four 1024×1024 images: Lena, Lake, Tree and Clock. We apply four phase DD algorithms together with Algorithm 1 to these four images. For all the algorithms, we choose the same neighborhood involved 4 connectivities and set $\gamma = 500$. For DD algorithms, the image domain is decomposed into 16×16 subdomains. The segmentation results of each algorithm are displayed in Fig.4 and the corresponding computation time is tabulated in Table 2.

In the following, we first test the sensitivity of the subdomain size to Algorithm 2 and Algorithm 3. We examine the computation time and approximation error (i.e., difference between full-domain solution and decomposed solution) of both DD



(a) (b)

FIGURE 5. CPU costs of nonoverlapping Algorithm 2 and overlapping Algorithm 3 with different subdomain sizes.

algorithms while the subdomain number is increased from 4 to 128 subdomains. Then we compare the memory consumption and minimal energy between DD algorithms and Algorithm 1 for these four images. For these tests, we fix $\gamma = 500$ for all algorithms.

- Sensitivity to the subdomain size

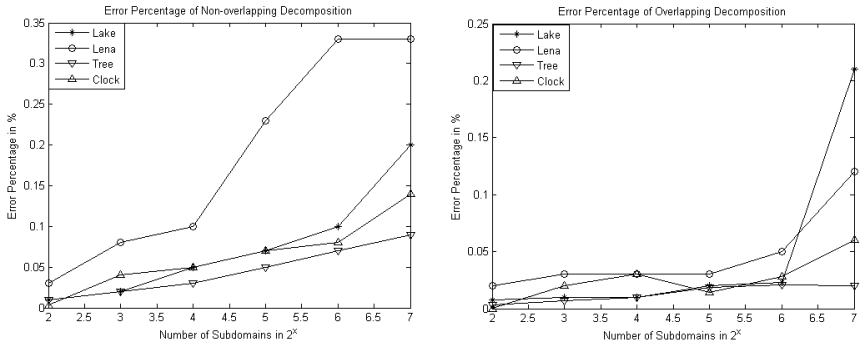
The subdomain size greatly affects the performance of Algorithm 2 and Algorithm 3. Therefore, we test the computational cost of DD based algorithms with regard to different subdomain sizes. For this purpose, we decompose the images into 2^2, 2^3, 2^4, 2^5, 2^6 and 2^7 subdomains respectively and implement DD algorithms to each case one by one. We plot the computation time of non-overlapping Algorithm 2 in Fig.5(a) and overlapping Algorithm 3 in Fig.5(b). Through the results, we see that DD algorithms can improve the computational efficiency compared to Algorithm 1.

Meanwhile, we evaluate an error estimation between DD based algorithms and Algorithm 1 with different decompositions. Continued the experiment, we let the images be decomposed into $2^2, 2^3, 2^4, 2^5, 2^6$ and 2^7 subdomains and use the same parameters. The error is computed by comparing the segmention results of Algorithm 2 or Algorithm 3 with the results of Algorithm 1 from the following formula

$$\epsilon = \frac{\sum_{p\in\mathcal{P}} \chi(\phi_p, \phi_p^0)}{M \times N}. \tag{31}$$

where ϕ_p^0 is from Algorithm 1 and ϕ_p is either from non-overlapping Algorithm 2 or from overlapping Algorithm 3. If $\phi_p = \phi_p^0$, we have $\chi(\phi_p, \phi_p^0) = 0$, otherwise we have $\chi(\phi_p, \phi_p^0) = 1$. We estimate the error between Algorithm 2 and Algorithm 1 and show the result in Fig.6(a) while we exhibit the similar result between Algorithm 3 and Algorithm 1 in Fig.6(b). From the results, it is clear that the error of non-overlapping algorithm is always somewhat larger than the corresponding error of overlapping algorithm. Since the error of overlapping Algorithm 3 keeps less than or around 0.2% when the images are decomposed from 4 to 128 subdomains, we can say that overlapping domain decomposition algorithm is quite a stable method for numerical application.

