A NOVEL DIFFEOMORPHIC MODEL FOR IMAGE REGISTRATION
AND ITS ALGORITHM

DAOPING ZHANG† AND KE CHEN†‡

Abstract. In this work, we investigate image registration by mapping one image to another in a variational framework and focus on both model robustness and solver efficiency. We first propose a new variational model with a special regularizer, based on the quasi-conformal theory, which can guarantee that the registration map is diffeomorphic. It is well known that when the deformation is large, many variational models including the popular diffusion model cannot ensure diffeomorphism. One common observation is that the fidelity error appears small while the obtained transform is incorrect by way of mesh folding. However direct reformulation from the Beltrami framework does not lead to effective models; our new regularizer is constructed based on this framework and added to the diffusion model to get a new model, which can achieve diffeomorphism. However the idea is applicable to a wide class of models. We then propose an iterative method to solve the resulting nonlinear optimization problem and prove the convergence of the method. Numerical experiments can demonstrate that the new model can not only get a diffeomorphic registration even when the deformation is large, but also possess the accuracy in comparing with the currently best models.

Key words. Image registration, diffeomorphic, Beltrami coefficient, optimization, Gauss-Newton scheme.

AMS subject classifications. 65D, 65M, 65K, 68U, 68W

1. Introduction. Image registration is to find a transformation to map the corresponding image data, which are taken at different times, from different sensors, or from different viewpoints, for the purpose of telling the difference or merging information. Nowadays, image registration is widely used in many areas, such as computer vision, biological imaging, remote sensing and medical imaging [6, 21, 26, 32, 36, 38, 40, 47, 57]. In reality, according to the specific application, image registration can be classified into two categories: mono-modal registration and multi-modal registration. For multi-modal registration, finding a suitable distance measure is the most essential step [22, 35, 36, 47, 57]. The idea of this paper will be applicable to multi-modal registration framework, but we focus on the mono-modal registration in this work.

In dealing with the mono-modal registration, there are many choices of a data fidelity term [33] and a common approach for computing this transformation is to use the sum of squared differences (SSD) to measure the difference between the reference image R and the deformed template image T [11]. However, minimization of SSD alone in image registration is an ill-posed problem in the sense of Hadamard since it may have many solutions. In order to overcome this difficulty, regularization is indispensable [38, 52]. However, the choice of the regularization term, which needs some prior information about physical properties and helps to avoid the local minima, depends on the specific application.

All registration models are nonlinear but they can be classified into two main categories according to the way deformation mapping is represented: linear registration and nonlinear registration. In linear registration, the deformation model is linear
and global, including rotation, translation, shearing and scaling [11, 38]. Although
the computation speed of a linear model is fast since it contains few variables, it is
commonly used as the pre-registration for starting a more sophisticated model. This
is mainly because linear models can not accommodate the local details (differences).
In contrast, nonlinear registration models inspired by physical processes of trans-
formations [47] such as the elastic model [5], fluid model [9], diffusion model [16],
TV (total variation) model [19], MTV (modified TV) model [12], linear curvature
model [17, 18], mean curvature model [14], Gaussian curvature model [27] and total
fractional-order variation model [56] are proposed to account for localised variation
in details, by allowing many degrees of freedom. The particular free-form deforma-
tion models based on B-splines lying between the above two types possess simplicity,
smoothness, efficiency and ability to describe local deformation with few degrees of
freedom [44, 45, 47]. For relatively small deformation, all models can be effective,
but for large deformation, not all models are effective and in particular few models
can guarantee a one-to-one mapping unless one fine tunes the coupling parameters
to reduce the deformation magnitude allowed (since the mapping quality is perfect if
deformation is zero) which in turn loses the ability of modeling large deformation.

Over the last decade, more and more researchers have focused on diffeomorphic
image registration where folding measured by the local invertibility quantity det(J_y)
is reduced or avoided. Here, y denotes the transformation in the registration model
and det(J_y) is the Jacobian determinant of y. Under desired assumptions, obtaining
a one-to-one mapping is a natural choice as reviewed in [47].

In 2004, Haber and Modersitzki [23] proposed an image registration model im-
posing volume preserving constraints, by ensuring det(J_y) is close to 1. Although
volume preservation is very important in some applications where some underly-
ing (e.g. anatomical) structure is known to be incompressible [47], it is not required or
reasonable in others. In a later work, the same authors [25] relaxed the constraint to
allow det(J_y) to lie in a specific interval. Yanovsky et al. [55] applied the symmetric
Kullback-Leibler distance to quantify det(J_y) to achieve a diffeomorphic mapping.
Burger et al. [7] designed a volume penalty term that ensured that shrinkage and
growth had the same cost in their variational functional. The constrained hierar-
chical parametric approach [41] ensures that the mapping is globally one-to-one and
thus preserves topology in the deformed image. Sdika [46] introduced a regularizer to
penalize the non-invertible transformation. In [51], Vercauteren et al. proposed an ef-
ficient non-parametric diffeomorphic image registration algorithm based on Thirion’s
demons algorithm [49]. In addition, a framework called Large Deformation Diffeo-
metric Mapping (LDDMM) can generate the diffeomorphic transformation for
image registration [37, 3, 15, 50]. An entirely different framework proposed by Lam
and Lui [30] obtains diffeomorphic registrations by constraining Beltrami coefficients
of a quasi-conformal map \( f = y_1(x) + i y_2(x) \), instead of controlling the map \( y(x) \)
directly.

In this paper, we aim to reformulate the Lam and Lui Beltrami measure as a
direct regularizer for controlling \( \det(J_y) \) and to assess the effectiveness of the resulting
variational models; though the idea applies to any commonly used models, we apply
it to the diffusion model as one simple example. Our contributions are two-fold:

- We propose a new Beltrami coefficient based regularizer that is explicitly
  expressed in terms of \( \det(J_y) \). This establishes a link between the Beltrami
  coefficient of the transformation and the quantity \( \det(J_y) \).

- An effective, iterative scheme is presented and numerical experimental results
  show that the new registration model has a good performance and produces
a diffeomorphic mapping while remaining competitive to the state-of-the-art models from non-Beltrami frameworks.

We remark that several interesting works that are concerned with reversible transformations (such as [8, 54]) may also benefit from this study.

The rest of the paper is organized as follows. Section 2 briefly reviews the basic mathematical formulation of image registration modeling, several typical regularization terms and how to get a diffeomorphic transformation for image registration. In Section 3, we propose a new regularizer and a new registration model. The effective discretization and numerical scheme are discussed in Section 4. Numerical experiment results are shown in Section 5, and finally a summary is concluded in Section 6.

2. Preliminaries, Regularization and Diffeomorphic Transformation. In general, image registration aims to compare, in space $\mathbb{R}^d$, two or more images or image sequences in a video. In this work, we consider the case of a pair of images $T, R : \Omega \subset \mathbb{R}^d \rightarrow \mathbb{R}$ and $d = 2$. Here by convention, $R$ is the Reference image and $T$ is the (moving) Template image.

The aim of image registration is to find a transformation $y(x)$ such that

$$ T \circ y(x) = T(y(x)) \approx R, $$

where $x = (x_1, x_2)$ and $y(x) = (y_1(x), y_2(x))$. That is, the transformation $y(x)$ moves $T$ to match $R$. If we define $y(x) = x + u(x)$, then $u(x) = (u_1(x), u_2(x))$ indicates how much $T$ moves i.e. $u(x)$ is the displacement. Thus, the determination of the transformation $y(x)$ is equivalent to the determination of the displacement field $u(x)$.

### 2.1. Data fidelity.

One way to ensure that $T(y)$ can approximate $R$ is to minimize the difference $T(y) - R$. A commonly used difference measure is the sum of squared differences (SSD) defined by

$$ D[y] = \frac{1}{2} \int_{\Omega} (T(y) - R)^2 \, dx = \frac{1}{2} \| T(y) - R \|^2 = \frac{1}{2} \| T(x + u) - R \|^2 = D[u] $$

where $\| \cdot \|^2$ denotes the squared $L_2$-norm. Of course, there are some other typical distance measures, including normalized cross correlation [38], mutual information [35, 38], normalized gradient fields [24, 39] and mass-preserving measure [7].

### 2.2. Regularization.

Minimizing any of the above mentioned measures is inefficient to obtain a unique transformation $y$ for image registration, because $\min D[y]$ is ill-posed [38, 39]. In order to overcome this problem, regularization is necessary. Combining distance measure and regularization gives the variational model for image registration:

$$ \min_u J(u) = D[u] + \alpha S[u], $$

where $D[u]$ is the distance measure from (1), $S[u]$ is the regularizer to be discussed and $\alpha$ is a positive parameter to balance these two terms.

There exist many regularizers and we can classify them into three categories:

- First order regularizers involving $|\nabla u|$ or $|\nabla \cdot u|$. The diffusion regularizer [16] and the TV regularizer [19] are well-known first order regularizers. The former one aims to control smoothness of the displacement and the latter one can preserve the discontinuity.
- Fractional order regularizer $\nabla^\alpha u$ with $\alpha \in (1, 2)$. In [56], a fractional order regularizer is used for image registration. Because the fractional order...
regularizer is a global regularizer, its implementation must explore the structured Toeplitz matrices. This regularizer can not only produce accurate and smooth solutions but also allow for a large rigid alignment \cite{56}.

- Second order regularizers involving $\nabla^2 u$ or $\nabla \cdot (\nabla u/|\nabla u|)$. These include the linear curvature regularizer \cite{17, 18}, mean curvature regularizer \cite{14} and Gaussian curvature regularizer \cite{27}.

The first two categories of models require an affine linear transformation in an initial pre-registration step while the latter category does not need a linear transformation in pre-registration.

Differing from the above three categories, an important class of fluid like models based on partial differential equations were developed to capture large deformations. Christensen et al. \cite{10} proposed an effective viscous fluid model characterized by a spatial smoothing of the velocity field. For the viscous fluid model, the deformation is governed by the Navier-Stokes equation:

\begin{equation}
\eta \nabla^2 v + (\eta + \lambda) \nabla (\nabla \cdot v) + F = 0, \quad v = \partial_t u + v \cdot \nabla u.
\end{equation}

Here, $\eta$ and $\lambda$ are the viscosity coefficients, the term $\nabla^2 v$ constrains the velocity field to vary smoothly, the term $\nabla (\nabla \cdot v)$ allows structures in the template to change in mass and $F$ is the nonlinear deformation force field, which can be defined by $(T(x+u) - R) \nabla T$. The velocity field $v$ is initialized as $0$ in implementation. In \cite{10}, the condition $|\det(J_y)| \geq 0.5$ is checked at each iteration and if not satisfied, restarting the numerical solver is initiated so that a diffeomorphic transform is obtained; see also \cite{38}. Further in \cite{55}, the model is enhanced by incorporating a volume preservation idea relating to minimizing $|\det(J_y) - 1|$ again to ensure diffeomorphism without restarting.

