Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 9468215, 17 pages
https://doi.org/10.1155/2020/9468215

Research Article

Constrained Linear Curvature Image Registration Model and Its Numerical Algorithm

Jin Zhang

School of Mathematical Sciences, Liaocheng University, Liaocheng, Shandong 252059, China

Correspondence should be addressed to Jin Zhang; zhangjinsunny321@163.com

Received 23 June 2020; Revised 13 August 2020; Accepted 20 August 2020; Published 13 October 2020

Academic Editor: Giovanni Lancioni

We propose a constrained linear curvature image registration model to explicitly control the deformation according to the transformed Jacobian matrix determinant using point-by-point inequality constraints in this paper. In addition, an effective numerical method is proposed to solve the resulting inequality constrained optimization model. Finally, some numerical examples are given to prove the obvious advantages of the curvature image registration model with inequality constraints.

1. Introduction

In image processing, people are interested not only in analyzing an image but also comparing or combining information from images which take different time, different places, different viewpoints, or different modalities. Thus, image registration is one of the most useful and challenging problems in the field of image processing. Its main idea is to find a geometric transformation which aligns points in one view of one object with corresponding points in another view of the same or similar object. At present, there are a large number of application areas which require image registration, such as computer vision, biological imaging, remote sensing, and medical imaging. For comprehensive surveys of these applications, refer to [1–5].

The basic framework of image registration can be described as follows: given two images of the same object, which are called reference image R and template image T, respectively, and our purpose is to find a vector value transformation φ as defined below:

$$\varphi(\mathbf{u})(\cdot)\colon \mathbb{R}^d \longrightarrow \mathbb{R}^d, \varphi(\mathbf{u})(\mathbf{x})\colon \mathbf{x} \longrightarrow \mathbf{x} + \mathbf{u}(\mathbf{x}), \tag{1}$$

or equivalently find the unknown displacement field $\mathbf{u}$:

$$\mathbf{u}\colon \mathbb{R}^d \longrightarrow \mathbb{R}^d, \mathbf{u}\colon \mathbf{x} \longrightarrow \mathbf{u}(\mathbf{x}) = \left(u_1(\mathbf{x}), u_2(\mathbf{x}), \ldots, u_d(\mathbf{x})\right)^\top, \tag{2}$$

so that the transformed template image $T(\varphi(\mathbf{x})) = T(\mathbf{x} + \mathbf{u}(\mathbf{x})) \triangleq T(\mathbf{u})$ is as similar to the reference image R as possible. Here, $d \in \mathbb{N}$ denotes spatial dimension of the given images.

Without loss of generality, here we focus on $d = 2$ throughout this paper, but it is easy to generalize to $d = 3$ with some additional modifications. The variational model is an important tool for studying image registration and has been widely concerned by many researchers [5–9]. This variational model treats the image registration problem as a minimization problem of the joint energy functional in the following form:

$$\min_{\mathbf{u}}\{\mathcal{J}_\alpha[\mathbf{u}] = \mathcal{D}(\mathbf{u}) + \alpha\mathcal{S}(\mathbf{u})\}, \tag{3}$$

where

$$\mathcal{D}(\mathbf{u}) = \frac{1}{2}\int_\Omega (T(\mathbf{u}) - R)^2 \mathrm{d}_\Omega, \tag{4}$$

where $\mathcal{D}(\mathbf{u})$ denotes distance measure which quantifies distance or similarity of the transformed template image $T(\mathbf{u})$ and reference R, for other choices on $\mathcal{D}(\mathbf{u})$, refer to [5, 7], $\mathcal{S}(\mathbf{u})$ is the deformation regularizer which constrains $\mathbf{u}$ and ensures the well-posedness of the problem, and $\alpha > 0$ is a regularization parameter which balances similarity and regularity of displacement.

We all know that different regularizers will produce displacement fields with different degrees of smoothness and the selection of a regularizer is critical to the solution of the problem and its properties; for more details, refer to [5]. Usually the choice of regularizer can be classified into two main categories: the first type is to limit the displacement field $\mathbf{u}(\mathbf{x})$ to the parametric model [10–14], for example, rigid or affine transformations (parameterized by rotation, scaling, and translation) or linear combinations of a set of basis functions (B-splines) [3, 5, 15–18]; the second type is based on the derivative of the displacement field. At present, there are regularizers based on first-order derivatives, such as elastic regularizer [19–21], diffusion regularizer [22], total variational regularizer [23, 24], modified total variational regularizer [25, 26], total fractional-order regularizer [27], and the ones based on higher-order derivatives, such as linear curvature [28, 29], mean curvature [8, 30], and Gaussian curvature [31]. It has been proved that, in many cases, the selection method of the first kind of a regularizer is too strict, and the required transformation cannot be guaranteed to be included in the parametric model. Therefore, the second kind of method is a common method to select a regularizer. For the second method, it is easy to implement for low-order regularizers, while they are less effective than high-order ones in producing smooth displacement fields which are important in some applications including medical imaging. Although the registration models based on a higher-order regularizer can produce more satisfactory registration results visually, they do not take into account mesh folding.

In fact, the regularity of the displacement field is also an important measure in image registration [32]. In many variational models (1) that currently exist, although they can produce satisfactory registration results visually, they cannot ensure that the transformation $\varphi(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\mathbf{x})$ found is reversible. The irreversibility of the transformation means that the displacement field is not regular. In this case, there will be mesh folding during the registration process, which is not allowed in practical applications. Therefore, it is necessary to avoid mesh folding during the registration process. Currently, a direct idea to avoid mesh folding is to use a larger regularization parameter α. However, such a value will cause the similarity between the transformed template image and the reference image to become worse. In order to avoid mesh folding phenomenon, some scholars have proposed to add an additional regular term $C(\mathbf{u})$ on the transformed Jacobian matrix determinant in the objective function formula (3) [33–36], i.e.,

$$\min_{\mathbf{u}}\left\{\mathcal{J}_{\alpha}[\mathbf{u}] = \mathcal{D}(\mathbf{u}) + \alpha\mathcal{S}(\mathbf{u}) + \beta\|C(\mathbf{u}) - 1\|^2\right\}, \tag{5}$$

where $C(\mathbf{u}) = \det(I_d + \nabla\mathbf{u})$ represents the determinant of the Jacobian matrix of the transformation. However, this method only penalizes the irregular displacement field as a whole, while the local displacement field cannot be guaranteed to be regular [32]. In addition, this method is only effective for the smaller regularizer parameter β, and increasing the value of β usually leads to ill-posed optimization problems [37]. To solve this problem, Haber and Modersitzki proposed a new registration model by adding additional explicit volume inequality constraints [32]; however, this constrained method usually leads to solving a large-scale highly nonlinear inequality constrained optimization problem. Other methods to ensure the regularity of the displacement field can be found in the literature [20, 38–43]. However, some of them require more computation time due to the complexity of the regularizer.

There are two purposes for image registration. One is to enhance some similarities between two images by geometrically transforming one of the given two images. The other is to ensure that this transformation is reasonable. In fact, it is equivalent to find geometric transformation φ and the displacement field $\mathbf{u}(\mathbf{x})$ in the framework of variational model. If the displacement field is irregular, the transformation is considered unreasonable, and then the mesh folding phenomenon will appear which is not allowed in practical applications. In this paper, we propose a new image registration model by integrating the evaluation criteria to measure the registration results directly into the basic framework of the variational model (3).

The rest of the paper is organized as follows: in Section 2, we propose a new constrained linear curvature image registration model. Then in Section 3, we discuss the numerical method for solving the new model by using a combination of the multiplier method and Gauss–Newton scheme with the Armijos line search and further combine with a multilevel method to achieve fast convergence. Next, some experimental results from syntectic and real images are illustrated in Section 4. Finally, conclusions and future work are summarized in Section 5.

2. Constrained Linear Curvature Image Registration Model

Firstly, we briefly review the Fischer–Modersitzki's linear curvature image registration model [25, 27]. Choosing $\mathcal{S}(\mathbf{u})$ in (3) based on an approximation to the curvature of the surface of the displacement field u_l is given by the following form:

$$\mathcal{S}^{\mathrm{LC}}(\mathbf{u}) = \frac{1}{2}\sum_{l=1}^{2}\int_{\Omega}(\Delta u_l)^2 \mathrm{d}_{\Omega}. \tag{6}$$

There are two major advantages to the particular choice of the regularizer. Firstly, it can penalize oscillations; secondly, without requiring an additional affine linear preregistration step, it can produce visually more satisfactory registration results than a diffusion model and an elastic model for smooth displacement fields. However, a mesh folding phenomenon is not considered in this linear curvature model. In order to avoid this, the evaluation criteria to measure the registration results are directly integrated into the basic framework of the variational model (3), and we propose a constrained linear curvature image registration model in the following form:

$$\min_{\mathbf{u}}\left\{\mathcal{J}_{\alpha}(\mathbf{u}) = \frac{1}{2}\int_{\Omega}(T(\mathbf{u}) - R)^2\mathrm{d}_{\Omega} + \frac{\alpha}{2}\sum_{l=1}^{2}\int_{\Omega}(\Delta u_l)^2\mathrm{d}_{\Omega}\right\}$$

$$\text{s.t.}\quad \mathcal{C}(\mathbf{u}) > 0, \tag{7}$$

where

$$\begin{aligned}\mathscr{C}(\mathbf{u}) &= \det\left(J(\varphi(\mathbf{x}))\right)\\ &= \begin{vmatrix} 1+(u_1)_x & (u_1)_y \\ (u_2)_x & 1+(u_2)_y \end{vmatrix}\\ &= \left(1+(u_1)_x\right)\left(1+(u_2)_y\right)-(u_1)_y(u_2)_x.\end{aligned} \tag{8}$$

Compared with model (5), our new model can ensure that the displacement field is regular both globally. In addition, the new model prevents mesh folding even for very small regularization parameters α. Finally, visually pleasing registration results can be obtained by using our new model with low computing time for smooth registration problems. The numerical solution of the new model (7) is given below.

3. Numerical Solution of the New Model

In general, it is difficult to solve the optimization problem (7) by the analytic method. Thus it is necessary to adopt the numerical method and appropriate discretization. In this paper, we choose the discretize-optimize method which aims to take advantage of efficient optimization techniques. In this section, we first discuss briefly the discretization we use and then describe the details of numerical algorithms.