- Comparison of memory consumption



(a) (b)

FIGURE 6. Approximation error of nonoverlapping Algorithm 2 and overlapping Algorithm 3 with different subdomain sizes.

TABLE 1. The memory consumption of Lena, Lake, Tree and Clock. In the table, the unit of the number is kiloBytes.

Problem	Subdomain Number	Algorithm 1	Algorithm 2	Algorithm 3
Lena	16×16	587,262	28,560	31,960
Lake	16×16	551,202	28,560	31,404
Tree	16×16	541,124	28,560	31,888
Clock	16×16	552,272	29,040	31,904

Next, we analyze the memory consumption of DD algorithms and Algorithm 1. Like the beginning, the images are decomposed into 16×16 subdomains. We record the largest memory demanded by each algorithm of each image in the experiment and tabulate them in Table 1. Therefore, from the table we can see, for a 1024×1024 image, Algorithm 1 requires a memory around $600MB$ while DD algorithms only need less than 10% of this amount. It is proved that DD based algorithms can greatly decrease the memory requirements compared to Algorithm 1. Thus, nonoverlapping Algorithm 2 and overlapping Algorithm 3 can make segmentation problem (8) with large data size be solvable on computers with small memory.

- Comparison of minimal energy

Moreover, we try to illustrate that the minimal energy obtained by DD algorithms approximates the minimal energy of Algorithm 1, which is recognized as the global minimizer over the image domain. We continue to decompose the image domain into 16×16 subdomains as before. The energies of DD algorithms are calculated using the cut results over the entire domain. For Algorithm 2, we add all the weights of the cut edges in each subdomain while we add the weights of the internal cut edges in each subdomain for Algorithm 3. We display the comparision result of each image in Fig.7. In the figure, the energy of Algorithm 1 is marked by "$*$" and the energies of non-overlapping and overlapping decomposition are denoted by "$+$" and "o" respectively. Through the experiment result, we see that the energy of DD based algorithms is convergent and the difference between the energy of DD algorithms and the energy of Algorithm 1 is acceptable to us.

One more 2D experiment, we implement our DD algorithms to a real brain MR image of high resolution. We use four phase image segmentation approach to extract

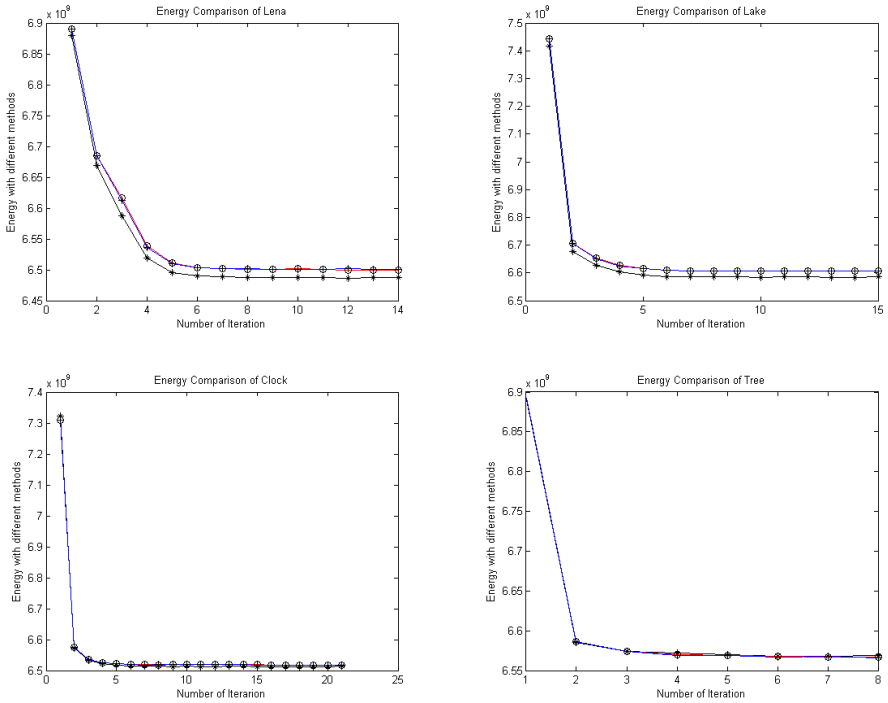


FIGURE 7. The energy comparision between Algorithm 1 (black curve), non-overlapping Algorithm 2 (blue curve) and overlapping Algorithm 3 (red curve).