Next, we review the \textbf{Diffusion} model \cite{16}

\begin{equation}
\min_u J(u) = D[u] + \alpha S_2[u] = \frac{1}{2} \int_\Omega (T(x+u) - R)^2 dx + \frac{\alpha}{2} \int_\Omega \sum_{\ell=1}^2 |\nabla u_\ell|^2 dx.
\end{equation}

It leads to the Euler-Lagrange equation:

\begin{align*}
(T(x+u) - R) \nabla u T(x+u) - \alpha \Delta u &= 0 \text{ i.e. } (T(x+u) - R) \partial_{u_\ell} T(x+u) - \alpha \Delta u_1 = 0, \\
(T(x+u) - R) \partial_{u_2} T(x+u) - \alpha \Delta u_2 &= 0,
\end{align*}

subject to $(\nabla u_\ell, n) = 0$ on $\partial\Omega$ and $\ell = 1, 2$. Particularly, there exits a fast implementation based on the so-called additive operator splitting (AOS) scheme \cite{38, 53}. In \cite{13}, a fast solver was developed for this model.

However, as with other models reviewed in the three categories, the obtained solution $u$ or $y$ is mathematically correct but often incorrect physically. This is due to no guarantee of mesh non-folding which is measured by $\det(J_y) > 0$ i.e. a positive determinant of the local Jacobian matrix $J_y$ of the transform $y$.

\textbf{2.3. Models of diffeomorphic transformation.} To achieve $\det(J_y) > 0$, one can find several recent works that impose this constraint in some direct ways. We review a few of such models before we present our new constraint. In the form of (4), the idea is to choose $S_1[y]$ in the following (note $y = x + u$)

\begin{equation}
\min_u J(u) = D[u] + \alpha S_2[u] + \beta S_1[y].
\end{equation}
Volume control. In 2004, Haber and Modersitzki \[23\] used volume preserving constraint (area in 2D) for image registration, namely $\det(J_y) = 1$. As a consequence, we can ensure that the transformation is diffeomorphic. However, volume preservation is not desirable when the anatomical structure is compressible in medical imaging.

Slack constraint. Improving on \[25\], the constraint $\det(J_y) = 1$ is relaxed and a slack constraint is proposed

$$M_a \leq \det(J_y) \leq M_b,$$

where a positive interval $[M_a, M_b]$ is provided by the user as prior information in the specific application e.g. $[0, 1]$.

Unbiased transform. In \[55\], according to the information theory, $\det(J_y)$ is controlled by the symmetric Kullback-Leibler distance

$$\int_{\Omega} |\det(J_y) - 1| \log(|\det(J_y)|) \, dx.$$

It can help to get an unbiased diffeomorphic transformation. This idea was tested with the fluid regularizer (first order).

Balance of shrinkage and growth. Geometrically $\det(J_y) = 1$ implies volume preservation. Similarly $\det(J_y) < 1$ implies shrinkage while $\det(J_y) > 1$ implies growth. A function that treats the cases of shrinkage and growth identically is $\phi(x) = ((x - 1)^2/x)^2$ since $\phi(1/x) = \phi(x)$. A volume penalty

$$\int_{\Omega} \left( \frac{(\det(J_y) - 1)^2}{\det(J_y)} \right)^2 \, dx$$

is used in the hyperelastic model \[7\], which ensures that shrinkage and growth have the same price.

LDDMM Framework. In LDDMM framework, the deformation is modeled by considering its velocity over time according to the transport equation. We can write its variational formulation as follows:

$$\min_{T,v} D(T(\cdot, 1), R) + \alpha S(v)$$

s.t. $\partial_t T(x, t) + v(x, t) \cdot \nabla T(x, t) = 0$ and $T(x, 0) = T$,

where $v : \Omega \times [0, 1] \to \mathbb{R}^2$ is the velocity and $T : \Omega \times [0, 1] \to \mathbb{R}$ is a series of images. For more details, please see \[37, 3, 15, 47, 50\]

Beltrami indirect control. In 2014, Lam and Lui \[30\] presented a novel approach in a Beltrami framework to obtain diffeomorphic registrations with large deformations using landmark and intensity information via quasi-conformal maps. Before introducing this model, we first describe some basic theories about quasi-conformal map and Beltrami coefficient.

A complex map $z = x_1 + ix_2 \mapsto f(z) = y_1(x_1, x_2) + iy_2(x_1, x_2)$ from a domain in $\mathbb{C}$ onto another domain is quasi-conformal if it has continuous partial derivatives and satisfies the following Beltrami equation:

$$\frac{\partial f}{\partial z} = \mu(f) \frac{\partial f}{\partial \bar{z}},$$

This manuscript is for review purposes only.
for some complex-valued Lebesgue measurable $\mu$ [4] satisfying $\|\mu\|_\infty < 1$. Here $\mu = \mu(y) = f_z/f_x$ is called the Beltrami coefficient explicitly computable from $y$ since

$$
\begin{align*}
\left\{ \begin{array}{ll}
\frac{\partial f}{\partial \bar{z}} & = 1 \equiv \frac{\partial f}{\partial x} - \frac{1}{2} \frac{\partial f}{\partial x_2} \equiv \frac{\partial f}{\partial x_1} + \frac{1}{2} \frac{\partial f}{\partial x_2} = (y_1)x_1 + (y_2)x_2 + 1(y_2)x_1 - (y_1)x_2 \\
\frac{\partial f}{\partial \bar{z}} & = 1 \equiv \frac{\partial f}{\partial x} - \frac{1}{2} \frac{\partial f}{\partial x_2} \equiv \frac{\partial f}{\partial x_1} + \frac{1}{2} \frac{\partial f}{\partial x_2} = (y_1)x_1 - (y_2)x_2 + 1(y_2)x_1 + (y_1)x_2 ,
\end{array} \right.
\end{align*}
$$

where $(y_1)x_1 = \partial y_1/\partial x_1$. Conversely $y = y^\mu$ can be computed for a given $\mu$ through solving $\mu(y) = \mu$.

A quasi-conformal map is a homeomorphism (i.e. one-to-one) and its first-order approximation takes small circles to small ellipses of bounded eccentricity [20]. As a special case, $\mu = 0$ means that the map $f$ is holomorphic and conformal, characterized by $f_z = 0$ or $y_1, y_2$ satisfying the Cauchy-Riemann equations $(y_1)x_1 = (y_2)x_2$, $(y_1)x_2 = -(y_2)x_1$.

Thus in the context of image registration, enforcing $\|\mu\|_\infty < 1$ provides the control for the transform $f$ and ensures homeomorphism. The quasi-conformal hybrid registration model (QCHR) in [30] is

$$\min_y \int_\Omega |\nabla \mu|^2 + \alpha \int_\Omega |\mu|^p + \beta \int_\Omega (T(y) - R)^2$$

subject to $y = (y_1, y_2)$ satisfying

1. $\mu = \mu(y)$;
2. $y(p_j) = q_j$ for $1 \leq j \leq m$ (Landmark constraints);
3. $\|\mu(y)\|_\infty < 1$ (bijectivity),

which indirectly controls $\det(J_y)$ via Beltrami coefficient, where $\mu(y)$ is the Beltrami coefficient of the transformation $y$. The above model is solved by a penalty splitting method. It minimizes the following functional:

$$\int_\Omega |\nabla \nu|^2 + \alpha \int_\Omega |\nu|^p + \sigma \int_\Omega |\nu - \mu|^2 + \beta \int_\Omega (T(y^\mu) - R)^2$$

subject to the constraints that $\|\nu\|_\infty < 1$ and $y^\mu$ be the quasi-conformal map with Beltrami coefficient $\mu$ satisfying $y^\mu(p_j) = q_j$ for $1 \leq j \leq m$. Then in each iteration, it needs to solve the following two subproblems alternately:

$$\mu_{n+1} = \arg \min \sigma \int_\Omega |\mu - \nu_n|^2 + \beta \int_\Omega (T(y^\mu) - R)^2
\text{s.t. } y^\mu(p_j) = q_j \text{ for } 1 \leq j \leq m$$

and

$$\nu_{n+1} = \arg \min \int_\Omega |\nabla \nu|^2 + \alpha \int_\Omega |\nu|^p + \sigma \int_\Omega |\nu - \mu_{n+1}|^2 .$$

In addition, it also solves the equation $\mu(y) = \mu$ by the linear Beltrami solver (LBS) [34] to find $y$ and ensures that $y$ matches the landmark constraints.

Thus, instead of controlling the Jacobian determinant of the transformation directly, controlling Beltrami coefficient is also a good alternative providing the same but indirect control. However, since their algorithm [30] has to deal with two main unknowns (the transformation $y$ and its Beltrami coefficient $\mu$) and one auxiliary unknown (the coefficient $\nu$) in a non-convex formulation, the increased cost, practical
implementation and convergence are real issues; for challenging problems, one cannot observe convergence and therefore the full capability of the model is not realized.

We are motivated to reduce the unknowns and simplify their algorithm. Our solution is to reformulate the problem in the space of the primary variable \( y \) or \( u \), not in the transformed space of variables \( \mu, \nu \). We make use of the explicit formula of \( \mu = \mu(y) \). Working with primal mapping \( y \) enables us to introduce the advantages of minimizing a Beltrami coefficient to the above reviewed variational framework (2), effectively unifying the two frameworks.

Hence, we propose a new regularizer based Beltrami coefficient and, in the numerical part, we can find that it is easy to be implemented. Moreover the reformulated control regularizer can potentially be applied to a large class of variational models.

3. The proposed image registration model. In this section, we aim to present a new regularizer based on Beltrami coefficient, which can help to get a diffeomorphic transformation. Then combining the new regularizer with the diffusion model, we present a novel model. Of course, combining with other models may be studied as well since the idea is the same.

For \( f(z) = y_1(x_1, x_2) + y_2(x_1, x_2) \), according to the Beltrami equation (7) and the definitions (8), we have

\[
\mu(f) = \frac{\partial f}{\partial z}/\frac{\partial f}{\partial \bar{z}} = \frac{((y_1)_x_1 - (y_2)_x_2) + i((y_2)_x_1 + (y_1)_x_2)}{((y_1)_x_1 + (y_2)_x_2) + i((y_2)_x_1 - (y_1)_x_2)};
\]

\[
|\mu(f)|^2 = \frac{((y_1)_x_1 - (y_2)_x_2)^2 + ((y_2)_x_1 + (y_1)_x_2)^2}{((y_1)_x_1 + (y_2)_x_2)^2 + ((y_2)_x_1 - (y_1)_x_2)^2} = \frac{\|J_f||^2 - 2\det(J_f)}{\|J_f||^2 + 2\det(J_f)}.
\]

Note \((y_1)_x_1(y_2)_x_2 - (y_2)_x_1(y_1)_x_2 = \det(J_f)\). So \(\det(J_f)\) can be represented by the Beltrami coefficient \(\mu(f)\)

\[
\det(J_f) = |f_z|^2(1 - |\mu(f)|^2)
\]

Clearly \(\det(\nabla f) > 0\) if \(|\mu(f)| < 1\), and by the inverse function theorem, the map \( f \) is locally bijective. We conclude that \( f \) is diffeomorphism if we assume that \( \Omega \) is bounded, simply connected. For more details about quasi-conformal theory, the readers can refer to [1, 20, 31].