3.1. Finite Difference Discretization. Assume that given discrete images have $m_1 \times m_2$ pixels. For simplicity, the image region is further assumed to be $\Omega = [0,1] \times [0,1] \subset \mathbb{R}^2$, and then each side of these $m_1 \times m_2$ cell-centered images has width $h_i = (1/m_i), i = 1, 2$. Thus the discrete domain can be denoted by

$$\Omega_h = \left\{\mathbf{x} \in \Omega \mid \mathbf{x} = \left(x_i, y_j\right)^\top = \left((i-0.5)h_1, (j-0.5)h_2\right)^\top, \quad i = 1, 2, \ldots, n_1; j = 1, 2, \ldots, n_2\right\}. \tag{9}$$

3.1.1. Discretization of Regularizer. The discrete form of the continuous displacement field $\mathbf{u} = (u_1, u_2)^\top$ can be represented by $\mathbf{u}^h = (u_1^h, u_2^h)$, where u_1^h and u_2^h are the discrete grid functions defined on the discrete region Ω_h. For convenience, let $(u_l^h)_{ij} = u_l^h(x_i, y_j)$, $i = 1, 2, \ldots, m_1$, $j = 1, 2, \ldots, m_2$, and $l = 1, 2$. Since the curvature regularizer is expressed based on the Laplacian operator Δ which can be regarded as the product of gradient operator ∇ and divergence operator $\nabla\cdot$, we introduce the symbols ∇^h and $\nabla^h\cdot$ to represent their discrete forms, respectively. The discrete gradient operator ∇^h can be defined at each pixel (i, j) by the following form:

$$\left(\nabla^h \mathbf{u}^h\right)_{i,j} = \left(\left(\nabla^h u_1^h\right)_{i,j}, \left(\nabla^h u_2^h\right)_{i,j}\right)^\top, \tag{10}$$

where

$$\left(\nabla^h u_l^h\right)_{i,j} = \left(\left(\partial_x^h u_l^h\right)_{i,j}, \left(\partial_y^h u_l^h\right)_{i,j}\right)^\top,$$

$$\left(\partial_x^h u_l^h\right)_{i,j} = \begin{cases} \dfrac{1}{h_1}\left(\left(u_l^h\right)_{i+1,j} - \left(u_l^h\right)_{i,j}\right), & \text{if } i < m_1;\\ 0, & \text{if } i = m_1; \end{cases}$$

$$\left(\partial_y^h u_l^h\right)_{i,j} = \begin{cases} \dfrac{1}{h_2}\left(\left(u_l^h\right)_{i,j+1} - \left(u_l^h\right)_{i,j}\right), & \text{if } j < m_2;\\ 0, & \text{if } j = m_2. \end{cases} \tag{11}$$

The displacement field $\mathbf{u}$ satisfies the homogeneous Neumann boundary conditions on the boundary $\partial\Omega$ of the image region Ω:

$$\frac{\partial u_l}{\partial \mathbf{v}} = 0, \quad l = 1, 2. \tag{12}$$

Through the analysis of continuous setting, we know that the discrete divergence operator is the negative conjugate transposition of the gradient operator, namely, $\nabla\cdot = -\nabla^*$. Thus, it can be defined by the following form:

$$\begin{aligned}(\nabla\cdot\boldsymbol{\omega})_{i,j} = &\begin{cases} \dfrac{1}{h_1}\left((\omega_1)_{i,j} - (\omega_1)_{i-1,j}\right) & \\ \dfrac{1}{h_1}\left((\omega_1)_{i,j}\right) & \\ -\dfrac{1}{h_1}(\omega_1)_{i-1,j} & \end{cases}\\ &+ \begin{cases} \dfrac{1}{h_2}\left((\omega_2)_{i,j} - (\omega_2)_{i,j-1}\right), & \text{if } 1 < i < n_1,\ 1 < j < n_2;\\ \dfrac{1}{h_2}\left((\omega_2)_{i,j}\right), & \text{if } i = j = 1;\\ -\dfrac{1}{h_2}(\omega_2)_{i,j-1}, & \text{if } i = n_1,\ j = n_2, \end{cases}\end{aligned} \tag{13}$$

where $\omega = (\omega_1, \omega_2)$ is a vector. For the convenience of calculation, the grid functions u_1^h and u_2^h can be changed into column vectors $\mathbf{u}_1^h$ and $\mathbf{u}_2^h$ according to lexicographical ordering, respectively:

$$\begin{aligned}u_1^h &= \left(u_{1,1,1}^h, \ldots, u_{1,n_1,1}^h, u_{1,1,2}^h, \ldots, u_{1,n_1,2}^h, \ldots, u_{1,n_1,n_2}^h\right)^\top,\\ u_2^h &= \left(u_{2,1,1}^h, \ldots, u_{2,n_1,1}^h, u_{2,1,2}^h, \ldots, u_{2,n_1,2}^h, \ldots, u_{2,n_1,n_2}^h\right)^\top.\end{aligned} \tag{14}$$

We can get $\mathbf{u}_1^h \in \mathbb{R}^N$, $\mathbf{u}_2^h \in \mathbb{R}^N$, and $\mathbf{U}^h = (\mathbf{u}_1^h, \mathbf{u}_2^h)^\top \in \mathbb{R}^{2N}$, where $N = n_1 n_2$. Discrete gradient operator $(\nabla^h u_l^h)_{i,j}$ can also be expressed as the product of matrix $A_k^\top \in \mathbb{R}^{2\times N}$ $(k = 1, 2, \ldots, N)$ and the vector $\mathbf{u}_l^h$ $(l = 1, 2)$ in the following form:

$$A_k^\top \mathbf{u}_l^h = \begin{cases} \left(\left(\mathbf{u}_l^h\right)_{k+1} - \left(\mathbf{u}_l^h\right)_k; \left(\mathbf{u}_l^h\right)_{k+n_2} - \left(\mathbf{u}_l^h\right)_k\right), & \text{if } k \bmod n_1 \neq 0 \text{ and } k + n_2 \leq N; \\ \left(0; \left(\mathbf{u}_l^h\right)_{k+n_2} - \left(\mathbf{u}_l^h\right)_k\right), & \text{if } k \bmod n_1 = 0 \text{ and } k + n_2 \leq N; \\ \left(\left(\mathbf{u}_l^h\right)_{k+1} - \left(\mathbf{u}_l^h\right)_k; 0\right), & \text{if } k \bmod n_1 \neq 0 \text{ and } k + n_2 > N; \\ (0; 0), & \text{if } k \bmod n_1 = 0 \text{ and } k + n_2 > N. \end{cases} \tag{15}$$

Let

$$\begin{aligned} A &= (A_1, A_2, \ldots, A_N) = \left(A_{1,1}, A_{1,2}, \ldots, A_{N,1}, A_{N,2}\right) \in \mathbb{R}^{N\times 2N}; \\ A_x &= \left(A_{1,1}, A_{2,1}, \ldots, A_{N,1}\right) \in \mathbb{R}^{N\times N}; \\ A_y &= \left(A_{1,2}, A_{2,2}, \ldots, A_{N,2}\right) \in \mathbb{R}^{N\times N}. \end{aligned} \tag{16}$$

By this notation, we can get

$$\begin{aligned} \nabla^h \mathbf{u}_1^h &= \begin{bmatrix} A_x^\top \\ A_y^\top \end{bmatrix} \mathbf{u}_1^h \triangleq B\mathbf{u}_1^h, \\ \nabla^h \mathbf{u}_2^h &= \begin{bmatrix} A_x^\top \\ A_y^\top \end{bmatrix} \mathbf{u}_2^h \triangleq B\mathbf{u}_2^h. \end{aligned} \tag{17}$$

Let $C = B^\top B$, $\mathbb{C} = \begin{bmatrix} C & 0 \\ 0 & C \end{bmatrix}$, and

$$\mathscr{B}[\mathbf{u}] = \sum_{l=1}^{2} (\Delta u_l)^2 = (\Delta u_1)^2 + (\Delta u_2)^2. \tag{18}$$

Then the discrete form of (18) is as follows:

$$\begin{aligned} \mathbb{B}^h\left[\mathbf{U}^h\right] &= \left(-B^\top B\mathbf{u}_1^h\right)^\top \left(-B^\top B\mathbf{u}_1^h\right) + \left(-B^\top B\mathbf{u}_2^h\right)^\top \left(-B^\top B\mathbf{u}_2^h\right) \\ &= \left(-C\mathbf{u}_1^h\right)^\top \left(-C\mathbf{u}_1^h\right) + \left(-C\mathbf{u}_2^h\right)^\top \left(-C\mathbf{u}_2^h\right) = \left(\mathbf{u}_1^h\right)^\top \\ &\quad \cdot (C^\top C)\mathbf{u}_1^h + \left(\mathbf{u}_2^h\right)^\top (C^\top C)\mathbf{u}_2^h \\ &= \left(\left(\mathbf{u}_1^h\right)^\top, \left(\mathbf{u}_2^h\right)^\top\right) \begin{bmatrix} C^\top C & 0 \\ 0 & C^\top C \end{bmatrix} \begin{bmatrix} \mathbf{u}_1^h \\ \mathbf{u}_2^h \end{bmatrix} \\ &= \left(\left(\mathbf{u}_1^h\right)^\top, \left(\mathbf{u}_2^h\right)^\top\right) \begin{bmatrix} C & 0 \\ 0 & C \end{bmatrix}^T \begin{bmatrix} C & 0 \\ 0 & C \end{bmatrix} \begin{bmatrix} \mathbf{u}_1^h \\ \mathbf{u}_2^h \end{bmatrix} \\ &= \left(\mathbf{U}^h\right)^\top \mathbb{C}^\top \mathbb{C}\mathbf{U}^h. \end{aligned} \tag{19}$$

According to the midpoint quadrature formula, the linear curvature regularizer $\mathscr{S}^{\mathrm{LC}}(\mathbf{u}) = (1/2)\int_\Omega \mathscr{B}[\mathbf{u}]d_\Omega$ has the following discrete form:

$$\mathscr{S}^h\left(\mathbf{U}^h\right) = \frac{1}{2} h_d \left(\mathbf{U}^h\right)^\top \mathbb{C}^\top \mathbb{C}\mathbf{U}^h, \tag{20}$$

where $h_d = h_1 h_2$.

3.1.2. Discretization of Template T and Reference R. For a given discrete image, if we want to know the gray value at any spatial location other than the grid point, then image interpolation is needed. In order to take full advantage of the fast and effective optimization method, a smooth cubic B-spline is used for interpolation. Next, $\mathscr{T}$ and $\mathscr{R}$ are used to represent the continuous smooth approximation of template image T and reference image R, respectively. Let

$$\begin{aligned} \mathbf{x}_c &= \left[x_{1,1}, \ldots, x_{n_1,1}, x_{1,2}, \ldots, x_{n_1,2}, \ldots, x_{1,n_2}, \ldots, x_{n_1,n_2}\right]^\top, \\ \mathbf{y}_c &= \left[y_{1,1}, \ldots, y_{n_1,1}, y_{1,2}, \ldots, y_{n_1,2}, \ldots, y_{1,n_2}, \ldots, y_{n_1,n_2}\right]^\top, \end{aligned} \tag{21}$$

and $\mathbf{X}_c^h = [\mathbf{x}_c; \mathbf{y}_c]$.

Thus the discrete reference image and transformed template image can be represented by the following form, respectively:

$$\overrightarrow{R} = \mathscr{R}\left(\mathbf{X}_c^h\right), \tag{22}$$

$$\overrightarrow{T}\left(\mathbf{U}^h\right) = \mathscr{T}\left(\mathbf{X}_c^h + \mathbf{U}^h\right), \tag{23}$$

and further we can get the Jacobian of $\overrightarrow{T}$:

$$\overrightarrow{T}_{\mathbf{U}^h} = \frac{\partial \overrightarrow{T}}{\partial U^h}\left(U^h\right) = \frac{\partial \mathscr{T}}{\partial \mathbf{U}_c^h}\left(\mathbf{U}_c^h\right), \tag{24}$$

where $\mathbf{U}_c^h = \mathbf{X}_c^h + \mathbf{U}^h$ and the Jacobian of $\overrightarrow{T}$ is a block matrix with diagonal blocks.

