Construction of Diffeomorphisms by Controlling Jacobian Determinant and Curl with Lagrange Multipliers

Zicong Zhou[1,*] and Guojun Liao[2]

[1] Department of Pharmacology and Neuroscience,
University of North Texas Health Science Center,
3500 Camp Bowie Blvd, Fort Worth, TX 76106, USA
[*]Corresponding Author: Zicong.Zhou@unthsc.edu
[2] Department of Mathematics, University of Texas at Arlington,
701 S. Nedderman Dr, Arlington, TX 76019, USA

Abstract. The latest version of the Variational Principle (VP) [16] generates unique non-folding grids (diffeomorphisms) with prescribed Jacobian determinant (JD) and curl whose solutions belongs a diffeomorphic Lie algebra. This was theoretically used for the deformation field of an optimal control approach to non-rigid image registration [18] (referred to as the VP-control method). In order to build a deep-learning optimization mechanism compatible with the idea of the VP-control method, several fundamental issues need to be addressed. In this paper, a Lagrange-Multiplier formulation of VP (referred to as LM-VP) is introduced, for three purposes: (1) arguing that optimizing over JD and curl indeed computationally reconstructs a given ground truth grid; (2) bypassing the necessity of computing the control functions of JD and curl in VP; (3) presenting a deep-learning loss function based on the VP-control method via a similar Lagrange-Multiplier formulation. Demonstrating these purposes is given with the model descriptions, computational algorithms and preliminary numerical examples. This work strongly supports a parallel research that emphasizes performance of the proposed loss function for deep-learning.

Keywords: Adaptive grid generation, Computational diffeomorphism, Jacobian determinant, Curl, deformable image registration, Lagrange Multiplier

1 Introduction

In the task of image registration, the deformation field is the solution warps the moving/-source image toward the fixed/target image so that the moved image becomes close to the fixed image. The quality of the registered image is predominantly depended on the model of the deformation field and its corresponding computational schemes. Image registration problems were traditionally handled with classical methods until the deep-learning approaches entered field. The classical methods are typically designed to capture specific properties of a meaningful deformation field. Different methods may offer various advantages over different formats of image. The computational schemes of classical methods are generally simple, straight forward and gradient-descent based, which require tons

of manually tuning parameters and selecting initializations for different datasets. This not only consume more computational time, but also sets extremely high thresholds for practical applications. It is inevitable to incorporate the deep-learning approaches. An early attempt was made to use our model under optimal control functions in [20], which has already demonstrated good performance across various datasets compared to some of the leading deep-learning methods of the time. The challenges encountered in [20] were: (1) the control functions are explicit in the optimizing scheme; therefore, (2) at the end of each training epoch, it performs one time of our registration with control functions to compute the control functions and the Laplacian of the deformation field, which is very computationally intensive; hence, (3) to balance the additional computations during the registration at the end of each epoch, it was forced the registration to loop only one iteration to approximate the values of control functions and the Laplacian of the deformation field, which feeds the indefinite values of control functions to the next iteration. To handle these situations, we design a Lagrange-Multipliers approach to VP (that is LM-VP) that (1) converts the control functions from constraints into regulerizers; (2) skips the necessity to compute the control functions explicitly; (3) carries the same motivation to modify the image registration method VP-control. Similarly, numerical examples confirm that the Lagrange-Multipliers approach is effective for the image registration problem as well. Based on these findings, we proposed a novel smoothing loss to modify our deep-learning optimization strategy in [20]. This paper examines the theoretical legitimacy and the results modified deep-learning optimization will be summarized in a separate work.

1.1 Grids as Deformation Fields for Image Registration

The problem of computational construction of diffeomorphisms is the study about how to present a desired and meaningful distribution of grid points over a volumetric domain of a certain 2D/3D object, or the surface of the object, as summarized in [13]. Sometime, it is also called adaptive generation of non-folding grids, such as, a background grid of an interested phenomena was constructed in [9]; variational optimization is used for grid generation in [3], etc. Different approaches to the task may inspired by purposely practical demands. In the problem of image registration, the diffeomorphism solutions (non-folding and smooth grids) are critically desired for re-sampling the moving (source) image to match as close to the fixed (target) image so that registered/re-sampled images are without distortions. Its idea has been applied in constructing deformable image registration methods, such as in [4,11]. The recently revised Variation Principle for grid generation was developed for the purpose of describing the deformation fields outputted from the problem of image registration.