3.1. New regularizer. Our new regularizer based on \(|\mu(f)| < 1\) to control the transformation to get a diffeomorphic mapping is

\[
S_1[y] = \int_\Omega \phi(|\mu|^2)dx, \quad |\mu|^2 = \frac{\|J_y||^2 - 2\det(J_y)}{\|J_y||^2 + 2\det(J_y)}
\]

which clearly involves the Jacobian determinant \(\det(J_y)\) in a non-trivial way and we explore the choices of \(\phi\) below.

Remark. Our new regularizer has two advantages: one is that the obtained transformation \( y \) do not need to possess \(\det(J_y) \to 1\); the other one is that we only compute the transformation and do not need to compute its Beltrami coefficient and introduce another auxiliary unknown as [30]. In addition, from the numerical experiments, we can see that our new regularizer is easy to implement and obtains accurate and diffeomorphic transformations.
3.2. The proposed model. The above regularizer (16) providing a constraint on \( y \) is ready to be combined with an existing model. In the framework (5), using (16), the first version of our new model takes the form

\[
\min_{\mathbf{u}} \frac{1}{2} |T(\mathbf{y}) - \mathcal{R}|^2 + \frac{\alpha}{2} |\nabla \mathbf{u}|^2 + \beta \int_{\Omega} \phi(|\mu|^2) \mathrm{d}x
\]

where \( \mathbf{u} = y(x) - x = (y_1(x), y_2(x)) - x \) is the deformation field, \( |\nabla \mathbf{u}|^2 = |\nabla u_1|^2 + |\nabla u_2|^2 \) and \( \mu = \mu(y) \). To promote \( |\mu(f)| < 1 \), our first and simple choice is \( \phi(v) = \phi_1(v) = \frac{1}{(v-1)^2}, \) which forces (17) and \( \phi(v) \) to reduce \( v \), at the initial guess \( v = 0 \) when \( \mathbf{u} = 0 \), since \( \phi_1(v) \to \infty \) when \( v \to 1 \).

Remark. From (9) and (17), we see that the QCHR model focuses on obtaining a smooth Beltrami coefficient and our model focuses on the diffeomorphic transformation itself. There are major differences between the regularizer in QCHR model and our new regularizer: the former is characterized by the Beltrami coefficient \( \mu \) directly and gradient of this Beltrami coefficient \( \mu \), while the latter is characterized by the Beltrami coefficient indirectly in terms of the transformation \( y \) and the gradient of \( \mathbf{u} \). Since \( y = x + \mathbf{u} \) is our desired transformation, our direct regularizers such as \( |\nabla \mathbf{u}|^2 \) make more sense than indirect regularizers such as \( |\nabla \mu|^2 \).

However as long as \( |\mu(f)| < 1 \), we would not give a preference to forcing \( |\mu(f)| \to 0 \).

To put some control on bias, similarly to [7], we are led to 2 more choices of a less unbiased function to modify \( S_1[y] \)

- \( \phi(v) = \phi_2(v) = \frac{v}{(v-1)^2} \): balance \( |\mu(f)| \) between 0 and 1 as \( \phi_2(v) = \phi_2(1/v) \);
- \( \phi(v) = \phi_3(v) = \frac{1}{(v-1)^2} \): encourage \( |\mu(f)| \to 0 \) and \( |\mu(f)| \neq 1 \);

Below, we list first order derivatives and second order derivatives for the above different \( \phi(v) \):

- \( \phi'_1(v) = \frac{2}{(v-1)^2} \) and \( \phi''_1(v) = \frac{6}{(v-1)^4} \);
- \( \phi'_2(v) = -\frac{v+1}{(v-1)^2} \) and \( \phi''_2(v) = -\frac{2v+4}{(v-1)^4} \);
- \( \phi'_3(v) = -\frac{2v}{(v-1)^2} \) and \( \phi''_3(v) = \frac{4v+2}{(v-1)^4} \)

which will be used in subsequent solutions. With a general \( \phi(v) \), the second version of our proposed model takes the form:

\[
\min_{\mathbf{u}} \frac{1}{2} \int_{\Omega} (T(x+\mathbf{u}) - \mathcal{R})^2 \mathrm{d}x + \frac{\alpha}{2} \int_{\Omega} \sum_{\ell=1}^2 |\nabla u_\ell|^2 \mathrm{d}x + \beta \int_{\Omega} \phi(|\mu|^2) \mathrm{d}x,
\]

where \( |\mu|^2 = \frac{(\partial_{u_1} u_1 - \partial_{x_2} u_2)^2 + (\partial_{u_2} u_2 + \partial_{x_1} u_1)^2 + (\partial_{x_1} u_1 - \partial_{x_2} u_2)^2}{(\partial_{x_1} u_1 + \partial_{x_2} u_2 + \partial_{x_2} u_2 + \partial_{x_1} u_1)^2} \) is written in component form ready for discretization, using \( y_1 = x_1 + u_1(x_1, x_2), \ y_2 = x_2 + u_2(x_1, x_2) \), and \( \partial_{x_1} u_1 = \partial u_1/\partial x_1 \).

Remark. For the existence or uniqueness of a solution of (18), this is out of the scope of the present work and will be considered in our forthcoming work.

4. The numerical algorithm. In this section, we will present a numerical algorithm to solve model (18). We choose the discretize - optimize approach. Directly discretizing this variational model gives rise to a finite dimensional optimization problem. Then we use optimization methods to solve this resulting problem.

4.1. Discretization. We use finite differences to discretize model (18) on a unit square domain \( \Omega = [0,1]^2 \). In implementation, we employ the nodal grid and define a spatial partition \( \Omega_h = \{ \mathbf{x}^{i,j} \in \Omega \mid \mathbf{x}^{i,j} = (x_1^i, x_2^j) = (ih, jh), 0 \leq i \leq n, 0 \leq j \leq n \}, \)
where $h = \frac{1}{n}$ and the discrete domain consists of $n^2$ cells of size $h \times h$. We discretize the displacement field $u$ on the nodal grid, namely $u^{i,j} = (u_{1}^{i,j}, u_{2}^{i,j}) = (u_{1}(x_{1}^{i}, x_{2}^{j}), u_{2}(x_{1}^{i}, x_{2}^{j}))$. For ease presentation, according to the lexicographical ordering, we reshape

$$X = (x_{1}^{0}, ..., x_{1}^{n}, x_{2}^{0}, ..., x_{2}^{n})^T \in \mathbb{R}^{2(n+1)^2 \times 1},$$

and

$$U = (u_{1}^{0,0}, ..., u_{1}^{n,n}, ..., u_{2}^{0,0}, ..., u_{2}^{n,n})^T \in \mathbb{R}^{2(n+1)^2 \times 1}.$$

### 4.1.1. Discretization of Term 1 in (18).

According to the cell-centred partition in Figure 1(a) and mid-point rule, we get

$$D[u] := \frac{1}{2} \int_{\Omega} (T(x + u(x)) - R(x))^2 dx$$

$$= \frac{h^2}{2} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (T(x^{i+\frac{1}{2}, j+\frac{1}{2}} + u(x^{i+\frac{1}{2}, j+\frac{1}{2}})) - R(x^{i+\frac{1}{2}, j+\frac{1}{2}}))^2.$$

Fig. 1. Partition of domain $\Omega = \cup_{i,j} \Omega_{i,j}$. Note that solutions $u_{1}$ and $u_{2}$ are defined at nodes.

Set $\tilde{R} = R(PX) \in \mathbb{R}^{n^2 \times 1}$ as the discretized reference image and $\tilde{T}(PX + PU) \in \mathbb{R}^{n^2 \times 1}$ as the discretized deformed template image, where $P \in \mathbb{R}^{2n^2 \times (n+1)^2}$ is an averaging matrix for the transfer from the nodal grid representation of $U$ to the cell centered positions.

Consequently, for SSD, we obtain the following discretization:

$$D[u] \approx \frac{h^2}{2} (\tilde{T}(PX + PU) - \tilde{R})^T (\tilde{T}(PX + PU) - \tilde{R}).$$

This manuscript is for review purposes only.
4.1.2. Discretization of Term 2 in (18). For the diffusion regularizer,

\[ S_{\text{diff}}[\mathbf{u}] := \alpha \frac{1}{2} \int_{\Omega} \sum_{\ell=1}^{2} |\nabla u_{\ell}|^{2} d\mathbf{x}, \]

according to the the partition in Figure 1(b) and mid-point rule, we have

\[ \int_{\Omega_{i,j}^{t+1}} |\partial_{x_{1}} u_{\ell}|^{2} d\mathbf{x} \approx h^{2} (\partial_{x_{1}}^{t+\frac{1}{2},j} u_{\ell})^{2} \quad 1 \leq j \leq n - 1, \]

or at the boundary half-boxes

\[ \int_{\Omega_{1,j}^{t+1}} |\partial_{x_{1}} u_{\ell}|^{2} d\mathbf{x} \approx h^{2} (\partial_{x_{1}}^{t+\frac{1}{2},j} u_{\ell})^{2} \quad j = 0, n. \]

And for \( \int_{\Omega_{1,j}^{t+1}} |\partial_{x_{2}} u_{\ell}|^{2} d\mathbf{x}, \ell = 1, 2, \) we have similar results.

As designed, we use compact (short) difference schemes to compute the \( \partial_{x_{1}} u_{\ell} \) and \( \partial_{x_{2}} u_{\ell}, \ell = 1, 2: \)

\[ \partial_{x_{1}}^{t+\frac{1}{2},j} u_{\ell} \approx \frac{u_{\ell}^{i+1,j} - u_{\ell}^{i,j}}{h}, \quad \partial_{x_{2}}^{t+\frac{1}{2},j} u_{\ell} \approx \frac{u_{\ell}^{i,j+1} - u_{\ell}^{i,j}}{h}. \]

Then (21) can be rewritten in the following formulation:

\[ S_{\text{diff}}[\mathbf{u}] \approx \alpha h^{2} U^{T} A^{T} G A U. \]

See Appendix A for details on \( A \) and \( G \).