3.1.3. Discretization of Distance Measure $\mathscr{D}$. Although it is in a continuous setting, it is not possible to compute integrals analytically. Thus it is necessary to use numerical integration. In discrete simulation, the midpoint quadrature formula can be used to approximate the integral. According to (22) and (23), the discrete form of distance measurement $\mathscr{D}(2)$ can be written directly as follows:

$$\mathscr{D}^h\left(\mathbf{U}^h\right) = \frac{1}{2} h_1 h_2 \left(\overrightarrow{T}\left(\mathbf{U}^h\right) - \overrightarrow{R}\right)^\top \left(\overrightarrow{T}\left(\mathbf{U}^h\right) - \overrightarrow{R}\right). \tag{25}$$

In addition, the derivative of the discrete functional $\mathscr{D}^h(\mathbf{U}^h)$ on $\mathbf{U}^h$ can also be calculated and has the following form:

$$d\mathscr{D}^h\left(\mathbf{U}^h\right) = h_1h_2\left(\overrightarrow{T}_{\mathbf{U}^h}\right)^\top\left(\overrightarrow{T}\left(\mathbf{U}^h\right) - \overrightarrow{R}\right). \tag{26}$$

Furthermore, we can calculate the second derivative $d^2\mathscr{D}^h(\mathbf{U}^h)$ of the distance measurement $\mathscr{D}^h(\mathbf{U}^h)$:

$$d^2\mathscr{D}^h\left(\mathbf{U}^h\right) = h_1h_2\left(\overrightarrow{T}_{\mathbf{U}^h}\right)^\top\overrightarrow{T}_{\mathbf{U}^h} + h_1h_2\sum_{i=1}^{N} d_i\left(\mathbf{U}^h\right)\nabla^2 d_i\left(\mathbf{U}^h\right), \tag{27}$$

where $d(\mathbf{U}^h) = \overrightarrow{T}(\mathbf{U}^h) - \overrightarrow{R} \in \mathbb{R}^N$. On one hand, it is consuming and numerically unstable to compute higher-order derivatives (27) in registering two images for practical applications. On the other hand, the difference between $\overrightarrow{T}(\mathbf{U}^h)$ and $\overrightarrow{R}$ will become smaller if the template image is well registered. To have an efficient and stable numerical algorithm as proposed in work [5], $d^2\mathscr{D}^h(\mathbf{U}^h)$ can be approximated by the following form:

$$d^2\mathscr{D}^h\left(\mathbf{U}^h\right) = h_1h_2\left(\overrightarrow{T}_{\mathbf{U}^h}\right)^\top\overrightarrow{T}_{\mathbf{U}^h}. \tag{28}$$

3.1.4. Discretization of Inequality Constraint Functional $\mathscr{C}(\mathbf{u})$. In model (7), the inequality constraint functional $\mathscr{C}(\mathbf{u})$ is defined by

$$\mathscr{C}(\mathbf{u}) = \left(1 + (u_1)_x\right)\left(1 + (u_2)_y\right) - (u_1)_y(u_2)_x. \tag{29}$$

According to the previous analysis, the discrete form of the partial derivative of the continuous displacement field element u_l can be expressed as follows:

$$\begin{aligned}\left(\mathbf{u}_l^h\right)_x &= A_x^\top\mathbf{u}_l^h \triangleq m_l;\\ \left(\mathbf{u}_l^h\right)_y &= A_y^\top\mathbf{u}_l^h \triangleq w_l,\\ l &= 1, 2.\end{aligned} \tag{30}$$

Obviously, $m_l \in \mathbb{R}^N$, and $w_l \in \mathbb{R}^N$, where $N = n_1 \times n_2$. Let

$$\begin{aligned}e &= (1, 1, \ldots, 1)^\top \in \mathbb{R}^N,\\ c &= (e + m_1)\circledast(e + w_2) - w_1\circledast m_2,\end{aligned} \tag{31}$$

where symbol $\circledast$ denotes the multiplication of the corresponding elements of two vectors and $c \in \mathbb{R}^N$. For the convenience of calculation, let c_i denote the i-th element of c, $i = 1, 2, \ldots, N$. Therefore, the continuous inequality constraint function $\mathscr{C}(\mathbf{u})$ has the following discrete form:

$$C^h\left(\mathbf{U}^h\right) = (c_1, c_2, \ldots, c_N)^\top. \tag{32}$$

Since the first-order variation of the continuous inequality constraint function $\mathscr{C}(\mathbf{u})$ on continuous displacement field $\mathbf{u}$ is as follows:

$$d\mathscr{C}(\mathbf{u}) = \left((u_2)_{xy} - (u_2)_{yx}, (u_1)_{yx} - (u_1)_{xy}\right)^\top. \tag{33}$$

Thus we can get the discrete form of the first-order variation $d\mathscr{C}(\mathbf{u})$:

$$dC^h\left(\mathbf{U}^h\right) = \begin{bmatrix} 0 & A_y^\top A_x^\top - A_x^\top A_y^\top \\ A_x^\top A_y^\top - A_y^\top A_x^\top & 0 \end{bmatrix}\begin{bmatrix}\mathbf{u}_1^h\\ \mathbf{u}_2^h\end{bmatrix} \triangleq \mathbb{A}\mathbf{U}^h. \tag{34}$$

Obviously, $dC^h(\mathbf{U}^h) \in \mathbb{R}^{2N}$, $0 \in \mathbb{R}^{N\times N}$, and $\mathbb{A} \in \mathbb{R}^{2N\times 2N}$.

3.2. Solving the Discrete Optimization Problem. According to the above analysis, inequality constrained functional (7) has the following discrete form:

$$\begin{aligned}&\min_{\mathbf{U}^h}\left\{\mathscr{J}_\alpha\left(\mathbf{U}^h\right) = \mathscr{D}^h\left(\mathbf{U}^h\right) + \alpha\mathcal{S}^h\left(\mathbf{U}^h\right)\right\}\\ &\text{s.t.}\quad C^h\left(\mathbf{U}^h\right) > 0.\end{aligned} \tag{35}$$

Below we use the multiplier method to numerically solve the inequality constrained optimization problem (35). The basic idea of this method is to transform the original problem into a series of unconstrained optimization problems to solve and simultaneously estimate the Lagrangian multiplier. For more details on multiplier scheme, see [37]. Before solving (35), let us briefly review the multiplier method of inequality constrained optimization.

3.2.1. Multiplier Method for Inequality Constrained Problems. Consider the following inequality constrained optimization problem:

$$\begin{aligned}&\min\quad f(x),\\ &\text{s.t.}\quad g_i(x) \geq 0,\quad i = 1, 2, \ldots, m.\end{aligned} \tag{36}$$

Let $y_i \geq 0$, and the above inequality constraint can be transformed into the following equivalent equality constraint problem:

$$\begin{aligned}&\min\quad f(x),\\ &\text{s.t.}\quad g_i(x) - y_i^2 = 0,\quad i = 1, 2, \ldots, m.\end{aligned} \tag{37}$$

In this case, the augmented Lagrange function can be expressed as

$$\widetilde{\varphi}(x, y, \lambda, \sigma) = f(x) - \sum_{i=1}^{m}\lambda_i\left[g_i(x) - y_i^2\right] + \frac{\sigma}{2}\sum_{i=1}^{m}\left[g_i(x) - y_i^2\right]^2. \tag{38}$$

In order to eliminate the auxiliary variables y, the minimization of $\widetilde{\varphi}$ with respect to variable y can be considered. According to the first-order necessary condition, let

$$\nabla_y\widetilde{\varphi}(x, y, \lambda, \sigma) = 0. \tag{39}$$

We can get

$$2y_i\lambda_i - 2\sigma y_i\left[g_i(x) - y_i^2\right] = 0,\quad i = 1, 2, \ldots, m. \tag{40}$$

Namely,

$$y_i\left[\sigma y_i^2 - (\sigma g_i(x) - \lambda_i)\right] = 0,\quad i = 1, 2, \ldots, m. \tag{41}$$

Therefore, when $\sigma g_i(x) - \lambda_i > 0$,

$$y_i^2 = \begin{cases} \frac{1}{\sigma}\left[\sigma g_i(x) - \lambda_i\right] & \sigma g_i(x) - \lambda_i > 0 \\ 0 & \sigma g_i(x) - \lambda_i \le 0 \end{cases}, \quad i = 1, 2, \ldots, m; \tag{42}$$

that is to say,

$$g_i(x) - y_i^2 = \begin{cases} \frac{\lambda_i}{\sigma} & \sigma g_i(x) - \lambda_i > 0 \\ g_i(x) & \sigma g_i(x) - \lambda_i \le 0 \end{cases}, \quad i = 1, 2, \ldots, m. \tag{43}$$

Thus when $\sigma g_i(x) - \lambda_i \le 0$, we have

$$-\lambda_i\left[g_i(x) - y_i^2\right] + \frac{\sigma}{2}\left[g_i(x) - y_i^2\right]^2 = -\lambda_i g_i(x) + \frac{\sigma}{2}\left[g_i(x)\right]^2 = \frac{1}{2\sigma}\left[\left(\sigma g_i(x) - \lambda_i\right)^2 - \lambda_i^2\right]. \tag{44}$$

And when $\sigma g_i(x) - \lambda_i > 0$, we can obtain

$$-\lambda_i\left[g_i(x) - y_i^2\right] + \frac{\sigma}{2}\left[g_i(x) - y_i^2\right]^2 = -\frac{1}{\sigma}\lambda_i^2 + \frac{1}{2\sigma}\lambda_i^2 = -\frac{1}{2\sigma}\lambda_i^2. \tag{45}$$

According to the above two cases,

$$-\lambda_i\left[g_i(x) - y_i^2\right] + \frac{\sigma}{2}\left[g_i(x) - y_i^2\right]^2 = \frac{1}{2\sigma}\left(\left[\min\{0, \sigma g_i(x) - \lambda_i\}\right]^2 - \lambda_i^2\right). \tag{46}$$

Substituting it into formula (38), we can get the corresponding augmented Lagrange function of (36):

$$\widetilde{\phi}(x, \lambda, \sigma) = \min_y \widetilde{\psi}(x, y, \lambda, \sigma) = f(x) + \frac{1}{2\sigma}\sum_{i=1}^{m}\left(\left[\min\{0, \sigma g_i(x) - \lambda_i\}\right]^2 - \lambda_i^2\right). \tag{47}$$

Since the multiplier vector needs to be updated to solve the inequality constrained optimization problems (36) by using the multiplier method, next we derive the multiplier iterative formula. Firstly, fix the penalty parameter σ to some value $\sigma_k > 0$ at its k-th iteration, and fix λ at the current estimate λ_k. Secondly, perform minimization with respect to x. Using x_k to denote the approximate minimizer of $\widetilde{\varphi}(x, y, \lambda, \sigma)$, then we can get by the optimality conditions for unconstrained minimization that

$$0 \approx \nabla_x \widetilde{\varphi}\left(x_k, y^k, \lambda^k, \sigma_k\right) = \nabla f(x_k) - \sum_{i=1}^{m}\left[\lambda_i^k - \sigma_k\left(g_i(x_k) - \left(y_i^k\right)^2\right)\right]\nabla\left(g_i(x_k) - \left(y_i^k\right)^2\right). \tag{48}$$