Our Grid Generation Method In [2], the deformation constructs $\boldsymbol{T}$ with prescribed JD, by a scalar function $0 < f \in C^1$, whose key component is the solution to a **divergence – curl** system (similar to the constraint equations in (3)). Its divergence is approximated by $f - 1$ and curl is assigned to $\mathbf{0}$ due to the challenge of realizing curl beforehand, where curl models the local cell-rotation. Consequently, grids formed by the deformation method are not unique, in [12]. To overcome it, the original VP was

proposed in [5] and studied further in [19]. In 3D grid generations, the original VP gives inaccurate approximations around the true solution. Attempts were made to analyze the uniqueness [14] of such transformations, but it only turned out this is true for the case of small deformation [15]. Fortunately, in despite of the general uniqueness problem remaining open, the latest version of VP works well and fast.

1.2 Variational Principle for Grid Generation

Before moving into the formation of LM-VP, let us briefly review what VP does and how each of the constituted factors work. Here, we only describe the revised-VP in [16]. Let a simply-connected, bounded $\Omega \subset \mathbb{R}^3$ (similar in $\mathbb{R}^2$) be the domain and $\boldsymbol{\omega} =< x, y, z >\in \Omega$. Let a scalar function $f_t > 0$ and a vector-valued function $\boldsymbol{g}_t$ (The subscript t labels "target") on Ω, satisfy

$$\int_\Omega f_t(\boldsymbol{\omega}) d\boldsymbol{\omega} = |\Omega| \qquad \text{and} \qquad \nabla \cdot \boldsymbol{g}_t = 0, \tag{1}$$

where the first indicates f_t like JD and the second requires $\boldsymbol{g}_t$ like curl, respectively. Given the original grid $\boldsymbol{\phi}_o \in H_0^2(\Omega)$, we look for a diffeomorphic transformation $\boldsymbol{\phi} = \boldsymbol{\phi_m} \circ \boldsymbol{\phi}_o = \boldsymbol{\phi_m}(\boldsymbol{\phi}_o) \in H_0^2(\Omega)$, where $\boldsymbol{\phi_m} = \boldsymbol{id} + \boldsymbol{u}$ is an intermediate transformation that left-translates $\boldsymbol{\phi}_o$ to $\boldsymbol{\phi}$ and implies $\delta\boldsymbol{\phi_m} = \delta\boldsymbol{u}$, such that, the cost functional — Sum of Squared Differences (SSD) is minimized:

$$\mathrm{SSD}_1(\boldsymbol{\phi}) = \frac{1}{2}\int_\Omega [(\det\nabla\boldsymbol{\phi} - f_t)^2 + |\nabla \times \boldsymbol{\phi} - \boldsymbol{g}_t|^2] d\boldsymbol{\omega} \tag{2}$$

$$\text{subjects to } \begin{cases} \nabla \cdot \boldsymbol{\phi_m} = f - 1 \\ \nabla \times \boldsymbol{\phi_m} = \boldsymbol{g} \end{cases} \Rightarrow \Delta\boldsymbol{\phi_m} = \nabla f - \nabla \times \boldsymbol{g} = \boldsymbol{F}(f, \boldsymbol{g}) \text{ in } \Omega, \tag{3}$$

with $\boldsymbol{u} = \boldsymbol{0}$ on $\partial\Omega$ where f, $\boldsymbol{g}$ and $\boldsymbol{F}$ are control functions. When SSD_1 is minimized, the final deformation field $\boldsymbol{\phi}$ is expected, by moving from $\boldsymbol{\phi}_o$ with $\boldsymbol{\phi_m}$, so that $\det\nabla\boldsymbol{\phi}$ is close to the prescribed JD, f_t, and $\nabla \times \boldsymbol{\phi}$ is closed to the prescribed curl, $\boldsymbol{g}_t$.