Remark. Note that here the matrix \( A \) is the discretized gradient matrix. So \( A^{T} G A \) is the discretized Laplace matrix.

4.1.3. Discretization of Term 3 in (18). For simplicity, denote \( |\mu(\mathbf{y})| = |\mu(\mathbf{x} + \mathbf{u})| \) by \( |\mu(\mathbf{u})| \). From (18), note that \( \phi(|\mu(\mathbf{u})|^{2}) \) involves only first order derivatives and all \( \mathbf{u}^{(i)} \) are available at vertex pixels. Thus it is convenient first to obtain

\[ \text{Fig. 2. Partition of a cell, nodal point } \square \text{ and center point } \circ. \text{ \( \Delta V_{1}V_{2}V_{5} \) is } \Omega_{i,j,k}. \]
approximations at all cell centres (e.g. at $V_5$ in Figure 2) and second to use local linear elements to facilitate first order derivatives. We shall divide each cell (Figure 2) into 4 triangles. In each triangle, we construct two linear interpolation functions to approximate the $u_1$ and $u_2$. Consequently, all partial derivatives are locally constants or $\phi((\mu(u))^2)$ is constant in each triangle.

According to the partition in Figure 2, we get

$$S_{\text{Beltrami}}[\mu] := \beta \int_{\Omega} \phi((\mu(u))^2)dx = \beta \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{4} \int_{\Omega_{i,j,k}} \phi((\mu(u))^2)dx.$$ (26)

Set $L_{i,j,k}(x) = (L_1^{i,j,k}(x), L_2^{i,j,k}(x)) = (a_1^{i,j,k}x_1 + a_2^{i,j,k}x_2 + a_3^{i,j,k}x_3, a_4^{i,j,k}x_1 + a_5^{i,j,k}x_2 + a_6^{i,j,k})$, which is the linear interpolation for $u$ in the $\Omega_{i,j,k}$. Note that $\partial_{x_1} L_1^{i,j,k} = a_1^{i,j,k}$, $\partial_{x_2} L_1^{i,j,k} = a_2^{i,j,k}$, $\partial_{x_1} L_2^{i,j,k} = a_4^{i,j,k}$ and $\partial_{x_2} L_2^{i,j,k} = a_5^{i,j,k}$. According to (18), the discretization of Beltrami regularizer can be written into following:

$$S_{\text{Beltrami}}[u] \approx \frac{\beta h^2}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{4} \phi \left( \frac{(a_1^{i,j,k} - a_5^{i,j,k})^2 + (a_2^{i,j,k} + a_4^{i,j,k})^2}{(a_1^{i,j,k} + a_5^{i,j,k} + 2) + (a_2^{i,j,k} - a_4^{i,j,k})^2} \right).$$ (27)

To simplify (27), define 3 vectors $\overline{r}(U), \overline{r}^1(U), \overline{r}^2(U) \in \mathbb{R}^{4n^2}$ by $\overline{r}(U)_\ell = \overline{r}^1(U)_{\ell} \overline{r}^2(U)_{\ell}$, $\overline{r}^1(U) = (a_1^{i,j,k} - a_5^{i,j,k})^2 + (a_2^{i,j,k} + a_4^{i,j,k})^2$, $\overline{r}^2(U)_{\ell} = 1/[(a_1^{i,j,k} + a_5^{i,j,k} + 2) + (a_2^{i,j,k} - a_4^{i,j,k})^2]$ where $\ell = (k-1)n^2 + (j-1)n + i \in [1, 4n^2]$. Hence, (27) becomes

$$S_{\text{Beltrami}}[u] \approx \frac{\beta h^2}{4} \overline{r}(\overline{r}(U)) e^T.$$ (28)

where $\overline{r}(\overline{r}(U)) = (\phi(\overline{r}(U)_1), \ldots, \phi(\overline{r}(U)_{4n^2}))$ denotes the pixel-wise discretization of $u_1, u_2$ at all cell centers, and $e = (1, \ldots, 1) \in \mathbb{R}^{4n^2}$. Here, $\overline{r}(U)$ is the square of the discretized Beltrami coefficient; we rewrite it in a compact form in Appendix B.

Finally, combining the above three parts (20), (25) and (28), we get the discretization formulation for model (18):

$$\min_U J(U) := \frac{h^2}{2} (\overline{T}_U (P_X + P_U) - \overline{R})^T (\overline{T}_U (P_X + P_U) - \overline{R}) + \frac{\alpha h^2}{2} U^T A^T G A U + \frac{\beta h^2}{4} \phi(\overline{r}(U)) e^T.$$ (29)

Remark. According to the definition of $\phi$ and $\overline{r}(U)_\ell \geq 0$, each component of $\phi(\overline{r}(U))$ is non-negative and differentiable.

4.2. Optimization method for the discretized problem (29). In the numerical implementation, we choose line search method to solve the resulting unconstrained optimization problem (29). In order to guarantee the search direction is a descent direction, we employ the Gauss-Newton direction as the standard direction involving non-definite Hessians does not generate a descent direction. Otherwise, using a Gauss-Newton approach presents two advantages: one is that we do not need to compute the second order term and it can save computation time; the other one is that this Gauss-Newton matrix is more important than the second term, either because of small second order derivatives or because of small residuals [42].

Let $J(U) : \mathbb{R}^{2(n+1)^2} \to \mathbb{R}$ be twice continuously differentiable, $U^k \in \mathbb{R}^{2(n+1)^2}$ and the approximated Hessian $H(U^k)$ positive definite. We model $J$ at the current point...
by the quadratic approximation $q^k(s)$,

$$J(U^k + s) \approx q^k(s) = J(U^k) + d_J(U^k)^T s + \frac{1}{2} s^T H(U^k)^T s,$$

where $s = U - U^k$ and $d_J(U^k) = \nabla J(U^k)$. Minimizing $q^k(s)$ yields

$$U^{k+1} = U^k - [H(U^k)]^{-1} d_J(U^k).$$

In order to guarantee the global convergence of the Gauss-Newton method, we employ the line search and its iteration is as follows:

$$U^{k+1} = U^k - \theta_k [H(U^k)]^{-1} d_J(U^k).$$

where $\theta_k$ is a step length.

Next, we will investigate the details about the approximated Hessian $H(U^k)$, step length $\theta_k$, stopping criteria and multilevel strategy.

### 4.2.1. Approximated Hessian $H$

We consider each of the three terms in $J(U)$ from (29) separately.

Firstly, we consider the discretized SSD

$$\frac{h^2}{2} (\tilde{T}(PX + PU) - \tilde{R})^T (\tilde{T}(PX + PU) - \tilde{R}).$$

Its gradient and Hessian are respectively

$$d_1 = h^2 P^T \tilde{T}^T (\tilde{T}^T (\tilde{U}) - \tilde{R}) \in \mathbb{R}^{2(n+1)^2 \times 1},$$

$$H_1 = h^2 P^T (\tilde{T}^T \tilde{T}) + \sum_{\ell=1}^{n^2} (\tilde{T}^T (\tilde{U}) - \tilde{R})_{\ell} \nabla^2 (\tilde{T}^T (\tilde{U}) - \tilde{R})_{\ell} P$$

where $\tilde{U} = PX + PU$ and $\tilde{T}_U = \frac{\partial \tilde{T}(\tilde{U})}{\partial \tilde{U}}$ as the Jacobian of $\tilde{T}$ with respect to $\tilde{U}$.

For $H_1$, we cannot ensure that it is positive semi-definite. If it is not positive definite, we may not get a descent direction. So we omit the second order term of $H_1$ to obtain the approximated Hessian of (33):

$$\hat{H}_1 = h^2 P^T (\tilde{T}^T \tilde{T}_U) P.$$

**Remark.** Evaluation of the deformed template image $T$ must involve interpolation because $\tilde{U}$ do not in general correspond to pixel points; in our implementation, as with [39], we use B-splines interpolation to get $\tilde{T}(\tilde{U})$.

Secondly, for the discretized diffusion regularizer $\frac{\alpha h^2}{2} U^T A^T G A U$, its gradient and Hessian are the following:

$$d_2 = \alpha h^2 A^T G A U \in \mathbb{R}^{2(n+1)^2 \times 1},$$

$$H_2 = \alpha h^2 A^T G A \in \mathbb{R}^{2(n+1)^2 \times 2(n+1)^2}.$$

Since $H_2$ is positive definite when $U$ is applied with Dirichlet boundary conditions, we do not approximate it.

Finally, for the discretized Beltrami term

$$\frac{\beta h^2}{4} \phi(\bar{r}(U)) e^T,$$
the gradient and the Hessian are as follows:

\[
\begin{align*}
\mathbf{d}_3 &= \frac{\beta h^2}{4} \mathbf{d}\mathbf{r}^T \mathbf{d}\phi(\mathbf{r}) \in \mathbb{R}^{2(n+1)^2 \times 1}, \\
H_3 &= \frac{\beta h^2}{4} (\mathbf{d}\mathbf{r}^T \mathbf{d}\phi(\mathbf{r}) \mathbf{d}\mathbf{r} + \sum_{i=1}^{4n^2} [\mathbf{d}\phi(\mathbf{r})]_{i} \nabla^2 \mathbf{r}_i) \in \mathbb{R}^{2(n+1)^2 \times 2(n+1)^2}
\end{align*}
\]

where \( \mathbf{d}\phi(\mathbf{r}) = (\phi'(\mathbf{r}_1), ..., \phi'(\mathbf{r}_{4n^2}))^T \) is the vector of derivatives of \( \phi \) at all cell centers,

\[
\begin{align*}
\mathbf{d}\mathbf{r} &= \text{diag}(\mathbf{r}^1) \mathbf{d}\mathbf{r}^2 + \text{diag}(\mathbf{r}^2) \mathbf{d}\mathbf{r}^3,
\mathbf{d}\mathbf{r}^1 &= 2 \text{diag}(A_1 U) A_1 + 2 \text{diag}(A_2 U) A_2,
\mathbf{d}\mathbf{r}^2 &= - \text{diag}(\mathbf{r}^2 \circ \mathbf{r}^2) [2 \text{diag}(A_3 U + 2) A_3 + 2 \text{diag}(A_4 U) A_4],
\mathbf{d}\mathbf{r}^3 &= \text{diag}(\mathbf{r}^3) \mathbf{d}\mathbf{r}^3,
\mathbf{d}\mathbf{r}^4 &= \text{diag}(\mathbf{r}^4) \mathbf{d}\mathbf{r}^4.
\end{align*}
\]

\( \odot \) denotes a Hadamard product, \( \mathbf{d}\mathbf{r}, \mathbf{d}\mathbf{r}^1, \mathbf{d}\mathbf{r}^2 \) are the Jacobian of \( \mathbf{r}, \mathbf{r}^1, \mathbf{r}^2 \) with respect to \( U \) respectively, \( [\mathbf{d}\phi(\mathbf{r})]_i \) is the \( i \)th component of \( \mathbf{d}\phi(\mathbf{r}) \) and \( \mathbf{d}^2 \phi(\mathbf{r}) \) is the Hessian of \( \phi \) with respect to \( \mathbf{r} \), which is a diagonal matrix whose \( i \)th diagonal element is \( \phi''(\mathbf{r}_i), 1 \leq i \leq 4n^2 \). Here \( \text{diag}(v) \) is a diagonal matrix with \( v \) on its main diagonal.