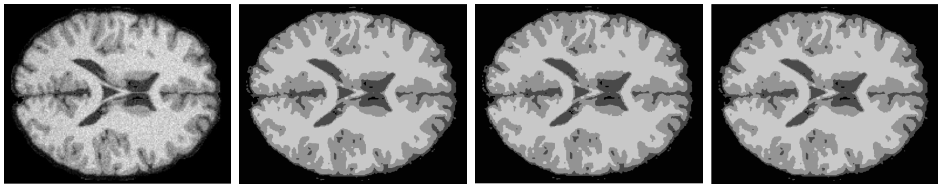


FIGURE 8. The comparision results of MR. From left to right: initial image, restored image by Algorithm 1, restored image by Algorithm 2 and restored image by Algorithm 3.

the 4 different classes of the brain image. We segment the MR image with TV^8 norm and display the results in the Fig.8 and the computation time in the Table 2. We can see that our decomposition methods can get almost the same result as by Algorithm 1 visually. In the meantime, the decomposition methods improve more than $\frac{1}{4}$ of the computation time.

6.2. 3D experiments. In this subsection, we test our nonoverlapping Algorithm 2 and overlapping Algorithm 3 to five 3D real data with name and size as follows: MRI($250 \times 250 \times 120$), Blow($512 \times 512 \times 512$), CGQ($512 \times 512 \times 512$), Hailuo($500 \times 500 \times 150$) and Huluo($512 \times 512 \times 512$).

For a start, we implement our multiphase graph cuts algorithms to a 3D MRI data. In this test, we apply four phase algorithms to the data and use $1000(10 \times 10 \times$

TABLE 2. Compution time in seconds for 2D experiments: Lena, Lake, Tree, Clock and MR.

Prob	Size	Neighbor	Subdomain No	Algorithm 1	Algorithm 2	Algorithm 3
Lena	1024×1024	4	16×16	51.641	38.797	31.688
Lake	1024×1024	4	16×16	57.45	44.657	42.875
Tree	1024×1024	4	16×16	29.859	22.594	25.86
Clock	1024×1024	4	16×16	74.203	56.438	63.344
MR	670×530	8	10×10	22.844	15.562	14.813

10) subdomains for both Algorithm 2 and Algorithm 3. For the neighborhood, we adopt both 6 and 26 connectivities for this 3D data respectively in the experiment. We display the computation time of each algorithm with two different connectivities in Table 5. Meantime, for different connectivities, we give the comparision results of a chosen slice of the data from each algorithm in Fig.9. From the detail of the results in Fig.9, it is easy to find out that overlapping Algorithm 3 approximates the result of Algorithm 1 better than nonoverlapping Algorithm 2. During the experiment, we also increase the subdomain number to be $20 \times 20 \times 20$ to see the effect of domain decomposition in improving the computation cost compared to Algorithm 1. The computation time is also shown in Table 5.