Let (x^*, y^*, λ^*) satisfy the KKT conditions for (37), then we have

$$\nabla f(x^*) - \sum_{i=1}^{m}\lambda_i^*\nabla\left(g_i(x^*) - \left(y_i^*\right)^2\right) = 0. \tag{49}$$

By comparing (48) with (49), we can deduce that

$$\lambda_i^* \approx \lambda_i^k - \sigma_k\left[g_i(x_k) - \left(y_i^k\right)^2\right], \quad i = 1, 2, \ldots, m. \tag{50}$$

According to (50), to improve the current estimate λ^k of the Lagrange multiplier vectors, the multiplier iteration formula can be given by the following form:

$$\lambda_i^{k+1} = \lambda_i^k - \sigma_k\left[g_i(x_k) - \left(y_i^k\right)^2\right]. \tag{51}$$

Then, taking (43) into the multiplier iteration formula (51), we have

$$\lambda_i^{k+1} = \begin{cases} 0, & \sigma_k g_i(x_k) - \lambda_i^k > 0, \\ \lambda_i^k - \sigma_k g_i(x_k), & \sigma_k g_i(x_k) - \lambda_i^k \le 0. \end{cases} \tag{52}$$

Furthermore, it can be written as

$$\lambda_i^{k+1} = \max\left\{0, \lambda_i^k - \sigma_k g_i(x_k)\right\} \ge 0, \quad i = 1, 2, \ldots, m. \tag{53}$$

Similarly, take (43) into the termination criterion

$$\left(\sum_{i=1}^{m}\left[g_i(x_k) - \left(y_i^k\right)^2\right]^2\right)^{(1/2)} \le \varepsilon. \tag{54}$$

We can get

$$\left(\sum_{i=1}^{m}\left[\min\left\{g_i(x_k), \frac{\lambda_i^k}{\sigma_k}\right\}\right]^2\right)^{(1/2)} \le \varepsilon. \tag{55}$$

3.2.2. Multiplier Method for Solving Model. Next, we use the multiplier method to solve the model (35). Firstly, we construct the augmented Lagrange function for solving model (35):

$$\psi\left(\mathbf{U}^h, \lambda, \sigma\right) = \mathcal{J}_\alpha\left(\mathbf{U}^h\right) + \frac{1}{2\sigma}\sum_{i=1}^{N}\left(\left[\min\left\{0, \sigma C_i^h\left(\mathbf{U}^h\right) - \lambda_i\right\}\right]^2 - \lambda_i^2\right). \tag{56}$$

The corresponding multiplier iteration formula has the following form:

$$\lambda_i^{k+1} = \max\left\{0, \lambda_i^k - \sigma_k C_i^h\left(\mathbf{U}^{h(k)}\right)\right\}. \tag{57}$$

And the corresponding stopping criterion is

$$\beta_k = \left(\sum_{i=1}^{N}\left[\min\left\{C_i^h\left(\mathbf{U}^{h(k)}\right), \frac{\lambda_i^k}{\sigma_k}\right\}\right]^2\right)^{(1/2)} \le \varepsilon. \tag{58}$$

Although the augmented Lagrangian function (57) of the model (35) contains the min function, it is still continuously differentiable; for details, see [37, 44]. The detailed steps of

the multiplier method for solving the model (35) can be summarized by Algorithm 1.

In Algorithm 1, the Gauss–Newton method is used to solve the unconstrained subproblem (56). And its main idea is to use a quadratic function $\widehat{\psi}$ instead of ψ near the iteration value $\mathbf{U}^{h(k)}$ of the previous step by the Taylor expansion given below:

$$\psi\left(\mathbf{U}^{h(k)}+\delta_{\mathbf{U}^h}\right)\approx\widehat{\psi}\left(\mathbf{U}^{h(k)}+\delta_{\mathbf{U}}^h\right)=\psi\left(\mathbf{U}^{h(k)}\right)+d\psi\left(\mathbf{U}^{h(k)}\right)\delta_{\mathbf{U}^h}+\frac{1}{2}\delta_{\mathbf{U}^h}^{\top}\mathbf{H}\delta_{\mathbf{U}^h},\tag{59}$$

where $d\psi(\mathbf{U}^{h(k)})$ is the Jacobian matrix of ψ at $\mathbf{U}^{h(k)}$ and $\mathbf{H}$ is the approximation of its Hessian. Due to $d^2\mathscr{D}^h(\mathbf{U}^{h(k)})$, $\mathbb{C}^{\top}\mathbb{C}$ and $(M(\mathbf{U}^{h(k)}))^{\top}M(\mathbf{U}^{h(k)})$ are both positive semidefine, and it is easy to prove that $\mathbf{H}$ is also positive semidefinite. Thus, we know that $\widehat{\psi}$ is convex. In this way, the nonconvex problem can be transformed into a convex problem to be solved. For further detailed description, see [37]. The detailed steps are described below.

Given the initial value $\mathbf{U}^{h(k)}$, the Jacobian matrix $d\psi(\mathbf{U}^{h(k)})$ and the Hessian matrix $\mathbf{H}$ are calculated by the following forms:

$$d\psi\left(\mathbf{U}^{h(k)}\right)=d\mathscr{D}^h\left(\mathbf{U}^{h(k)}\right)+\alpha h_d\,\mathbb{C}^{\top}\mathbb{C}\,\mathbf{U}^{h(k)}+\left(M\left(\mathbf{U}^{h(k)}\right)\right)\circledast\left(\sigma C^h\left(\mathbf{U}^{h(k)}\right)-\boldsymbol{\lambda}\right),\tag{60}$$

$$\mathbf{H}=d^2\mathscr{D}^h\left(\mathbf{U}^{h(k)}\right)+\alpha h_d\mathbb{C}^{\top}\mathbb{C}+\left(M\left(\mathbf{U}^{h(k)}\right)\right)\left(M\left(\mathbf{U}^{h(k)}\right)\right)^{\top},\tag{61}$$

and in each outer iteration step, respectively, where $M(\mathbf{U}^{h(k)})=dC^h(\mathbf{U}^h)\in\mathbb{R}^{2N}$ is defined by (34). In formula (59), the disturbance value $\delta_{\mathbf{U}^h}$ can be obtained by finding the stability point of the quadratic function $\widehat{\psi}$, namely,

$$\mathbf{H}\delta_{\mathbf{U}^h}=-d\psi\left(\mathbf{U}^{h(k)}\right).\tag{62}$$

Usually, $\mathbf{H}$ is the positive definite. Thus the equation (62) can be solved by the preconjugate gradient method. To ensure that the objective function (59) is descending, the standard Armijo line search method can be used. The specific steps for the Armijos line search can be summarized by using Algorithm 2. And the Gauss–Newton method mentioned above is described by using Algorithm 3. In order to provide a good initial value, the Gauss–Newton method with Armijo line search and the multilevel method are combined to solve model (56), which can reduce the risk of getting trapped at an unwanted minimizer and save computing time. Firstly, we use the initial value $\mathbf{U}^{h(0)}$ and the Gauss–Newton method with the Armijos line search to solve (56) on the coarsest level. Secondly, the solution on the coarsest level is interpolated to the next finer level; next it is used as the initial value, and the same method is used to solve the model (56) on the finer level, where bilinear interpolation operator I_H^h is used. Finally, this process is repeated until the loop terminates. We summarize the multilevel method using Algorithm 4.

4. Numerical Experiments

In this part, we use three experiments to show that our new model (CLC) has good performance by comparing it with diffeomorphic demons (DDemons) [43], linear curvature model (LC) [28], mean curvature model (MC) [8], hyperelastic regularizer (Hyper) [20], and Zhang–Chen model (ZC) [42]. In order to illustrate the capabilities of our model, we select two pairs of artificial images. In addition, considering the important application of image registration in biomedical images, we also use a pair of medical lung images for experiments.

In order to quantify the quality of the registered image, the relative reduction ε of the dissimilarity, which is given by [7], is shown as follows:

$$\varepsilon=\frac{\mathscr{D}(\mathbf{u})}{\mathscr{D}_{\text{stop}}}\times 100\%.\tag{63}$$

And the minimum value $\mathscr{F}$ of the determinant of the Jacobian matrix J of the transformation φ

$$J=\begin{bmatrix}1+u_{1x} & u_{1y}\\ u_{2x} & 1+u_{2y}\end{bmatrix},\qquad \mathscr{F}=\min(\det(J)),\tag{64}$$

are used, where $\mathscr{D}(\mathbf{u})$ is defined by equation (4), $\mathbf{u}$ is the current iteration, and $\mathscr{D}_{\text{stop}}$ is the value of $\mathscr{D}(\mathbf{u})$ at $\mathbf{u}=0$.

4.1. Test 1: A Pair of Lena Images. In the first experiment, we used a pair of Lena images with a size of 256×256. The test images and registered ones using our new model are shown in Figure 1. The transformed template image obtained using the other five models and the image difference after registration are represented by Figure 2. For Example 1 shown in Figures 1(a) and 1(b), the registration results using our new model and linear curvature (LC [28]) model and other four models are recorded in Tables 1 and 2, respectively. For a smaller regularizer parameter $\alpha=3.12e-3$, from Figures 1 and 2, we can see that our new model and hyperelastic regularizer-, linear curvature-, and mean one-based image registration models can produce visually satisfactory registration results. However, we find that the latter two have mesh folding by Tables 1 and 2. Although the demons-based registration model and ZC model do not produce folding, the registration effects are relatively poor. From Tables 1 and 2, we can further see that for the same image with different sizes, our new model can give very satisfactory registration results without folding.

In order to analyze how much our model will be affected when changing the regularizer parameter α, we use Algorithm 4 to test Example 1, and the corresponding quantitative measurement values are recorded in Tables 3–5. From Tables 3 to 5, we find that the registration quality becomes worse and worse with the increase of the regularizer parameter α. However, when alpha is greater than or equal to 3.12×10^{-3}, our new model does not produce folding for Example 1. For this example, our new model is able to produce better registration results, especially when the

Step 1: input the initial value: $\mathbf{U}^{h(0)} \in \mathbb{R}^{2N}$, the objective function $\mathcal{J}_\alpha(\mathbf{U}^h)$ and its gradient $d\mathcal{J}_\alpha(\mathbf{U}^h)$, inequality constrained vector $C^h(\mathbf{U}^h)$, and the transpose of its Jacobian matrix $dC^h(\mathbf{U}^h)$; let $\max k := 10$, $\sigma_1 := 1$, $\varepsilon := 10^{-5}$, $\vartheta := 0.3$, $\eta := 0.2$, $k := 0$, $\lambda^1 := (10^{-3}, 10^{-3}, \ldots, 10^{-3})^\top \in \mathbb{R}^N$, and $\beta_k := 10$.
Step 2: solving the subproblem. With $\mathbf{U}^{h(k-1)}$ as the initial point, solve the minimum value $\mathbf{U}^{h(k)}$ of the unconstrained subproblem (51) by using the Gauss–Newton scheme with Armijo line search.
Step 3: check the termination condition. If $\beta_k \le \varepsilon$ or $k > \max k$, where β_k is defined by (57), the iteration is stopped, and $\mathbf{U}^{h(k)}$ is output as the approximate minimum of the original problem; otherwise, go to Step 4.
Step 4: update penalty parameters. If $\beta_k \ge \vartheta\beta_{k-1}$, let $\sigma_{k+1} := \eta\sigma_k$; otherwise, set $\sigma_{k+1} := \sigma_k$.
Step 5: update multiplier vector. Calculate $\lambda_i^{k+1} = \max\{0, \lambda_i^k - \sigma_k C_i^h(\mathbf{U}^{h(k)})\}$, $i = 1, 2, \ldots, N$.
Step 6: set $k := k + 1$, and go to Step 1.