More details about \( \mathbf{r}^1, \mathbf{r}^2, A_1, A_2, A_3 \) and \( A_4 \) are shown in Appendix B and some illustration of our notation is given in Appendix C.

To extract a positive semi-definite part out of (38), we omit the second order term and obtain the approximated Hessian as

\[
H_3 = \frac{\beta h^2}{4} \mathbf{d}\mathbf{r}^T \mathbf{d}\phi(\mathbf{r}) \mathbf{d}\mathbf{r}.
\]

Therefore for functional \( J(U) \) in (29) with any choice of \( \phi \), we obtain its gradient

\[
d_J = d_1 + d_2 + d_3
\]

and approximated Hessian:

\[
H = \hat{H}_1 + \hat{H}_2 + \hat{H}_3.
\]

4.2.2. Search Direction. At each iteration, using (41) and (42), we need to solve the Gauss-Newton system to find the search direction of (29):

\[
H\delta U = -d_J,
\]

where \( \delta U \) is the search direction. In our implementation, we use MINRES with diagonal preconditioning to solve this linear system [2, 43].

4.2.3. Step Length. We use the standard Armijo strategy with backtracking to find a suitable step length \( \theta \). In the implementation, we also need to check that \( \mathbf{r}(U) \) (54) is smaller than 1. Recall that \( \mathbf{r}(U) \) is the norm square of the discretized Beltrami term. As a safe guard, we choose \( T_0 = 10^{-8} \) and \( \text{Tol} = 10^{-12} \) as the lower bound of the step length \( \theta \) and \( \theta \|\delta U\| \) [7, 28, 42, 48]. The algorithm is summarized in Algorithm 1.

4.2.4. Stopping Criteria. Here, we adopt the stopping criteria as in [39]:

\[
\begin{align*}
(1.a) \quad & \|J(U^{i+1}) - J(U^i)\| \leq \tau_J (1 + \|J(U^0)\|), \\
(1.b) \quad & \|y^{i+1} - y^i\| \leq \tau_W (1 + \|y^0\|), \\
(1.c) \quad & \|d_J\| \leq \tau_G (1 + \|J(U^0)\|), \\
(2) \quad & \|d_J\| \leq \epsilon, \\
(3) \quad & i \geq \text{MaxIter}.
\end{align*}
\]

This manuscript is for review purposes only.
Algorithm 1 Armijo Line Search: \((U, ID) \leftarrow ALS(U, \delta U)\)

Step 1: Initialisation. Set \(ID = 0\), \(\theta = 1\), \(Tol = 10^{-12}\), \(T0 = 10^{-8}\) and \(\eta = 10^{-4}\).

\[\text{Compute } J(U) \text{ and } d_J.\]

Step 2: Feasibility checking.

**while** \(\theta \|\delta U\| \geq Tol\) **do**

\[U^{\text{new}} = U + \theta \delta U;\]

**if** \(\|\tilde{r}(U^{\text{new}})\| \leq 1\) **then**

If \(\theta \geq T0\), exit this while loop and go to Step 3, else if \(\theta < T0\), go to Step 4.

**end if**

Reduce the factor \(\theta\) by \(\theta = \theta / 2;\)

**end while**

Step 3: Line Search.

**while** \(\theta \|\delta U\| \geq Tol\) **do**

Compute \(J(U^{\text{new}});\)

**if** \(J(U^{\text{new}}) < J(U) + \theta d_J^T \delta U\) **then**

If \(\theta \geq T0\), exit this algorithm with \(U = U^{\text{new}}\), else if \(\theta < T0\), go to Step 4.

**end if**

Reduce the factor \(\theta\) by \(\theta = \theta / 2;\)

\[U^{\text{new}} = U + \theta \delta U;\]

**end while**

Step 4: Set \(ID = 1\) and \(U = U^{\text{new}}.\)

Here, eps is the machine precision and MaxIter is the maximal number of outer iterations. We set \(\tau_J = 10^{-3}\), \(\tau_W = 10^{-2}\), \(\tau_G = 10^{-2}\) and MaxIter= 500. If any one of (1) (2) and (3) is satisfied, the iterations are terminated. Hence, a Gauss-Newton numerical scheme with Armijo line search can be developed. The resulting Gauss-Newton numerical scheme by using Armijo line search is summarized in Algorithm 2.

Algorithm 2 Gauss-Newton scheme by using Armijo line search for Image Registration: \((U, ID) \leftarrow GNAIRA(\alpha, \beta, U^0, T, R)\)

Step 1: Set \(i = 0\) at the solution point \(U^i = U^0.\)

Step 2: For (29), compute the energy functional \(J(U^i)\), its gradient \(d_J^i\) and the approximated Hessian \(H^i\) by (42).

**while** “none of the listed 3 stopping criteria are satisfied” **do**

— Solve the Gauss-Newton equation: \(H^i \delta U^i = -d_J^i;\)

— \((U^{i+1}, ID) \leftarrow ALS(U^i, \delta U^i)\) by Algorithm 1;

**if** \(ID = 1\) **then**

Exit this algorithm.

**else**

\[i = i + 1;\]

Compute \(J(U^i), d_J^i\) and \(H^i;\)

**end if**

Next, we discuss the global convergence result of Algorithm 2 for our reformulated problem (29). Firstly, we review some relevant theorem.
**Theorem 1 ([28]).** For the unconstrained optimization problem

$$\min_U J(U)$$

let an iterative sequence be defined by $U^{i+1} = U^i + \theta \delta U^i$, where $\delta U^i = -(H^i)^{-1} d_J(U^i)$ and $\theta$ is obtained by Algorithm 1. Assume that three conditions are met: (i). $d_J$ be Lipschitz continuous; (ii). the matrices $H^i$ are SPD (iii). there exist constants $\bar{\kappa}$ and $\lambda$ such that the condition number $\kappa(H^i) \leq \bar{\kappa}$ and the norm $||H^i|| \leq \lambda$ for all $i$. Then either $J(U^i)$ is unbounded from below or

$$\lim_{i \to \infty} d_J(U^i) = 0$$

and hence any limit point of the sequence of iterates is a stationary point.

**Remark.** In the above discretization leading to (29), we do not need to introduce the boundary condition. However for theory purpose, in the following, we will prove our convergence result under the Dirichlet boundary condition (namely, the boundary is fixed) and this condition is needed to prove the symmetric positive definite (SPD) property of the approximated Hessians. In practical implementation, such a condition is not required as confirmed by experiments.

In addition, define an important set $X := \{U \mid \mathbf{r}(U)_\ell \leq 1 - \epsilon, 1 \leq \ell \leq 4n^2\}$ for small $\epsilon$. So $U \in X$ means that the transformation is diffeomorphic. Under the suitable $\beta$, we assume that each $U^i$ generated by Algorithm 2 is in the $X$.

Secondly we stage a simple lemma that is needed shortly for studying $H^i$.

**Lemma 2.** Let a matrix be comprised of 3 submatrices $H = H_1 + H_2 + H_3$. If $H_1$ and $H_2$ are symmetric positive semi-definite and $H_3$ is SPD, then $H$ is SPD with $\lambda_{h_3} \leq \lambda_h$, where $\lambda_{h_3}$ and $\lambda_h$ are the minimum eigenvalues of $H_3$ and $H$ separately.

**Proof.** According to Rayleigh quotient, we can find a vector $v$ such that

$$\lambda_h = \frac{v^T H v}{v^T v}.$$  \hspace{1cm} (45)

Then we have

$$\lambda_{h_3} \leq \frac{v^T H_1 v}{v^T v} + \frac{v^T H_2 v}{v^T v} + \frac{v^T H_3 v}{v^T v} = \frac{v^T H v}{v^T v} = \lambda_h,$$  \hspace{1cm} (46)

which completes the proof. \(\square\)

**Theorem 3.** Assume that $T$ and $R$ are twice continuously differentiable. For (29), when $\phi = \phi_1, \phi_2$ or $\phi_3$, by using Algorithm 2, we obtain

$$\lim_{i \to \infty} d_J(U^i) = 0$$  \hspace{1cm} (47)

and hence any limit point of the sequence of iterates produced by Algorithm 2 is a stationary point.

**Proof.** It suffices to show that Algorithm 2 satisfies the requirements of Theorem 1. Recall $\mathbf{r}(U)$ and we can see that it is continuous. Here, we use the Dirichlet boundary condition and we can assume that $||U||$ is bounded. Then $\mathbf{r}(U)$ is a continuous mapping from a compact set to $\mathbb{R}^{4n^2 \times 1}$ and $\mathbf{r}(U)$ is proper. So for some small $\epsilon > 0$, $X$ is compact.

This manuscript is for review purposes only.
Firstly, we show that in $\mathcal{X}$, $d_J$ of (29) is Lipschitz continuous. When $\phi = \phi_1, \phi_2$ or $\phi_3$, the term $\phi(\tilde{r}(U))e^T$ in the (29) is twice continuously differentiable with respect to $U \in \mathcal{X}$. In addition, $T$ and $R$ are twice continuously differentiable. So (29) is twice continuously differentiable with respect to $U \in \mathcal{X}$ and $d_J$ is Lipschitz continuous.

Secondly, we show that in $\mathcal{X}$, $H^i = \hat{H}_1^i + \hat{H}_2^i + \hat{H}_3^i$ is SPD. By the construction of $\hat{H}_1^i$ and $\hat{H}_3^i$, they are symmetric positive semi-definite. $\hat{H}_2^i$ is symmetric positive definite under the Dirichlet boundary condition. Consequently $H^i$ is SPD.

Thirdly, we show that both $\kappa(H^i)$ and $\|H^i\|$ are bounded. We notice that in each iteration, $\hat{H}_3^i = \alpha h^2 A^TGA$ is constant and we can set $\|\hat{H}_3^i\| = M_2$. For $\hat{H}_1^i = h^2 P(T_0^T \hat{T}_0)P$, we get its upper bound $M_1$ because $T$ is twice continuously differentiable and $\mathcal{X}$ is compact. For $\phi = \phi_1, \phi_2 \text{ or } \phi_3$, $\phi$ is twice continuously differentiable with respect to $U \in \mathcal{X}$, then we have $\|\hat{H}_1^i\| \leq \frac{\beta h^2}{4} \|d\tilde{r}^T\| \|d^2\phi(\tilde{r})\| \|d\tilde{r}\| \leq M_3$.