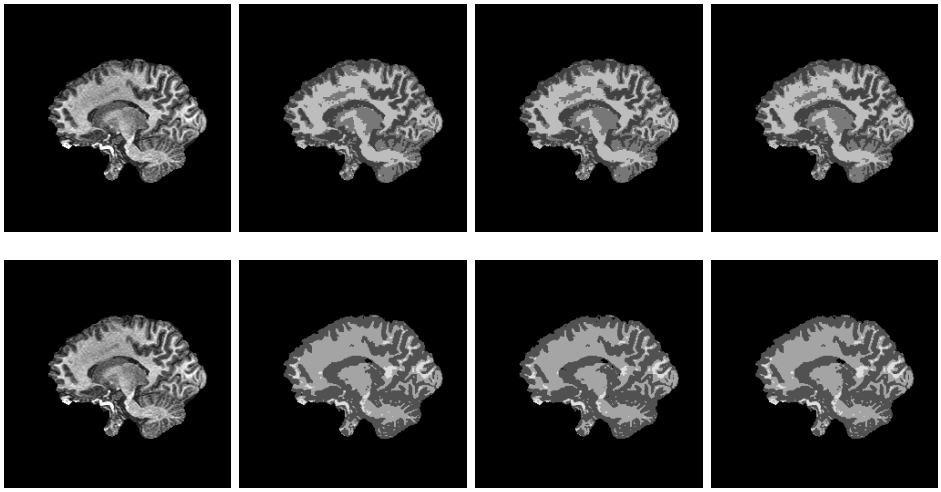


FIGURE 9. The comparison results of MRI. Row one are comparision results with 6 connectivity of slice nr.: 50. Row two are the comparision results with 26 connectivity of slice nr.: 50. Each row from left to right: inital image, restored image by Algorithm 1, restored image by non-overlapping Algorithm 2 and restored image by overlapping Algorithm 3.

Secondly, we apply six phase algorithms to segment the data CGQ. This CGQ is a real machine component with noise. Since this data is too large for Algorithm 1 to handle, we extract a $256 \times 256 \times 256$ data from initial data to compare the computation time between Algorithm 1 and DD algorithms. For DD Algorithm 2 and Algorithm 3, we use $32 \times 32 \times 32$ subdomains for both data of size either $512 \times 512 \times 512$ or $256 \times 256 \times 256$. We choose one slice from the results of Algorithm 1 and overlapping Algorithm 3 with data size $256 \times 256 \times 256$ and show these phase

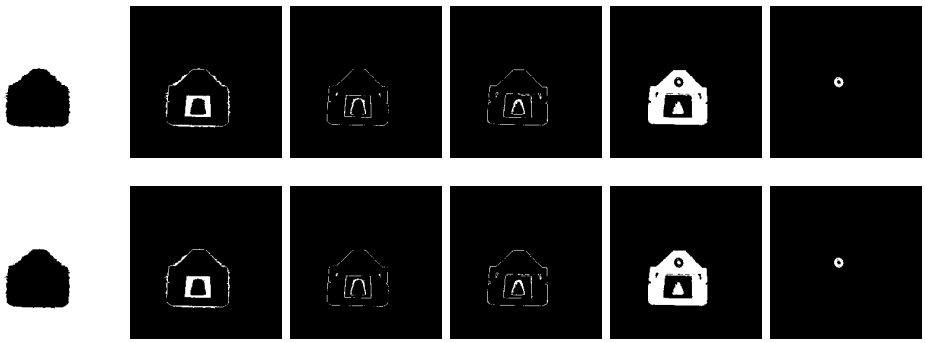


FIGURE 10. The CGQ results of slice nr.: 150. Row one from left to right is results of phase 1 to phase 6 of Algorithm 1. Row two from left to right is results of phase 1 to phase 6 of overlapping Algorithm 3.

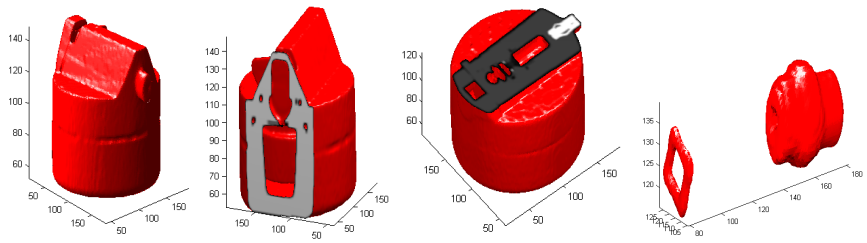


FIGURE 11. The surface results of 6 phase experiment: CGQ from overlapping Algorithm 3. All the results are from extracted data with proportion 1:2 to initial image.

results one by one in Fig.10. Meanwhile, we display some surface results from overlapping Algorithm 3 in Fig.11 and the computation time of each algorithm in Table 5. For the initial $512 \times 512 \times 512$ data, we fix the iteration being 11 for both Algorithm 2 and Algortihm 3 and record the corresponding computation time in Table 5.