Algorithm 1: Multiplier scheme.

Step 1: input initial value: $\mathbf{U}^{h(k)} \in \mathbb{R}^{2N}$; $\delta_{\mathbf{U}^h} \in \mathbb{R}^N$, $\psi(\mathbf{U}^{h(k)})$, and $d\psi(\mathbf{U}^{h(k)})$ are calculated by (56) and (60), respectively. Set $t := 1$, maxIter $:= 10$, $\eta := 10^{-4}$, and $k := 0$.
Step 2: set $\mathbf{U}_t^{h(k)} := \mathbf{U}^{h(k)} + t\delta_{\mathbf{U}^h}$, and compute $\psi(\mathbf{U}_t^{h(k)})$.
Step 3: check the termination condition. If $\psi(\mathbf{U}_t^{h(k)}) < \psi(\mathbf{U}^{h(k)}) + t\eta(d\psi(\mathbf{U}^{h(k)}))^\top \delta_{\mathbf{U}^h}$, then the iteration is stopped; otherwise, go to Step 4.
Step 4: Set $t := t/2$; $k := k + 1$; and go to Step 2.

Algorithm 2: Armijo line search method.

Step 1: set $k := 0$, maxIter: $= 10$, and $\varepsilon = 2.2e - 13$; input initial value: $\mathbf{U}^{h(k)} \in \mathbb{R}^{2N}$.
Step 2: compute $\psi(\mathbf{U}^{h(k)})$, $d\psi(\mathbf{U}^{h(k)})$, and $\mathbf{H}$ by using (56), (60), and (61), respectively.
Step 3: check the termination condition. If $d\psi(\mathbf{U}^{h(k)}) \le \varepsilon$ or $k \ge$ maxIter, stop, and $(\mathbf{U}^{h(*)}) := \mathbf{U}^{h(k)}$ is the output.
Step 4: Solve the Quasi–Newton equation (62) by Preconjugate Gradient method. If the equation (61) has a solution $\delta_{\mathbf{U}^h}$ which meets $d\psi(\mathbf{U}^{h(k)})\delta_{\mathbf{U}^h} < 0$, then go to Step 5; otherwise, set $\delta_{\mathbf{U}^h} = -d\psi(\mathbf{U}^{h(k)})$, and go to Step 5.
Step 5: the step factor t is solved by the Armijos line search technique.
Step 6: set $\mathbf{U}_t^{h(k)} := \mathbf{U}^{h(k)} + t\delta_{\mathbf{U}^h}$, $k := k + 1$; then go to Step 2.

Algorithm 3: The Gauss–Newton method.

Step 1: given initial values: Maxlevel: $=$ ceil$(\log 2(\min(m_1, m_2)))$ and Minlevel: $= 3$; given multilevel representation of the reference image R and the template image T.
Step 2: set $\mathbf{U}^{h(0)} = 0$ on the coarsest level $l =$ Minlevel, then solve (51) by using Algorithm 1; otherwise, go to Step 3.
Step 3: the initial value $\mathbf{U}^{h(0)}$ on the finer level is obtained by interpolation operator I_H^h.
Step 4: set $l := l + 1$; then go to Step 2.

Algorithm 4: Multilevel method.

(a) (b) (c)

Figure 1: Continued.

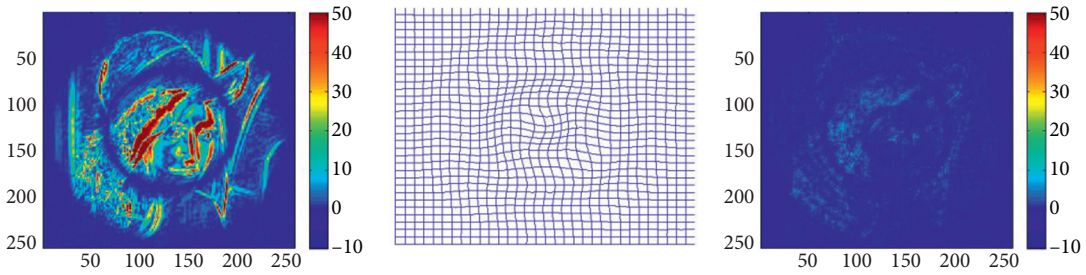


Figure 1: Registration results of testing Lena images using our new model. (a) Reference image, (b) template image, (c) the transformed template image using our new model, (d) image difference before registration ($\varepsilon = 100\%$), (e) the transformation $\mathbf{x} + \mathbf{u}(\mathbf{x})$, and (f) image difference after registration ($\varepsilon = 0.36\%$).

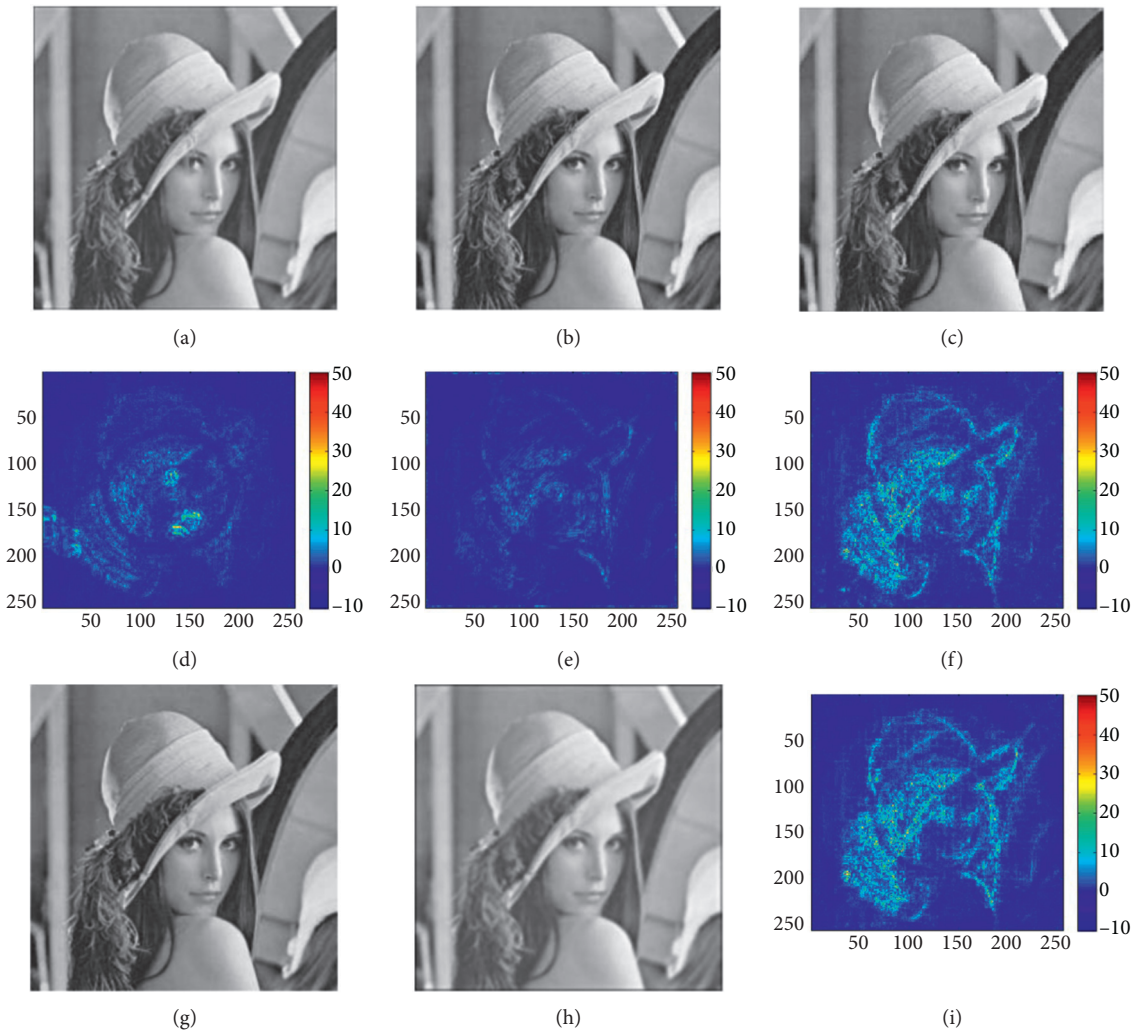
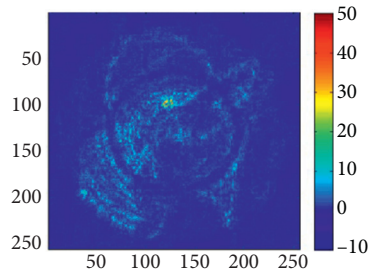


Figure 2: Continued.



(j)

Figure 2: Registration results for a pair of Lena images with a size of 256×256 using other five models. The transformed template image $T(\mathbf{x} + \mathbf{u}(\mathbf{x}))$ from (a) the LC model [28], (b) Hyper model [20], and (c) DDemons model [43]. The image difference after registration from (d) the LC model ($\varepsilon = 1.48\%$) [28], (e) Hyper model [20] ($\varepsilon = 0.52\%$), and (f) DDemons model [43] ($\varepsilon = 2.12\%$). The transformed template image $T(\mathbf{x} + \mathbf{u}(\mathbf{x}))$ from (g) the ZC model [42] and (h) MC model [8]. The image difference after registration from (i) the ZC model [42] ($\varepsilon = 3.48\%$) and (j) MC model [8] ($\varepsilon = 1.41\%$).

Table 1: Quantitative measurements using our new model and LC model [28] for processing Example 1 shown in Figures 1(a) and 1(b).

Layer	Model					
	LC $\alpha = 3.12e-3$			CLC $\alpha = 3.12e-3$		
	$\varepsilon(\%)$	$\mathcal{N}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{N}$	$\mathcal{F}$
$h = (1/256)$	1.48	315	−1.2228	0.36	0	0.5791
$h = (1/128)$	1.16	92	−0.8856	0.22	0	0.4997
$h = (1/64)$	1.04	20	−0.7184	0.39	0	0.5946
$h = (1/32)$	0.66	4	−0.3730	0.40	0	0.0168
$h = (1/16)$	1.23	1	−0.5629	1.04	0	0.0486

$\mathcal{N}$ represents the number of mesh folding of the transformation $\mathbf{x} + \mathbf{u}(\mathbf{x})$. $\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 2: Quantitative measurements using the DDemons model [43], ZC model [42], Hyper model [20], and MC model [8] for processing Example 1 shown in Figures 1(a) and 1(b).