Hence, we have

\begin{equation}
\|H^i\| \leq \|\hat{H}_1^i\| + \|\hat{H}_2^i\| + \|\hat{H}_3^i\| \leq M_1 + M_2 + M_3.
\end{equation}

So set $M = M_1 + M_2 + M_3$ and $\|H^i\| \leq M$. Set $\sigma$ as the minimum eigenvalue of $H^i$. According to Lemma 2, the smallest eigenvalue $\lambda_{min}(H^i)$ should be larger than $\sigma$. The largest eigenvalue $\lambda_{max}(H^i)$ should be smaller than $M$ due to $\lambda_{max} \leq \|H^i\|$. So the conditional number of $H^i$ is smaller than $\frac{M}{\sigma}$.

Finally, we can find that (29) has lower bound 0. So by applying Theorem 1, we finish the proof.

4.3. Multi-Level Strategy. In practice, we employ the multilevel strategy. We firstly coarsen the template $T$ and the reference $R$ by $L$ levels. Here, we set $T_L = T$ and $R_L = R$ in the finest level and $T_1$ and $R_1$ in the coarsest level. Then we can obtain $U_1$ by solving our model (18) on the coarsest level. In order to give a good initial guess for the finer level, we adopt an interpolation operator on $U_1$ to obtain $U_2$ as the initial guess for the next level. We repeat this process and get the final registration on the finest level. A multi-level strategy has several advantages: in the coarse level, only important patterns can be considered and it is a standard technique used in order to avoid getting trapped in a meaningless local minimum; the computational speed is very fast because of less variables than on the fine level; the solution on the coarse level can be a good initial guess for the fine level.

The multilevel scheme representing our main algorithm is summarized below where $L^H_T$ is an interpolation operator based on bi-linear interpolation techniques and $L^H_R$ is a restriction operator for transferring information to a coarser level.

5. Numerical Results. In this section, we will give some numerical results to illustrate the performance of our model (18). We hope to achieve 3 aims:

1). Which choice of $\phi$ is the best for our model (18)?
2). We wish to compare with the current state-of-the-art methods (with codes listed for readers' benefit) in the literature for good diffeomorphic mapping:
   (a) Hyperelastic Model [7]: code from http://www.siam.org/books/fa06/
   (b) LDDMM [37]: code from https://github.com/C4IR/FAIR.m/tree/master/add-ons/LagLDDMM
   (c) Diffeomorphic Demons (DDemons) [51]: code from http://www.insight-journal.org/browse/publication/154
   (d) QCHR [30]: code provided by the author Dr. Kamin Chu Lam.

All of the tests are performed on a PC with 3.40 GHz Intel(R) Core(TM) i7-4770 microprocessor, and with installed memory (RAM) of 32 GB.
Algorithm 3 Multilevel Image Registration: \( U \leftarrow \text{MLIR}(\alpha, \beta, U^0, T, R) \)

Step 1: Compute the largest possible number of levels based on size of \( T, R \):
\[
L = \text{Maxlevel}; \ 	ext{Define the coarsest level as level 1.}
\]
Work out the multilevel representation of given images \( R \) and \( T \):
\[
R_L = R, T_L = T;
R_{L-1} = T^H_R R_L, \quad T_{L-1} = T^H_R T_L; \ldots ;
R_1 = T^H_R R_2, T_1 = T^H_R T_2.
\]
Step 2: Set the initial guess on the coarsest level:
\[
U_L = U^0, \quad U^0_j = I_H^t U_{j+1}^0, \ j = L - 1, \ldots, 1.
\]
Step 3: Apply Algorithm 2 on the coarsest level \( i = 1 \) with \( U^0_1 \):
\[
(U_1, ID) \leftarrow \text{GNAIRA}(\alpha, \beta, U^0_1, T_1, R_1);
\]
if \( ID = 1 \) then
Exit this algorithm;
end if
for level \( j = 2 : L \) do
Interpolate the solution from a coarser level \( U^0_j = I_H^t U_{j-1}^0 \);
Apply Algorithm 2 on level \( j \) :
\[
(U_j, ID) \leftarrow \text{GNAIRA}(\alpha, \beta, U^0_j, T_j, R_j);
\]
if \( ID = 1 \) then
Exit this algorithm;
end if
end for

3). Most importantly, we like to test and highlight the advantages of our new model.

Let \( y \) be the final transform obtained by a particular model for registering two
given images \( T, R \). We use the following three measures to quantify the performance
of this model and use them for later comparisons:

(i). \( \text{Re}_{SSD} \) (the relative Sum of Squared Differences) which is given by
\[
\text{Re}_{SSD} = \frac{\| T(y) - R \|^2}{\| T - R \|^2}; \tag{49}
\]

(ii). \( \min \det(J_y) \) and \( \max \det(J_y) \) that are the minimum and the maximum of the
Jacobian determinant of this transformation;

(iii). Jaccard similarity coefficient (\( JSC \)) as defined by
\[
JSC = \frac{|DT_r \cap R_r|}{|DT_r \cup R_r|}, \tag{50}
\]
where \( DT_r \) and \( R_r \) represent respectively the segmented regions of interest
(e.g. certain image feature such as an organ) in the deformed template (after
registration) and the reference. Hence, \( JSC \) is the ratio of the intersection
of \( DT_r \) and \( R_r \) to the union of \( DT_r \) and \( R_r \) \[29\]. \( JSC = 1 \) shows that a
perfect alignment of the segmentation boundary and \( JSC = 0 \) indicates that
the segmented regions have no overlap after registration.

Before computing \( JSC \), in the first three examples below, we have employed a
segmentation algorithm to segment the main features in both \( T \) and \( R \) but for
the 4th example, the segmentation was manually done for both \( T \) and \( R \).

In practice, we scale the intensity value of \( T \) and \( R \) to \([0, 255]\). Here, we state a strategy
for choosing the parameters. For our model \((18)\), \( \alpha \) should be related to energy \( D[u_0] \).
where \( u_0 \) is the initial guess for the displacement, and \( \beta \) should be related to \( \alpha \).

Empirically, we set \( \alpha \in [\alpha_1, \alpha_2] \), where \( \alpha_1 = 0.5D[u_0]10^{-2} \) and \( \alpha_2 = 2D[u_0]10^{-2} \).

Respectively for \( \phi = \phi_1, \phi_2, \phi_3 \), we set \( \beta \in [3\alpha, 5\alpha], [0.5\alpha, 2\alpha] \) and \([\alpha, 5\alpha] \). For simplicity, we denote by New 1, New 2 and New 3 the model \((18)\) with \( \phi_1, \phi_2 \) and \( \phi_3 \) respectively.

It should be noted that a good registration result should produce a small \( Re_{SSD} \), be diffeomorphic and yield a large \( JSC \) value for a region of interest.

### 5.1. Example 1 — Improvement over the diffusion model
In this example, we test a pair of real medical images, X-ray Hands of resolution 128 \( \times \) 128.

Figure 3 (a-b) show the template and the reference. We compare our model with the diffusion model and study the improvement over it. In implementation, for both models, we use a five-step multilevel strategy.

We conduct two experiments using different parameters:

i). \textbf{Fixed parameters}. Our first choice uses fixed parameters. For New 1-3, we set \( \beta = 7, \beta = 1 \) and \( \beta = 9 \) respectively, and fix \( \alpha = 2 \). To be fair, we also choose \( \alpha = 2 \) for the diffusion model. In this case, Figure 3 shows the deformed templates \( T(\gamma) \) from 4 models. From it, we can see that all four models can produce visually satisfactory results. To differentiate them, we have to check the quantitative measures from Table 1. We can notice that the transformation obtained by the diffusion model is non-diffeomorphic due to \( \min \det(J_\gamma) < 0 \) (i.e. mesh folded, though visually pleasing and the \( Re_{SSD} \) is small). Figure 4 illustrates the transform \( \gamma = x + u \) locally at its folding point. In contrast, our New 1-3 can generate diffeomorphic transformations.

ii). \textbf{Optimized parameters}. The second choice uses the fine tuned parameters for the diffusion model. We tested \( \alpha \in [1, 500] \) and found the smallest \( \alpha = 430 \) with which the diffusion model generates a diffeomorphic transformation. Then for our model, we also set \( \alpha = 430 \) (which is not optimized in order to favour the former) and set \( \beta = 5 \) for New 1-3 (to test the robustness of our model). Table 1 shows the detailed results for this second test. From it, we can see that the \( Re_{SSD} \) and \( JSC \) of our model are similar to the diffusion model. And the transformations obtained by New 1-3 are all diffeomorphic while the diffusion model is only diffeomorphic with the help of an optimized \( \alpha \). This shows that our model possesses the robustness (in the sense of not requiring optimized \( \alpha \)) with the help of a positive \( \beta \).

Hence, this example demonstrates that our New 1-3 are robust and can all help to get an accurate and diffeomorphic transformation.

### Table 1

<table>
<thead>
<tr>
<th>Test example 1 – Comparison of the new model (New 1-3) with the diffusion model based on fixed ( \alpha ) and an optimized ( \alpha ) for the latter. Clearly the latter model can produce an incorrect result if not tuned while New 1-3 are less sensitive to ( \alpha ) with the help of ( \beta ).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>First Test ( \alpha = 2 )</strong></td>
</tr>
<tr>
<td>New 1</td>
</tr>
<tr>
<td>New 2</td>
</tr>
<tr>
<td>New 3</td>
</tr>
<tr>
<td>Diffusion Model</td>
</tr>
<tr>
<td><strong>Second Test ( \alpha = 430 )</strong></td>
</tr>
<tr>
<td>New 1</td>
</tr>
<tr>
<td>New 2</td>
</tr>
<tr>
<td>New 3</td>
</tr>
<tr>
<td>Diffusion Model</td>
</tr>
</tbody>
</table>

### 5.2. Example 2 – Test of large deformation and comparison of models
As known, if the underlying deformation is small, it is generally believed that most
models can deliver diffeomorphic transformations. This belief is true if one keeps increasing $\alpha$, which in turn compromises the registration quality by resulting in an increase in $Re_{SSD}$ (as seen in 2 tests of $\alpha$ in Example 1 where the larger $\alpha = 430$ achieves diffeomorphism for diffusion with a worse $Re_{SSD}$ value).

Therefore to test the capability of a registration model, we need to take an example that requires large deformation. To this end, we consider Example 2—a classic synthetic example consisting of a Disc and a C shape of resolution $128 \times 128$ as shown in Figure 5 (a-b). We compare our 3 models (New 1-3) with 5 other models: the hyperelastic model, LDDMM, DDemons, QCHR and the diffusion model in registration quality and performance. For this example, we use a five-step multilevel strategy for our model, the hyperelastic model and the diffusion model. For LDDMM and QCHR, we use a three-step multilevel strategy. We use a one-step multilevel
strategy for DDemons as we found that multilevel does not improve the results.