For the data Blow, we implement three phase segmentation algorithms and adopt the neighborhood with 6 connectivity. Similarly, we extract a $256 \times 256 \times 256$ data from initial Blow data for the comparision of computation cost between Algorithm 1 and DD algorithms. For Algorithm 2 and Algorithm 3, we use $32 \times 32 \times 32$ subdomains for Blow data of size either $512 \times 512 \times 512$ or $256 \times 256 \times 256$. The numerical results of Blow are provided in Fig.12 and the computation time of different algorithms is shown in Table 5.

Besides, we develop a series of tests on Blow to show the advantages of our DD based algorithms. Firstly, we give a comparison of the computation time with different sizes of data extracted from Blow to illustrate the necessity of DD method. We tabulate the computation time from Algorithm 1 with different sizes of data in Fig.13(a) and use Fig.13(b) to manifest that DD can greatly reduce the computation cost when solve the segmentation problem. Then we compute the approximation error between DD algorithms and Algorithm 1 and list both computation time and approximation error of nonoverlapping Algorithm 2 and overlapping Algorithm 3 in Table 3. From the numbers in Table 3, it shows our DD algorithms require less

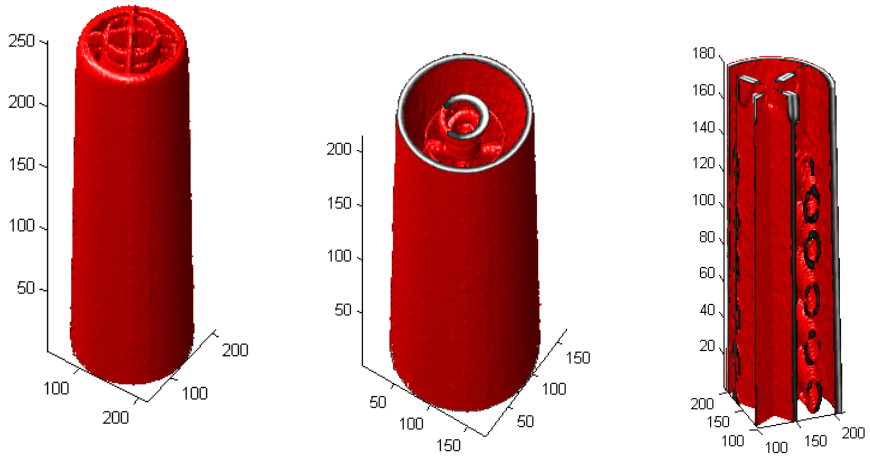
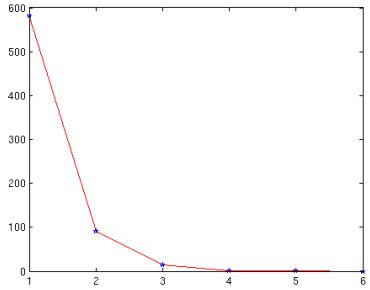


FIGURE 12. The surface results of 3 phase experiment: Blow from overlapping Algorithm 3. All the results are from extracted data with proportion 1:2 to initial image.