Layer	Model											
	DDemons $\sigma_{\text{fluid}} = 1$, $\sigma_{\text{diff}} = 0.756$			ZC $\alpha = 3.12e-3$			Hyper $\alpha = 3.12e-3$			MC $\alpha = 3.12e-3$		
	$\varepsilon(\%)$	$\mathcal{N}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{N}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{N}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{N}$	$\mathcal{F}$
$h = (1/256)$	2.12	0	0.0434	3.48	0	0.0649	0.52	0	0.3038	1.41	8	−0.1055
$h = (1/128)$	4.63	0	0.1764	3.29	0	0.0660	1.05	0	0.6497	1.14	1	−0.0273
$h = (1/64)$	11.99	0	0.5432	6.43	0	0.1191	4.12	0	0.7116	3.04	0	0.1613
$h = (1/32)$	23.69	0	0.6899	10.33	0	0.0440	15.84	0	0.7193	2.91	2	−0.2810
$h = (1/16)$	48.14	0	0.8650	28.45	0	0.3792	85.55	0	0.8247	2.17	0	0.1656

$\mathcal{N}$ represents the number of mesh folding of the transformation $\mathbf{x} + \mathbf{u}(\mathbf{x})$. $\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 3: Dependent results of our new model on the regularizer parameter α by testing Example 1 (Figures 1(a) and 1(b)).

$\alpha = e^{-3}$	3.12	4	5	8	10	30
ε (%)	0.3574	0.3734	0.3876	0.4169	0.4307	0.4996
$\mathcal{F}$	0.5791	0.5968	0.6119	0.6387	0.6493	0.6826

$\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 4: Dependent results of our new model on the regularizer parameter α by testing Example 1 (Figures 1(a) and 1(b)).

$\alpha = e^{-3}$	80	100	200	300	400	500
ε (%)	0.5836	0.6118	0.7488	0.8903	1.0380	1.1914
$\mathcal{F}$	0.6736	0.6719	0.6678	0.6669	0.6679	0.6699

$\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 5: Dependent results of our new model on the regularizer parameter α by testing Example 1 (Figures 1(a) and 1(b)).

$\alpha = e^{-3}$	$1e3$	$2e3$	$3e3$	$4e3$	$5e3$	$6e3$
ε (%)	2.0285	3.8424	5.6738	7.4549	9.200	10.8363
$\mathcal{F}$	0.6895	0.7350	0.7534	0.7688	0.7798	0.7878

$\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

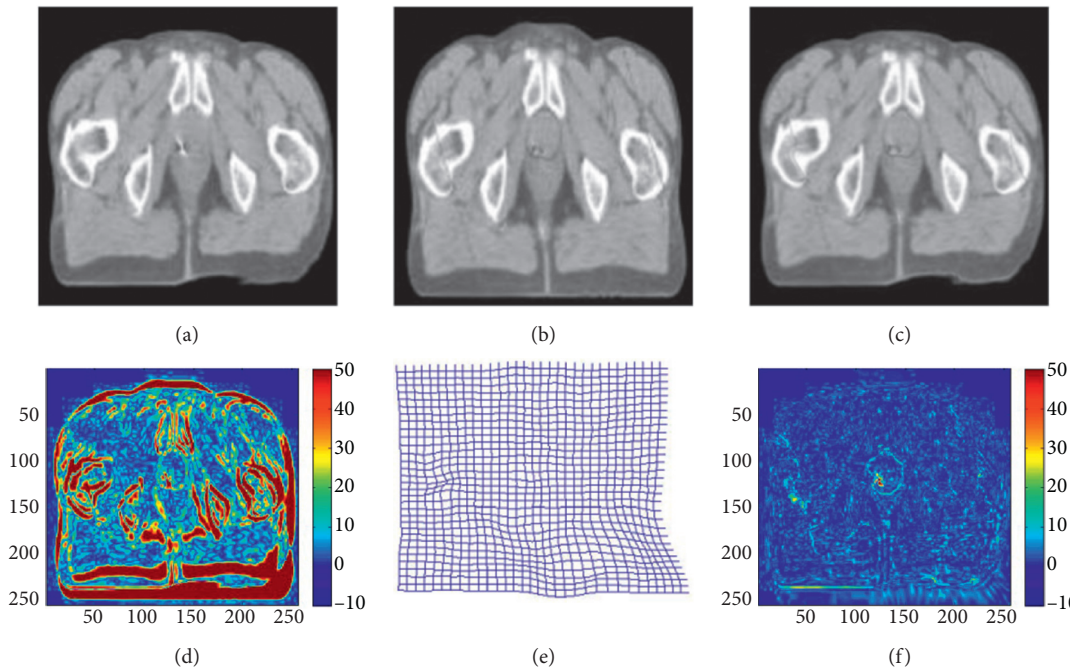


Figure 3: Registration results of testing a pair of medical lung images using our new model. (a) Reference image, (b) template image, (c) the transformed template image using our new model, (d) image difference before registration ($\varepsilon = 100\%$), (e) the transformation $\mathbf{x} + \mathbf{u}(\mathbf{x})$, and (f) image difference after registration ($\varepsilon = 1.23\%$).

regularizer parameter α is taken in an appropriate range $[3.12 \times 10^{-3}, 3 \times 10^{-1}]$.

4.2. Test 2: A Pair of Medical Lung Images. In this part, we select a pair of medical lung images with a size of 256×256 for testing. For each model, we choose the optimal parameters. The test images and the corresponding registered ones are shown in Figures 3 and 4, respectively. The registration results on different layers are recorded in Tables 6 and 7, respectively. According to Figures 3 and 4, we can observe that the other five models can produce visually relatively satisfactory registration effects except the DDemons model. From the image difference after registration and the results recorded in Tables 6 and 7, we find that although our model requires slightly more computing time than the LC model and MC model when the resolution (256×256) is larger, it produces more satisfactory registration effects than them. In addition, from the above tables we can also see that when the resolution ($\leq 128 \times 128$) of the image is relatively small, our model can produce more satisfactory registration effects than the other models in a relatively short time. Although the ZC model and Hyper model produce slightly better results than our new model for the image with the size of 256×256, our new model is more robust with respect to mesh parameter h. As can be seen from Tables 8–10, with the increase of the regularizer parameter α, the registration effects gradually become worse. But when α is taken in a certain range $[0.02846, 0.1]$, our model can still produce satisfactory registration effects. From the tables above, we also find that when the resolution of the image changes and the regularizer parameter α is taken in a certain range, the change of registration results generated by our new model is not particularly significant. This shows that our new model is more robust.

4.3. Test 3: A Pair of Artificial Image. A pair of artificial image with a size of 256×256 is used in the third experiment. For each model, we still choose its optimal parameters. Figure 5 represents the test images and registration results using our new model (CLC). The transformed template images and image differences after registration from all other models are shown in Figure 6. The registration results on different layers using our new model (CLC) and LC model [28] and other four models are summarized in Tables 11 and 12, respectively. In Figures 5 and 6, we can see that all five models except the DDemons model are fine in producing

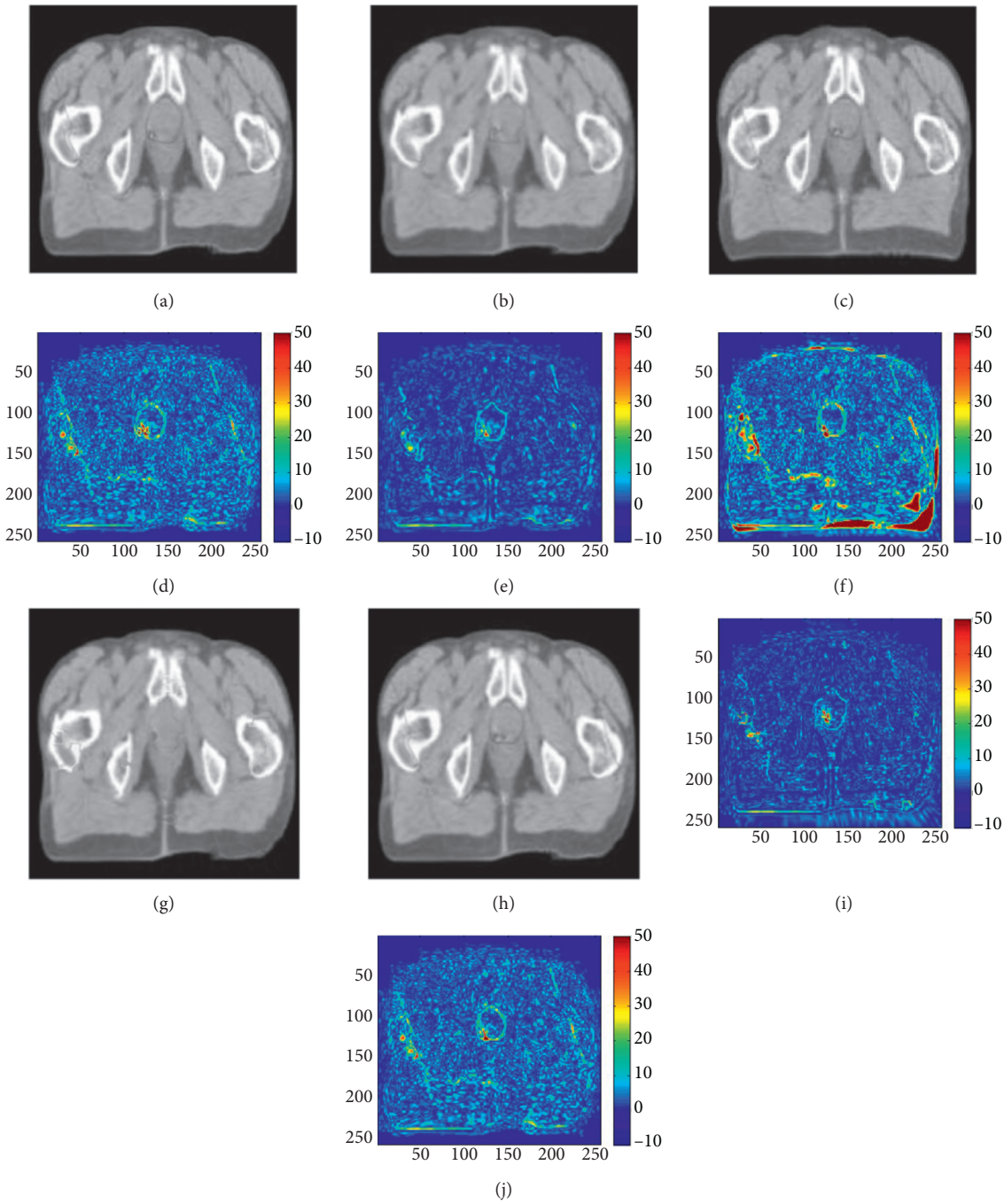


Figure 4: Registration results for a pair of lung images with size of 256×256 using other five models. The transformed template image $T(\mathbf{x} + \mathbf{u}(\mathbf{x}))$ from (a) the LC model [28], (b) Hyper model [20], and (c) DDemons model [43]. The image difference after registration from (d) the LC model [28] ($\varepsilon = 2.41\%$), (e) Hyper model [20] ($\varepsilon = 1.01\%$), and (f) DDemons model [43] ($\varepsilon = 16.74\%$).The transformed template image $T(\mathbf{x} + \mathbf{u}(\mathbf{x}))$ from (g) the ZC model [42] and (h) MC model [8]. The image difference after registration from (i) the ZC model [42] ($\varepsilon = 1.22\%$) and (j) MC model [8] ($\varepsilon = 2.15\%$).