Following our stated strategy for choosing the parameter for our model, we set
\( \beta = 80, 120, 100 \) for New 1-3 respectively and fix \( \alpha = 70 \). To be consistent, we also set
\( \alpha = 70 \) for the diffusion model. For the hyperelastic model, LDDMM and QCHR, we
set respectively \( \{ \alpha_1 = 100, \alpha_2 = 0, \alpha_3 = 18 \} \), \( \alpha = 400 \) and \( \{ \alpha = 0.1, \beta = 1 \} \) as used in
the literature [7, 37, 30] for the same example. For the parameters of DDemons, we
tried to optimize the parameters \( \{ \sigma_s, \sigma_g \} \) in the domain \([0.5, 5] \times [0.5, 5] \) and took the
optimal choice \( \{ \sigma_s = 1.5, \sigma_g = 3.5 \} \).

We now present the comparative results. Figure 5 (c-j) show that except for
the diffusion model, all the other models can produce the accepted registered results.
Especially, our model and LDDMM are slightly better than the hyperelastic model,
DDemons and QCHR. It is pleasing to see that the new model produces equally
good results for this challenging example. From Table 2, we see that our New 1-3,
hyperelastic model, LDDMM, DDemons and QCHR produce \( \text{min det}(J_y) > 0 \) i.e.
the transformations obtained by these five models are diffeomorphic but the diffusion
model fails again with \( \text{min det}(J_y) < 0 \).

Because New 1-3 are motivated by the QCHR model, we now discuss the results
about these two types of models. On the one hand, according to Table 2, we can
find that our model takes less time. This is because, as we have mentioned, the
algorithm for QCHR needs to solve alternatively two subproblems (including several
linear systems) in each iteration. Its convergence cannot be guaranteed. However,
our model only needs to solve one linear system in each iteration. In addition, we
employ the Gauss-Newton method which can be superlinearly convergent under the
appropriate conditions. As we have also remarked, the QCHR algorithm can have
convergence problems. This is now illustrated in Figure 6 where we plot the relative
residual of our model (New 3) and the relative residual of QCHR. We observe that
New 3 decreases to below \( 10^{-2} \) though not monotonically, but the relative residual of
QCHR does not decrease and is over 0.1.

On the other hand, we can compare the obtained solutions’ quality by checking
the energy functionals. Using the same QCHR functional, the QCHR solution for
Example 2 has the value 1042 while the transformation obtained by New 3 gives the
value 147 which is much smaller. This indicates that the result obtained by the QCHR
algorithm is not accurate. This is consistent with the fact that the \( \text{Re}_{\text{SSD}} \) and \( JSC \)
of New 3 are also better than QCHR. Both discussions reach the same conclusion:
the QCHR algorithm cannot obtain the minimizer of the original QCHR functional.

### 5.3. Example 3 – Comparison of models for a challenging test.

Here, we illustrate the fact that area preservation between images can become unnecessary
and trying to enforce it (as in the hyperelastic model) can fail to register an image.
We choose the particular template and reference images, as shown in Figure 7 (a-b),
having significantly different areas in their main features – here the area of ‘Disc’ is much larger than ‘C’. The resolution of the images is $512 \times 512$. We test the performance of New 1-3 and other models. In this example, we use a seven-step multilevel strategy for New 1-3, the hyperelastic model and the diffusion model. For LDDMM and QCHR, we use a five-step multilevel strategy. We use a single level for DDemons (since multilevels do not help).

In choosing the parameters for all the models to register this example, we first follow our strategy to set $\beta = 250, 50, 100$ for New 1-3 respectively and fix $\alpha = 50$. To be consistent, we also set $\alpha = 50$ for the diffusion model. For the hyperelastic model, we also set $\alpha_l = 50$ because it contains the diffusion term, and take $\alpha_s = 0$. For the third parameter $\alpha_v$ in the hyperelastic model, we test it in the range $[55, 150]$ and choose its optimal value $\alpha_v = 75$. For LDDMM and QCHR, we set the default value $\alpha = 400$ and $\{\alpha = 0.1, \beta = 1\}$ as the previous example. For the parameters of DDemons, we test the parameters $\{\sigma_s, \sigma_g\}$ in the domain $[0.5, 5] \times [0.5, 5]$ and choose

---

**Fig. 5.** Test example 2 results of Disc to C. The percentage value shows $\text{Re}_{\text{SSD}}$ error. In the top row, there are the template and the reference. In the second and third row, there are the deformed templates obtained by New 1-3 and 5 other models separately. The landmarks in the template and reference are only used for QCHR and the last result (j) by Diffusion is evidently not correct.
The solid line indicates the relative residual of New 3. And the dot line shows the relative residual of the second subproblem in QCHR. Here, we can find that in the same 50 iterations, the relative residual of New 3 is decreasing to below $10^{-2}$, however the relative residual of QCHR is not decreasing and over 0.1. Hence, the convergence of the algorithm for QCHR can not be guaranteed.
Table 3
Example 3 – Comparison of the new model (New 1-3) with 5 other models.

<table>
<thead>
<tr>
<th>Model</th>
<th>Resolution</th>
<th>Res SSD</th>
<th>min det(J_y)</th>
<th>max det(J_y)</th>
<th>JSC</th>
<th>time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>New 1</td>
<td>512 x 512</td>
<td>3.06%</td>
<td>0.0325</td>
<td>38.2272</td>
<td>78.93%</td>
<td>412.87</td>
</tr>
<tr>
<td>New 2</td>
<td>512 x 512</td>
<td>0.08%</td>
<td>0.0035</td>
<td>64.3050</td>
<td>97.84%</td>
<td>281.95</td>
</tr>
<tr>
<td>New 3</td>
<td>512 x 512</td>
<td>0.07%</td>
<td>0.0064</td>
<td>60.7493</td>
<td>97.82%</td>
<td>262.17</td>
</tr>
<tr>
<td>Hyperelastic Model</td>
<td>512 x 512</td>
<td>3.85%</td>
<td>0.4895</td>
<td>7.0781</td>
<td>75.49%</td>
<td>46.16</td>
</tr>
<tr>
<td>LDDMM</td>
<td>512 x 512</td>
<td>0.41%</td>
<td>0.0184</td>
<td>40.2844</td>
<td>95.03%</td>
<td>218.32</td>
</tr>
<tr>
<td>DDemons</td>
<td>512 x 512</td>
<td>2.83%</td>
<td>9.6 x 10^{-6}</td>
<td>44.8529</td>
<td>80.56%</td>
<td>&gt; 5000</td>
</tr>
<tr>
<td>QCHR Model</td>
<td>512 x 512</td>
<td>2.03%</td>
<td>0.0207</td>
<td>4.4744</td>
<td>84.23%</td>
<td>4716.7</td>
</tr>
<tr>
<td>Diffusion Model</td>
<td>512 x 512</td>
<td>0.32%</td>
<td>-38.8337</td>
<td>286.3411</td>
<td>94.68%</td>
<td>5.52</td>
</tr>
</tbody>
</table>

Fig. 7. Example 3 results of a large Disc to small letter C: in the top row, there are the template and reference. In the second and third row, there are the deformed templates obtained by model (18) and other models separately. The landmarks in the template and reference are only used for QCHR.

in Table 4. Clearly the table indicates that QCHR does generate a smoother Beltrami coefficient than our model New 3 for both Examples 2-3, not a smoother deformation field. Hence, the model which only regularizes the Beltrami coefficient rather than the deformation is not sufficient to produce an accurate deformed template.

(ii). On the second point, we now illustrate the importance of landmarks for
Fig. 8. Tests of QCHR with different landmarks: Example 2 (row 1) and Example 3 (row 2). On the left 3 columns of row 3, we show the registered templates for row 1. On the right 3 columns of row 3, we show the registered templates for row 2. Here, we can see that the accuracy of QCHR improves with the increase of landmarks.

Table 4

Comparison of smoothness measures for solutions obtained by New 3 and QCHR. The Beltrami coefficient $\mu$ obtained by QCHR is smoother than New 3 and the displacement $u$ obtained by New 3 is smoother than QCHR.

<table>
<thead>
<tr>
<th>Example</th>
<th>$|\nabla u|_{L^2}$</th>
<th>$|\mu(y)|_{L^2}$</th>
<th>$|\nabla \mu(y)|_{L^2}$</th>
<th>RE SSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QCHR with 16 pairs of landmarks</td>
<td>2.1099</td>
<td>0.6930</td>
<td>0.2762</td>
<td>4.90%</td>
</tr>
<tr>
<td>New 3</td>
<td>1.6355</td>
<td>0.5024</td>
<td>0.2800</td>
<td>0.06%</td>
</tr>
<tr>
<td>Example 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QCHR with 20 pairs of landmarks</td>
<td>1.3466</td>
<td>0.5851</td>
<td>0.0868</td>
<td>2.03%</td>
</tr>
<tr>
<td>New 3</td>
<td>1.3913</td>
<td>0.3352</td>
<td>0.1090</td>
<td>0.07%</td>
</tr>
</tbody>
</table>

As a final comparison of New 3 with LDDMM and QCHR, Figure 9 plots the magnitudes of the Jacobian determinants of their transformations. It can be seen that New 3 and LDDMM give a similar pattern but both are different from QCHR.

5.4. Example 4— Comparison of the new model with other models. In the final test, we test a pair of anonymized CT images in resolution $512 \times 512$ from the Royal Liverpool University Hospital. Figure 10 (a-b) show the template and the reference. The template was taken in September 2016 and the reference was taken in May 2016. We want to compare the changes of our interested regions of abdominal aortic aneurysm with stents inserted inside them (with cross sections shown as two
while ‘circles’ in images in Figure 10 (a-b)) during these 4 months. In addition, the
interested region is used to compute JSC. The small white region on top of the
images helps us to identify the correct slice to compare.

Here, following the previous example, we use the same multilevel strategy: a
seven-step multilevel strategy for our model, the hyperelastic model and the diffusion
model, a five-step multilevel strategy for LDDMM and QCHR and a one-step
multilevel strategy for DDemons.

Following our strategy for choosing the parameter of our model, we set $\alpha = 20$ and
set $\beta = 100, 40, 75$ with New 1-3 respectively. For the diffusion model and LDDMM,
we test $\alpha$ from $[100, 2000]$ and set the optimal value 1300 and 500 respectively. For
the hyperelastic model, we set $\{\alpha_l = 20, \alpha_s = 0, \alpha_v = 50\}$. We use the default value
$\{\alpha = 0.1, \beta = 1\}$ for QCHR. For the parameters of DDemons, we test the parameters
$\{\sigma_s, \sigma_g\}$ in the domain $[0.5, 5] \times [0.5, 5]$ and choose $\{\sigma_s = 4, \sigma_g = 4.5\}$.