Image Size	CPU Time
256 x 256 x 256	581.21
128 x 128 x 128	90.93
64 x 64 x 64	13.54
32 x 32 x 32	1.21
16 x 16 x 16	0.07
8 x 8 x 8	0

(a)



(b)

FIGURE 13. Computation costs in seconds for different resolution. In (b), numbers on x-coordinate denote the image sizes in table (a).

time and keep a quite small error compared to Algorithm 1, especially overlapping Algorithm 3 which always keeps the error under 0.2%. Futhermore, we tabulate the largest physical memory used in the experiment regarding to these extracted data from Blow in Table 4. Corresponding to 2D experiment results, both DD algorithms save considerable quantity of memory and make large size data be solvable for us.

For the test of Hailuo and Huluo, we apply two phase segmentation algorithms and adopt the neighborhood involved 6 pixels. We use $20 \times 20 \times 20$ subdomains for Hailuo experiment and $32 \times 32 \times 32$ subdomains for Huluo experiment when we implement both non-overlapping Algorithm 2 and overlapping Algorithm 3. The numerical results of Algorithm 3 of these two data are supplied in Fig.14 and the computation time of each algorithm to both data is displayed in Table 5. These two phase experiments also demonstrate that domain decomposition can make large size data be segmented and save much computation costs in the meantime.

TABLE 3. CPU costs and error estimation of Algorithm 2 and Algorithm 3 for Blow experiment. The corresponding computation time of Algorithm 1 are listed in Fig.13(a).

Problem Size	γ	Subdomain Number	Algorithm 2		Algorithm 3	
			time	error	time	error
$256 \times 256 \times 256$	250	$32 \times 32 \times 32$	143.54	0.25%	202.03	0.053%
$128 \times 128 \times 128$	250	$16 \times 16 \times 16$	16.01	0.32%	22.84	0.16%
$64 \times 64 \times 64$	250	$8 \times 8 \times 8$	2.8	0.33%	5.7	0.039%
$32 \times 32 \times 32$	250	$4 \times 4 \times 4$	0.39	0.41%	0.77	0.042%
$16 \times 16 \times 16$	250	$2 \times 2 \times 2$	0.05	0.097%	0.06	0%
$8 \times 8 \times 8$	250	$2 \times 2 \times 2$	0	0.059%	0	0%

TABLE 4. Memory consumption of Algorithm 1, Algorithm 2 and Algorithm 3 for Blow experiment. The numbers are physical memory that a task has used (in kiloBytes) in experiment.

Problem Size	γ	Subdomain Number	Algorithm 1	Algorithm 2	Algorithm 3
$512 \times 512 \times 512$	500	$32 \times 32 \times 32$	– –	4,723,000	2,847,972
$256 \times 256 \times 256$	250	$32 \times 32 \times 32$	10,724,676	723,276	391,836
$128 \times 128 \times 128$	250	$16 \times 16 \times 16$	1,322,256	91,704	50,956
$64 \times 64 \times 64$	250	$8 \times 8 \times 8$	188,912	15,840	8,504
$32 \times 32 \times 32$	250	$4 \times 4 \times 4$	21,760	3,500	3,240
$16 \times 16 \times 16$	250	$2 \times 2 \times 2$	3,676	1,728	1,944
$8 \times 8 \times 8$	250	$2 \times 2 \times 2$	1,428	1,192	1,376

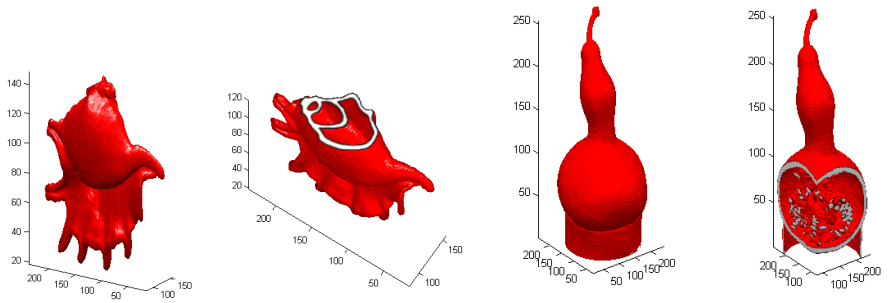


FIGURE 14. The surface results of 2 phase experiments: Hailuo and Huluo from overlapping Algorithm 3. All the results are from extracted data with proportion 1:2 to initial image.