Table 6: Quantitative measurements using our new model and LC model [25] for processing Example 2 shown in Figures 3(a) and 3(b).

Layer	Model					
	LC $\alpha = 4.84e-2$			CLC $\alpha = 2.846e-2$		
	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$
$h = (1/256)$	2.41	14.5	0.0006	1.23	44.1	0.0186
$h = (1/128)$	1.96	7.6	0.0910	0.76	8.9	0.2540
$h = (1/64)$	1.17	4.0	0.1363	0.43	2.6	0.4581
$h = (1/32)$	1.07	2.6	0.2579	0.29	1.1	0.5437
$h = (1/16)$	0.81	1.2	−0.3257	0.13	0.5	0.6884

$\mathcal{T}$ represents the total run time which includes the output of the image (seconds). $\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 7: Quantitative measurements using the DDemons model [43], ZC model [42], Hyper model [20], and MC model [8] for processing Example 2 shown in Figures 3(a) and 3(b).

Layer	Model											
	DDemons $\sigma_{\text{fluid}} = 2$, $\sigma_{\text{diff}} = 2.336$			ZC $\alpha = 2$, $\beta = 8$			Hyper $\alpha = 10$, $\beta = 260$			MC $\alpha = 8e-2$		
	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$
$h = (1/256)$	16.74	12.1	0.2791	1.22	52.1	0.0013	1.01	13.6	0.0215	2.15	15.4	0.0080
$h = (1/128)$	16.17	6.7	0.0719	1.68	9.4	0.0029	1.14	3.6	0.4266	1.52	5.6	0.0844
$h = (1/64)$	22.72	3.8	0.4960	2.38	4.6	0.0091	1.46	1.7	0.4859	0.87	3.1	0.2770
$h = (1/32)$	20.29	4.3	0.8279	4.34	1.0	0.0148	3.18	1.2	0.5913	0.50	2.0	0.4201
$h = (1/16)$	35.67	3.0	0.9464	16.48	0.4	0.0459	26.01	0.3	0.6572	0.37	1.2	0.5814

$\mathcal{T}$ represents the total run time which includes the output of the image(seconds). $\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 8: Dependent results of our new model on the regularizer parameter α by testing Example 2 (Figures 3(a) and 3(b)).

α	0.02846	0.03	0.04	0.05	0.06	0.07
ε (%)	1.2282	1.2364	1.3463	1.4268	1.4926	1.5496
$\mathcal{F}$	0.0186	0.0042	0.0894	0.1836	0.2524	0.3152

$\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 9: Dependent results of our new model on the regularizer parameter α by testing Example 2 (Figures 3(a) and 3(b)).

α	0.1	0.2	0.5	1	5	10
ε (%)	1.6850	1.9600	2.3250	2.6287	3.6470	4.3160
$\mathcal{F}$	0.4019	0.5192	0.6901	0.7352	0.8425	0.8998

$\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 10: Dependent results of our new model on the regularizer parameter α by testing Example 2 (Figures 3(a) and 3(b)).

α	50	100	500	1000	5000	10000
ε (%)	6.9837	8.9523	15.3188	18.2166	21.3379	23.6637
$\mathcal{F}$	0.9903	0.9887	0.9832	0.9818	0.9820	0.9802

$\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

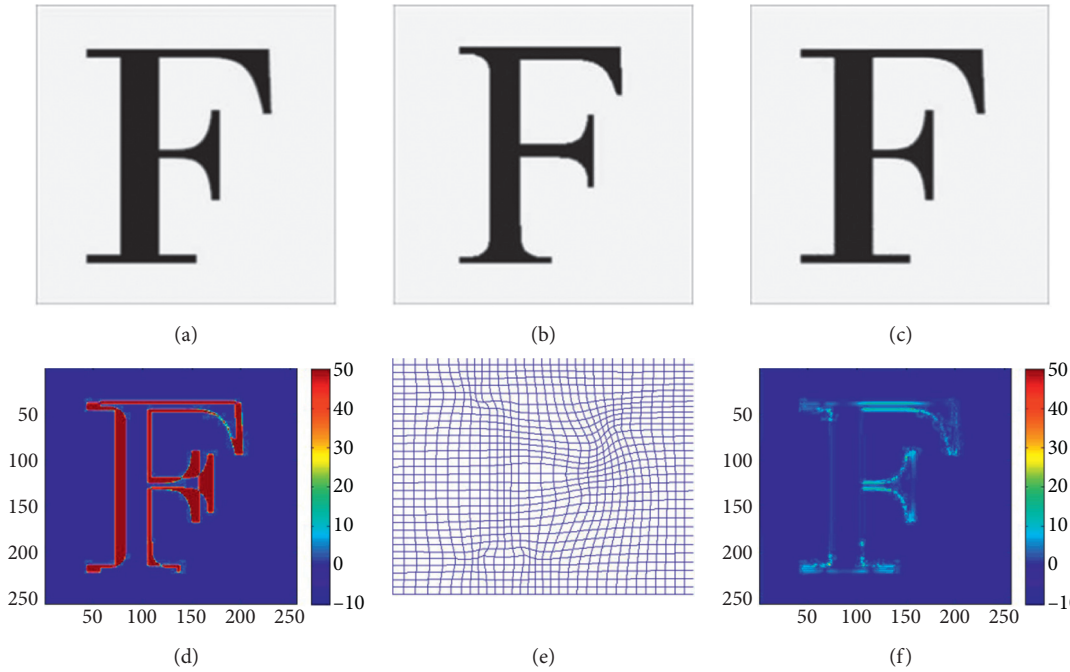


FIGURE 5: Registration results of testing a pair of artificial images using our new model. (a) Reference image, (b) template image, and (c) the transformed template image using our new model (CLC). The image difference (d) before registration ($\varepsilon = 100\%$), (e) the transformation, and (f) after registration using our new model (CLC) ($\varepsilon = 0.07\%$), respectively.

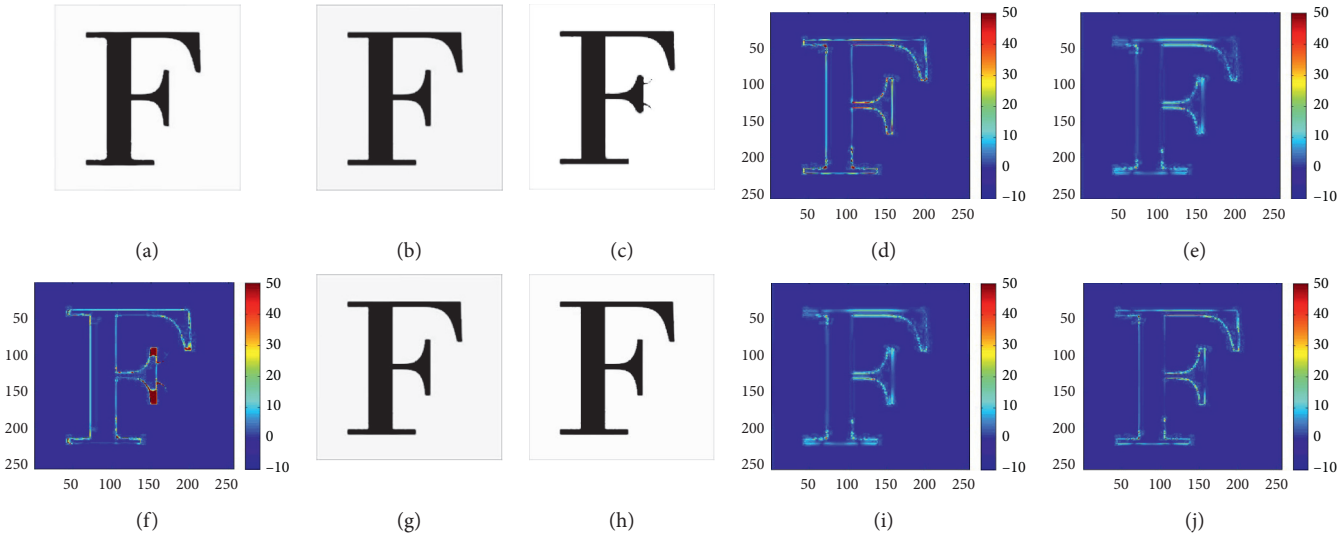


FIGURE 6: Registration results for a pair of lung images with size of 256×256 using other five models. The transformed template image $T(\mathbf{x} + \mathbf{u}(\mathbf{x}))$ from (a) the LC model [28], (b) Hyper model [20], and (c) DDemons model [43]. The image difference after registration from (d) the LC model [28] ($\varepsilon = 0.73\%$), (e) Hyper model [20] ($\varepsilon = 0.10\%$), and (f) DDemons model [43] ($\varepsilon = 5.26\%$). The transformed template image $T(\mathbf{x} + \mathbf{u}(\mathbf{x}))$ from (g) ZC model [42] and (h) MC model [8]. The image difference after registration from (i) the ZC model [42] ($\varepsilon = 0.12\%$) and (j) MC model [8] ($\varepsilon = 0.19\%$).

satisfactory registration results. According to Tables 11 and 12, we find that although our new model requires slightly more computing time when the resolution (256×256) is large, it has the best value of ε. We also find that LC model and MC model have mesh folding during the registration process. In addition, our new model is more robust than the ZC model and Hyper model with the change of grid parameter h. For this example, an accurate regularizer parameter α is also not needed. From Tables 13 to 15, we also find our proposed new model can produce acceptable registration results for any α between 0.1 and 1.

Table 11: Quantitative measurements using our new model (CLC) and LC model [25] for processing Example 3.

Layer	Model					
	LC $\alpha = 4.2$			CLC $\alpha = 1e-1$		
	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$
$h = (1/256)$	0.73	11.0	0.4683	0.07	44.0	0.4410
$h = (1/128)$	0.88	5.1	0.4903	0.08	10.5	0.4554
$h = (1/64)$	1.58	4.0	0.5074	0.18	2.3	0.2405
$h = (1/32)$	9.40	2.6	−0.5134	0.23	1.1	0.3042
$h = (1/16)$	9.43	1.7	−0.0834	1.38	0.4	0.2617

$\mathcal{T}$ represents the total run time which includes the output of the image(seconds). $\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 12: Quantitative measurements using the DDemons model [43], ZC model [42], Hyper model [20], and MC model [8] for processing Example 3 shown in Figures 5(a)–5(c).

Layer	Model											
	DDemons $\sigma_{\text{fluid}} = 1$, $\sigma_{\text{diff}} = 0.4$			ZC $\alpha = 35$, $\beta = 40$			Hyper $\alpha = 240$, $\beta = 300$			MC $\alpha = 4e-1$		
	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$	$\varepsilon(\%)$	$\mathcal{T}$	$\mathcal{F}$
$h = (1/256)$	5.26	29.6	0.0581	0.12	1.8	0.1436	0.10	3.3	0.4233	0.19	9.5	0.4524
$h = (1/128)$	1.96	24.1	0.0499	0.38	1.2	0.1376	0.52	2.3	0.5167	0.19	5.6	0.4416
$h = (1/64)$	1.39	4.7	0.3079	2.35	0.9	0.0541	1.62	1.8	0.5216	0.78	3.8	−0.5506
$h = (1/32)$	9.34	7.4	0.4454	7.24	0.6	0.0633	6.04	0.6	0.5640	0.67	2.5	−0.5497
$h = (1/16)$	5.44	6.3	0.6979	12.17	0.5	0.0682	36.82	0.5	0.5940	0.66	1.6	0.1492

$\mathcal{T}$ represents the total run time which includes the output of the image (seconds). $\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 13: Dependent results of our new model on the regularizer parameter α by testing Example 3 (Figures 5(a)–5(c)).