With the optimized parameters, all the models in this example generate diffeomorphic
transformations as seen from Table 5. DDemons and QCHR for this example
are not as good as other models because they give worse Re_SSD and JSC. A worse
JSC means the interested regions obtained by these two methods have significant
differences from the reference (Figure 10 (h-i)). The diffusion model obtains a good
JSC, however its deformed template is a bit far (overall) from the reference (since
Re_SSD = 10.02%). The other 2 models (Hyperelastic, LDDMM) generate good
Re_SSD and JSC. However, our models produce the lowest Re_SSD and the best
JSC. Hence, for this example of real images, our model is competitive to the state-
of-the-art methods. Though there is broad agreement between Re_SSD and JSC,
one has to combine with segmentation models to ensure the strict agreement.

This manuscript is for review purposes only.
Fig. 10. Example 4 – Registration results of a pair of CT images: the template $T$ and the reference $R$ in the top row. The contours show the regions of interest. In the second and third rows, we show the deformed templates obtained by 8 models. The 5 landmarks in the template and the reference are only used by QCHR.

**Remark.** According to the above four examples, our New 1 is not robust while New 2-3 can both generate accurate and diffeomorphic transformations. However, we recommend New 3 as the first choice because of the least computing time and the best quality, and New 2 as the second choice.

We also test these four examples with the Dirichlet boundary condition. Similar results for Examples 1 and 4 are obtained. However, for Examples 2 and 3, the transformations would be different since the boundary is better modeled by the Neumann’s condition.

**6. Conclusions.** Controlling mesh folding is a key issue in image registration models to ensure local invertibility. Many existing models either do not impose any further controls on the underlying transformation beyond smoothness (so potentially generating unrealistic or non-physical transforms or mapping) or impose a direct (often strongly biased e.g. towards area or volume preservation) control on some explicit
function of the measure $\det(J_p)$. This paper introduces a novel, unbiased and robust regularizer which is reformulated from Beltrami coefficient framework to ensure a diffeomorphic transformation. Moreover we find that a direct approach (our New 1) from this Beltrami reformulation provides an alternative but less competitive method but further refinements (especially our New 3) of this new regularizer can give rise to more robust models than the existing methods. We highly recommend our model New 3 i.e. (18) with $\phi = \phi_3$.

In designing optimization methods for solving the resulting highly nonlinear variational model, we give a suitable approximation of the exact Hessian matrix which is necessary to derive a convergent iterative method. Our test results can show that the new model (New 1-3, especially New 3) is competitive with the state-of-the-art models. The main advantage lies in robustness. Our future work will include extensions to 3D problems, multi-modality models and development of faster iterative solvers.

**Appendix A. Computation of matrices $A$ and $G$ in §4.1.2.** Set $B = I_2 \otimes I_{n+1} \otimes \partial_{n}^{1,h} \in \mathbb{R}^{2n(n+1) \times 2(n+1)^2}$, $C = I_2 \otimes \partial_{n}^{1,h} \otimes I_{n+1} \in \mathbb{R}^{2n(n+1) \times 2(n+1)^2}$,

$$\partial_{n}^{1,h} = \frac{1}{h^2} \begin{bmatrix} -1 & 1 & \cdots & \cdots & -1 & 1 \\ -1 & 1 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ -1 & 1 \\ 1 & -1 \\ 1 & -1 \end{bmatrix} \in \mathbb{R}^{n,n+1}, \quad A = \begin{bmatrix} B \\ C \end{bmatrix} \in \mathbb{R}^{4n(n+1) \times 2(n+1)^2},$$

where $\otimes$ denotes a Kronecker product. To represent the difference between interior and boundary pixels, we need to introduce a diagonal matrix

$$G = \begin{bmatrix} G_1 & 0 & 0 \\ 0 & G_2 & 0 \\ 0 & 0 & G_1 \\ 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{4n(n+1) \times 4n(n+1)},$$

where $G_1$ and $G_2$ are diagonal matrices. For $G_1, G_{1,4+i+j,n-i+j+n} = 1$ if $0 \leq i \leq n - 1$, $1 \leq j \leq n - 1$ or if $0 \leq i \leq n - 1$, $j = 0$, $n$. Similarly, for $G_2, G_{2,4+i+j+n-i+j+n} = 1$ if $1 \leq i \leq n - 1$, $0 \leq j \leq n - 1$ or if $i = 0$, $n$, $0 \leq j \leq n - 1$.

**Appendix B. Computation of the vector $\vec{r}(U)$ in §4.1.3.** We demonstrate how to build the linear interpolation $L$ in $\Delta V_1 V_2 V_5$, in Figure 2.

First of all, denote the 3 vertices of this triangle by $V_1 = x^{1,1}$, $V_2 = x^{2,1}$ and $V_5 = x^{1,5,1.5}$. Set $L(V_1) = (u_1^{1,1}, u_2^{1,1})$, $L(V_2) = (u_1^{2,1}, u_2^{2,1})$ at the vertex pixels, and $L(V_5) = (u_1^{1.5,1.5}, u_2^{1.5,1.5})$ at the cell centre (approximated values). Here the linear approximations are $L(x_1, x_2) = (a_1 x_1 + a_2 x_2 + a_3, a_4 x_1 + a_5 x_2 + a_6)$.

After substituting $V_1, V_2$ and $V_5$ into $L$, we get

$$\begin{bmatrix} x_1^2 - x_1^1.5 \\ x_2^1 - x_1^1.5 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} u_1^{1.1} - u_1^{1.5,1.5} \\ u_2^{1.1} - u_1^{1.5,1.5} \end{bmatrix},$$

$$\begin{bmatrix} x_1^1 - x_1^1.5 \\ x_2^1 - x_1^1.5 \end{bmatrix} \begin{bmatrix} a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} u_1^{1.1} - u_2^{1.5,1.5} \\ u_2^{1.1} - u_2^{1.5,1.5} \end{bmatrix}.$$

Then

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \frac{1}{\det} \begin{bmatrix} x_1^2 - x_1^1.5 & -x_2^2 + x_2^1.5 \\ -x_1^2 + x_1^1.5 & x_2^1 - x_1^1.5 \end{bmatrix} \begin{bmatrix} u_1^{1.1} - u_1^{1.5,1.5} \\ u_1^{2.1} - u_1^{1.5,1.5} \end{bmatrix}.$$

This manuscript is for review purposes only.
According to (52) and (53), we can formulate two matrices $D1 \in \mathbb{R}^{4n^2 \times (n+1)^2}$ and $D2 \in \mathbb{R}^{4n^2 \times (n+1)^2}$ such that

$$\begin{align*}
a_1 - a_5 &= [D1, -D2]U = A_1 U \in \mathbb{R}^{4n^2 \times 1}, \\
a_4 + a_2 &= [D2, D1]U = A_2 U \in \mathbb{R}^{4n^2 \times 1}, \\
a_1 + a_5 &= [D1, D2]U = A_3 U \in \mathbb{R}^{4n^2 \times 1}, \\
a_4 - a_2 &= [D2, -D1]U = A_4 U \in \mathbb{R}^{4n^2 \times 1}.
\end{align*}$$

Here, $a_\theta = (a_\theta^1, ..., a_\theta^{4n^2})^T$, $\theta = 1, 2, 4, 5$, where $a_\theta^l = a_\theta^{i,j,k}$ and $l = (k-1)n^2 + (j-1)n + i$.

Next using the Hadamard product $\odot$, we get a compact form for

$$\begin{align*}
\tilde{r}^1(U) &= A_1 U \odot A_1 U + A_2 U \odot A_2 U, \\
\tilde{r}^2(U) &= 1/((A_3 U + 2) \odot (A_3 U + 2) + A_4 U \odot A_4 U), \\
\tilde{r}(U) &= \tilde{r}^1 \odot \tilde{r}^2 \in \mathbb{R}^{4n^2 \times 1}.
\end{align*}$$

Appendix C. Computing the gradient and approximated Hessian of the term (37). Here, as an example, we set $n = 2$ and $\phi = \phi_1$ to compute the gradient and approximated Hessian of the discretized Beltrami term (37).

Because of $n = 2$, we have

$$U = (u_0^1, ..., u_1^0, ..., u_1^2, ..., u_2^0, ..., u_2^2, ..., u_2^{2,2})^T \in \mathbb{R}^{18 \times 1}.$$ 

From (52)-(53), we can formulate two matrices $D1, D2 \in \mathbb{R}^{16 \times 9}$ respectively by:

$$\begin{align*}
\begin{bmatrix}
-2 & 2 \\
-2 & 2
\end{bmatrix}
\begin{bmatrix}
-1 & -1 & 1 & 1 \\
-1 & -1 & 1 & 1
\end{bmatrix} =
\begin{bmatrix}
-1 & -1 & 1 & 1 \\
-1 & -1 & 1 & 1
\end{bmatrix}.
\end{align*}$$

Then we can build $A_1, A_2, A_3$ and $A_4$ and compute $\tilde{r}^1, \tilde{r}^2$ and $\tilde{r}$ by (54). According to (39), we have $d\vec{f} \in \mathbb{R}^{16 \times 18}$.

When $\phi(v) = \phi_1(v)$, we have $\phi'_1(v) = \frac{2}{(v_1 - 1)^2}$, $\phi''_1(v) = \frac{6}{(v_1 - 1)^4}$ and so $d\phi(\vec{r}) = (\frac{2}{(v_1 - 1)^2}, ..., \frac{2}{(v_{16} - 1)^2})^T$ in (38). In (40) the $ith$ diagonal element $[d^2 \phi(\vec{r})]_{ii} = \frac{6}{(v_i - 1)^4}$, $1 \leq i \leq 16$. Similarly when $\phi(v) = \phi_2$, $d\phi(\vec{r}) = (\frac{2}{(v_1 - 1)^3}, ..., \frac{2}{(v_{16} - 1)^3})^T$ and $[d^2 \phi(\vec{r})]_{ii} = \frac{12}{(v_i - 1)^5}$. When $\phi(v) = \phi_3$, $d\phi(\vec{r}) = (\frac{2}{(v_1 - 1)^4}, ..., \frac{2}{(v_{16} - 1)^4})^T$ and $[d^2 \phi(\vec{r})]_{ii} = \frac{24}{(v_i - 1)^6}$.

Hence, we can get $d_3$ in (38) and $H_3$ in (40).
REFERENCES

Computational Methods for Inverse Problems

Image matching as a diffusion process: an analogy with Maxwell’s demons

Optimization theory and methods: nonlinear programming

Solution of sparse indefinite systems of linear equations