TABLE 5. Compution time in seconds for 3D experiments. – denotes problem can not be handled by the method.

Prob	Size	γ	Neigh	Phase	Subdomain No	Alg 1	Alg 2	Alg 3
MRI	$250 \times 250 \times 120$	500	6	4	$10 \times 10 \times 10$	713.28	501.96	674.57
MRI	$250 \times 250 \times 120$	300	26	4	$10 \times 10 \times 10$	4897.16	2172.37	2629.56
MRI	$240 \times 240 \times 120$	500	6	4	$20 \times 20 \times 20$	879.07	313.08	317.25
CGQ	$256 \times 256 \times 256$	500000	6	6	$32 \times 32 \times 32$	6634.12	1793.87	1809.92
CGQ	$512 \times 512 \times 512$	300000	6	6	$32 \times 32 \times 32$	– –	10277.2	17401.5
Blow	$256 \times 256 \times 256$	250	6	3	$32 \times 32 \times 32$	621.59	143.54	202.03
Blow	$512 \times 512 \times 512$	500	6	3	$32 \times 32 \times 32$	– –	934.4	1280.57
Hailuo	$500 \times 500 \times 300$	500	6	2	$20 \times 20 \times 20$	619.01	102.94	219.04
Huluo	$256 \times 256 \times 256$	2000	6	2	$32 \times 32 \times 32$	245.09	49.53	98.04
Huluo	$512 \times 512 \times 512$	2000	6	2	$32 \times 32 \times 32$	– –	400.9	780.29

7. Conclusion

In this work, we propose a new method to minimize the Mumford-Shah model with piecewise constant level set representation. We apply the domain decomposition methods to image segmentation and use graph cuts algorithm to minimize the energy functionals. The proposed method improves the computation efficiency. Even more, it greatly reduces the memory cost and enables us to solve very large size images effectively. Due to the monotonicity property of the algorithms, its numerical performance is very robust. It is remarkable that the algorithm can segment 3D images with $4 \times 10^8 (512 \times 512 \times 512 \times 3)$ voxels in just a few minutes and the quality is comparable with traditional variational methods.

References

[1] E. Bae and X.C. Tai. Graph cuts for the multiphase Mumford-Shah model using piecewise constant level set methods. *UCLA, Applied Mathematics, CAM-report 08-36*, 2008.

[2] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *Ninth IEEE International Conference on Computer Vision, 2003. Proceedings*, pages 26–33, 2003.

[3] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[4] Y. Boykov, V. Kolmogorov, D. Cremers, and A. Delong. An integral solution to surface evolution PDEs via geo-cuts. *Lecture Notes in Computer Science*, pages 409–422, 2006.

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[6] T.F. Chan and T.P. Mathew. Domain decomposition algorithms. *Acta Numerica*, 3:61–143, 2008.

[7] T.F. Chan and L.A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.

[8] G. Chung and L.A. Vese. Energy minimization based segmentation and denoising using a multilayer level set approach. *Lecture Notes in Computer Science*, 3757:439–455, 2005.

[9] J. Darbon. A note on the discrete binary mumford-shah model. *Lecture Notes in Computer Science*, 4418:283–294, 2007.

[10] J. Darbon and M. Sigelle. Image restoration with discrete constrained total variation part I: Fast and exact optimization. *Journal of Mathematical Imaging and Vision*, 26(3):261–276, 2006.

[11] J. Darbon and M. Sigelle. Image restoration with discrete constrained Total Variation part II: Levelable functions, convex priors and non-convex cases. *Journal of Mathematical Imaging and Vision*, 26(3):277–291, 2006.

[12] A. Delong and Y. Boykov. A scalable graph-cut algorithm for nd grids. In *Proceedings of Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[13] A. Dervieux and F. Thomasset. A finite element method for the simulation of Rayleigh-Taylor instability. *Lecture Notes in Mathematics*, 771:145–159, 1979.