α	0.1	0.2	0.3	0.4	0.5	0.6
ε (%)	0.0673	0.1104	0.1398	0.1634	0.1839	0.2025
$\mathcal{F}$	0.4410	0.4508	0.4566	0.4610	0.4647	0.4679

$\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 14: Dependent results of our new model on the regularizer parameter α by testing Example 3 (Figures 5(a)–5(c)).

α	0.7	0.8	0.9	1.0	2.0	3.0
ε (%)	0.2198	0.2364	0.2523	0.2678	0.4118	0.5519
$\mathcal{F}$	0.4707	0.4732	0.4717	0.4705	0.4674	0.4690

$\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

Table 15: Dependent results of our new model on the regularizer parameter α by testing Example 3 (Figures 5(a)–5(c)).

α	4.0	5.0	6.0	10.0	20.0	30.0
ε (%)	0.6938	0.8386	0.9859	1.5980	3.0317	5.0536
$\mathcal{F}$	0.4711	0.4717	0.4721	0.4771	0.4791	0.5214

$\mathcal{F} > 0$ means that the transformation does not include folding or cracking.

5. Conclusions

Avoiding mesh folding is a key issue to ensure the invertibility of transformation in the image registration model. In this paper, we propose a constrained linear curvature image registration model by integrating the evaluation criteria to measure the registration results directly into the basic framework of the variational model. In order to solve the new model, we use a combination of the multiplier method and Gauss–Newton scheme with the Armijos line search and further combine with a multilevel method to achieve fast convergence. To illustrate the good performance of our new

model, we compare it with five representative models based on the LC model [28], MC model [8], DDemons model [43], Hyper model [20], and ZC model [42] using three numerical examples. Numerical experiments show that our proposed model is more efficient and robust than the competing models. Future works will consider the use of homotopy method to solve the corresponding inequality constraint registration model.

Data Availability

The image datasets used to support the findings of this study are included within the article.

Conflicts of Interest

The author declares no conflicts of interest.

Acknowledgments

This research work was supported by the Natural Science Foundation of China (NSFC) (No. 11801249) and Natural Science Foundation of Shandong Province (No. ZR2017BA034).

References

[1] L. G. Brown, "A survey of image registration techniques," *ACM Computing Surveys*, vol. 24, no. 4, pp. 325–376, 1992.

[2] A. A. Goshtasby, *2-D and 3-D Image Registration: For Medical, Remote Sensing, and Industrial Applications*, Wiley-Interscience, New Jersey, NY, USA, 2006.

[3] B. Zitová and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing*, vol. 21, no. 11, pp. 977–1000, 2003.

[4] A. Sotiras, C. Davatzikos, and N. Paragios, "Deformable medical image registration: a survey," *IEEE Transactions on Medical Imaging*, vol. 32, no. 7, pp. 1153–1190, 2013.

[5] J. Modersitzki, *Numerical Methods for Image Registration*, Oxford University Press, New York, NY, USA, 2004.

[6] G. Aubert and P. Kornprobst, *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations*, Springer, New York, NY, USA, 2006.

[7] J. Modersitzki, *FAIR: Flexible Algorithms for Image Registration*, SIAM Publications, Philadelphia, PA, USA, 2009.

[8] N. Chumchob, K. Chen, and C. Brito-Loeza, "A fourth-order variational image registration model and its fast multigrid algorithm," *Multiscale Modeling & Simulation*, vol. 9, no. 1, pp. 89–128, 2011.

[9] A. Theljani, K. Chen, and K. Chen, "An augmented Lagrangian method for solving a new variational model based on gradients similarity measures and high order regulariation for multimodality registration," *Inverse Problems & Imaging*, vol. 13, no. 2, pp. 309–335, 2019.

[10] M. Jenkinson and S. Smith, "A global optimisation method for robust affine registration of brain images," *Medical Image Analysis*, vol. 5, no. 2, pp. 143–156, 2001.

[11] M. Jenkinson, P. Bannister, M. Brady, and S. Smith, "Improved optimization for the robust and accurate linear registration and motion correction of brain images," *Neuroimage*, vol. 17, no. 2, pp. 825–841, 2002.

[12] M. Xia and B. Liu, "Image registration by "Super-Curves"," *IEEE Transactions on Image Processing*, vol. 13, no. 5, pp. 720–732, 2004.

[13] P. Zhilkin and M. E. Alexander, "Affine registration: a comparison of several programs," *Magnetic Resonance Imaging*, vol. 22, no. 1, pp. 55–66, 2004.

[14] N. Chumchob and K. Chen, "A robust affine image registration method," *International Journal of Numerical Analysis & Modeling*, vol. 6, no. 2, pp. 311–334, 2009.

[15] B. Zhao, G. E. Christensen, J. H. Song et al., "Tissue-volume preserving deformable image registration for 4DCT pulmonary images," in *Proceedings of the CVPR Workshops*, pp. 481–489, Las Vegas, NV, USA, June 2016.

[16] J. Dong, K. Lu, J. Xue, S. Dai, R. Zhai, and W. Pan, "Accelerated nonrigid image registration using improved Levenberg-Marquardt method," *Information Sciences*, vol. 423, pp. 66–79, 2018.

[17] C. L. Chan, C. Anitescu, Y. Zhang, and T. Rabczuk, "Two and three dimensional image registration based on B-spline composition and level sets," *Communications in Computational Physics*, vol. 21, no. 2, pp. 600–622, 2017.

[18] W. Sun, W. J. Niessen, and S. Klein, "Randomly perturbed B-splines for nonrigid image registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1401–1413, 2017.

[19] R. Bajcsy and S. Kovačič, "Multiresolution elastic matching," *Computer Vision, Graphics, and Image Processing*, vol. 46, no. 1, pp. 1–21, 1989.

[20] M. Burger, J. Modersitzki, and L. Ruthotto, "A hyperelastic regularization energy for image registration," *SIAM Journal on Scientific Computing*, vol. 35, no. 1, pp. B132–B148, 2013.

[21] I. N. Figueiredo, C. Leal, L. Pinto, P. N. Figueiredo, and R. Tsai, "Hybrid multiscale affine and elastic image registration approach towards wireless capsule endoscope localization," *Biomedical Signal Processing and Control*, vol. 39, pp. 486–502, 2018.

[22] B. Fischer and J. Modersitzki, "Fast diffusion registration," *Inverse Problems, Image Analysis, and Medical Imaging*, vol. 313, pp. 117–127, 2002.

[23] L. Hömke, C. Frohn-Schauf, S. Henn et al., "Total variation based image registration," in *Proceedings of the International Conference on PDE-Based Image Processing and Related Inverse Problems Series: Mathematics and Visualization*, pp. 305–323, Oslo, Norway, August 2006.

[24] C. Frohn-Schauf, S. Henn, and K. Witsch, "Multigrid based total variation image registration," *Computing and Visualization in Science*, vol. 11, no. 2, pp. 101–113, 2008.

[25] N. Chumchob and K. Chen, "A variational approach for discontinuity-preserving image registration," *East-West Journal of Mathematics*, vol. 2010, pp. 266–282, 2010.

[26] J. Zhang, K. Chen, and B. Yu, "An improved discontinuity-preserving image registration model and its fast algorithm," *Applied Mathematical Modelling*, vol. 40, no. 23-24, pp. 10740–10759, 2016.

[27] J. Zhang and K. Chen, "Variational image registration by a total fractional-order variation model," *Journal of Computational Physics*, vol. 293, pp. 442–461, 2015.

[28] B. Fischer and J. Modersitzki, "Curvature based image registration," *Journal of Mathematical Imaging and Vision*, vol. 18, no. 1, pp. 81–85, 2003.

[29] B. Fischer and J. Modersitzki, "A unified approach to fast image registration and a new curvature based registration technique," *Linear Algebra and Its Applications*, vol. 380, no. 2, pp. 107–124, 2004.

[30] J. Zhang, K. Chen, F. Chen, and B. Yu, "An efficient numerical method for mean curvature-based image registration model," *East Asian Journal on Applied Mathematics*, vol. 7, no. 1, pp. 125–142, 2017.

[31] I. Mazlinda, K. Chen, and C. Brito, "A novel variational model for image registration using Gaussian curvature," *Journal of Geometry, Imaging and Computing*, vol. 1, no. 4, pp. 417–446, 2015.

[32] E. Haber and J. Modersitzki, "Image registration with guaranteed displacement regularity," *International Journal of Computer Vision*, vol. 71, no. 3, pp. 361–372, 2006.

[33] G. E. Christensen, "Consistent linear-elastic transformations for image matching," in *Proceedings of the Biennial International Conference on Information Processing in Medical Imaging*, Springer, Visegrad, Hungary, pp. 224–237, June 1999.

[34] T. Rohlfing, C. R. Maurer, D. A. Bluemke, and M. A. Jacobs, "Volume-preserving nonrigid registration of MR breast images using free-form deformation with an incompressibility constraint," *IEEE Transactions on Medical Imaging*, vol. 22, no. 6, pp. 730–741, 2003.

[35] M. Droske and M. Rumpf, "A variational approach to nonrigid morphological image registration," *SIAM Journal on Applied Mathematics*, vol. 64, no. 2, pp. 668–687, 2004.

[36] N. Debroux, S. Ozeré, and C. Le Guyader, "A non-local topology-preserving segmentation-guided registration model," *Journal of Mathematical Imaging and Vision*, vol. 59, no. 3, pp. 432–455, 2017.

[37] S. Wright and J. Nocedal, *Numerical Optimization*, Springer, New York, NY, USA, 1999.

[38] J. Zhang, K. Chen, and B. Yu, "A novel high-order functional based image registration model with inequality constraint," *Computers & Mathematics with Applications*, vol. 72, no. 12, pp. 2887–2899, 2016.

[39] A. Trouvé, "Diffeomorphisms groups and pattern matching in image analysis," *International Journal of Computer Vision*, vol. 28, no. 3, pp. 213–221, 1998.

[40] P. Dupuis, U. Grenander, and M. I. Miller, "Variational problems on flows of diffeomorphisms for image matching," *Quarterly of Applied Mathematics*, vol. 56, no. 3, pp. 587–600, 1998.

[41] E. Haber and J. Modersitzki, "Numerical methods for volume preserving image registration," *Inverse Problems*, vol. 20, no. 5, pp. 1621–1638, 2004.

[42] D. Zhang and K. Chen, "A novel diffeomorphic model for image registration and its algorithm," *Journal of Mathematical Imaging and Vision*, vol. 60, no. 8, pp. 1261–1283, 2018.

[43] T. Vercauteren, X. Pennec, A. Perchant, and N. Ayache, "Diffeomorphic demons: efficient non-parametric image registration," *NeuroImage*, vol. 45, no. 1, pp. S61–S72, 2009.

[44] L. T. Biegler, O. Ghattas, M. Heinkenschloss et al., *Real-Time PDE-Constrained Optimization*, SIAM Publications, Philadelphia, PA, USA, 2007.