[14] N. El-Zehiry, S. Xu, P. Sahoo, and A. Elmaghraby. Graph cut optimization for the Mumford-Shah model. In *Proc. of the Int. conf. Visualization, Imaging, and Image Processing, Palma de Mallorca, Spain (August 2007)*, pages 182–187.

[15] D. Firsov and S. H. Lui. Domain decomposition methods in image denoising using gaussian curvature. *J. Comput. Appl. Math.*, 193(2):460–473, 2006.

[16] L. Ford and D. Fulkerson. Flows in networks. *Princeton University Press*, 1962.

[17] A.V. Goldberg and R.E. Tarjan. A new approach to the maximum-flow problem. *Journal of the ACM (JACM)*, 35(4):921–940, 1988.

[18] H. Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, 2003.

[19] H. Ishikawa and D. Geiger. Segmentation by grouping junctions. In *1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1998. Proceedings*, pages 125–131, 1998.

[20] Y.M. Jung, S.H. Kang, and J. Shen. Multiphase image segmentation via modica-mortola phase transition. *SIAM Journal on Applied Mathematics*, 67(5):1213–1232, 2007.

[21] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004.
[22] N. Komodakis, G. Tziritas, and N. Paragios. Fast, approximately optimal solutions for single and dynamic MRFs. In *Computer Vision and Pattern Recognition*, pages 1–8. Citeseer, 2007.
[23] C. Li, C. Xu, C. Gui, and M. Fox. Level set evolution without re-initialization: A new variational formulation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 430–436. Citeseer, 2005.
[24] J. Lie, M. Lysaker, and X.C. Tai. A binary level set model and some applications to Mumford-Shah image segmentation. *IEEE Transactions on Image Processing*, 15(5):1171–1181, 2006.
[25] J. Lie, M. Lysaker, and X.C. Tai. A variant of the level set method and applications to image segmentation. *Mathematics of Computation*, 75(255):1155–1174, 2006.
[26] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Comm. Pure Appl. Math*, 42(5):577–685, 1989.
[27] S. Osher and J.A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, pages 12–49, 1988.
[28] J.E. Solem, N.C. Overgaard, and A. Heyden. Initialization Techniques for Segmentation with the Chan-Vese Model. In *Proceedings of the 18th International Conference on Pattern Recognition-Volume 02*, pages 171–174. IEEE Computer Society, 2006.
[29] X.C. Tai, O. Christiansen, P. Lin, and I. Skjælaaen. Image segmentation using some piecewise constant level set methods with MBO type of projection. *International Journal of Computer Vision*, 73(1):61–76, 2007.
[30] X.C. Tai and M. Espedal. Rate of convergence of some space decomposition methods for linear and nonlinear problems. *SIAM Journal of Numerical Analysis*, pages 1558–1570, 1998.
[31] X.C. Tai and J. Xu. Global and uniform convergence of subspace correction methods for some convex optimization problems. *Mathematics of Computation*, 71(237):105–124, 2002.
[32] RD Velasco. Thresholding using the ISODATA clustering algorithm. *IEEE Transactions on Systems Man and Cybernetics*, 10(11):771–774, 1980.
[33] L.A. Vese and T.F. Chan. A multiphase level set framework for image segmentation using the Mumford and Shah model. *International Journal of Computer Vision*, 50(3):271–293, 2002.
[34] S. Vicente, V. Kolmogorov, and C. Rother. Graph cut based image segmentation with connectivity priors. In *Proceedings of Computer Vision and Pattern Recognition, June*, volume 8, 2008.

Division of Mathematical Science, School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore and Department of Mathematics, University of Bergen, Johannes Brunsgate 12, N-5008 Bergen, Norway.
E-mail: tai@mi.uib.no

Division of Mathematical Science, School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore.
E-mail: DUAN0010@ntu.edu.sg