2 Lagrange Multiplier Formulation of Variational Principle

Consider the same simply-connected, bounded $\Omega \subset \mathbb{R}^3$ (similar in $\mathbb{R}^2$) domain with $\boldsymbol{\omega} =< x, y, z >\in \Omega$, given the original transformation $\boldsymbol{\phi}_o$, and the target transformation $\boldsymbol{\phi}_t \in H_0^2(\Omega)$, we look for a diffeomorphic transformation $\boldsymbol{\phi} = \boldsymbol{\phi_m} \circ \boldsymbol{\phi}_o = \boldsymbol{\phi_m}(\boldsymbol{\phi}_o) \in H_0^2(\Omega)$ as close to $\boldsymbol{\phi}_t$, where $\boldsymbol{\phi_m}$ is an intermediate transformation that left-translates $\boldsymbol{\phi}_o$ to $\boldsymbol{\phi}$, such that, the objective loss functional of the Lagrange Multiplier formulation of VP, $\mathcal{L}$ is minimized over the Lagrange multipliers λ_f and $\boldsymbol{\lambda_g} =< \lambda_{\boldsymbol{g}1}, \lambda_{\boldsymbol{g}2}, \lambda_{\boldsymbol{g}3} >$ on Ω, which $\mathcal{L}$ is defined as follows:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\phi}, \lambda_f, \boldsymbol{\lambda_g}, f, \boldsymbol{g}) =& \frac{1}{2}\int_\Omega \|\boldsymbol{\phi} - \boldsymbol{\phi}_t\|^2 d\boldsymbol{\omega} + \int_\Omega [\lambda_f(\det\nabla\boldsymbol{\phi} - f)] d\boldsymbol{\omega} + \frac{1}{2}\int_\Omega f^2 d\boldsymbol{\omega} \\ &+ \int_\Omega [\boldsymbol{\lambda_g} \cdot (\nabla \times \boldsymbol{\phi} - \boldsymbol{g})] d\boldsymbol{\omega} + \frac{1}{2}\int_\Omega |\boldsymbol{g}|^2 d\boldsymbol{\omega}, \end{aligned} \tag{4}$$

where $f(\boldsymbol{\omega}) > 0$, $\boldsymbol{g}(\boldsymbol{\omega})$ are control functions on Ω satisfy, respectively, $\int_\Omega f(\boldsymbol{\omega})d\boldsymbol{\omega} = |\Omega|$ and $\nabla \cdot \boldsymbol{g} = 0$, so that f acts like JD and the second requires $\boldsymbol{g}$ like curl. Here, we denote the Sum of the Squared Differences,

$$\mathrm{SSD}_2(\boldsymbol{\phi}) = \frac{1}{2}\int_\Omega \|\boldsymbol{\phi} - \boldsymbol{\phi}_t\|^2 d\boldsymbol{\omega}. \tag{5}$$

The problem formulated this way is to check whether the desired grid can be attained with a given grid, but follows the path requirement of the by the control functions.

The Necessary Conditions of the Lagrange Multipliers To obtain the necessary conditions for minimizing (4), the variational gradients of $\mathcal{L}$ with respect to the variations of Lagrange multipliers $\delta\lambda_f$, $\delta\boldsymbol{\lambda_g}$, the control functions δf, $\delta\boldsymbol{g}$, the diffeomorphic transformation $\delta\boldsymbol{\phi}$ are needed. Let $\epsilon \in \mathbb{R}$, it is derived as follows, firstly,

$$\forall \delta\lambda_f(\boldsymbol{\omega}) \text{ on } \Omega, \delta\mathcal{L} = \frac{d}{d\epsilon}\Big|_{\epsilon=0}\mathcal{L}(\lambda_f+\epsilon\delta\lambda_f) = \int_\Omega [(\delta\lambda_f)(\det\nabla\boldsymbol{\phi}-f)]d\boldsymbol{\omega}, \Rightarrow \frac{\partial\mathcal{L}}{\partial\lambda_f} = \det\nabla\boldsymbol{\phi}-f.$$

$$\forall \delta\boldsymbol{\lambda_g} \text{ on } \Omega, \quad \delta\mathcal{L} = \frac{d}{d\epsilon}\Big|_{\epsilon=0}\mathcal{L}(\boldsymbol{\lambda_g} + \epsilon\delta\boldsymbol{\lambda_g}) = \int_\Omega [\delta\boldsymbol{\lambda_g}\cdot(\nabla\times\boldsymbol{\phi} - \boldsymbol{g})]d\boldsymbol{\omega},$$
$$\Rightarrow \frac{\partial\mathcal{L}}{\partial\boldsymbol{\lambda_g}} < \frac{\partial\mathcal{L}}{\partial\lambda_{\boldsymbol{g}1}}, \frac{\partial\mathcal{L}}{\partial\lambda_{\boldsymbol{g}2}}, \frac{\partial\mathcal{L}}{\partial\lambda_{\boldsymbol{g}3}} >$$
$$=< (\phi_{3y} - \phi_{2z} - g_1), (-\phi_{3x} + \phi_{1z} - g_2), (\phi_{2x} - \phi_{1y} - g_3) >= \nabla\times\boldsymbol{\phi} - \boldsymbol{g}.$$

So, the State Equations are obtained by

$$\frac{\partial\mathcal{L}}{\partial\lambda_f} = 0 \quad \text{and} \quad \frac{\partial\mathcal{L}}{\partial\boldsymbol{\lambda_g}} = \boldsymbol{0}, \quad \Rightarrow \begin{cases} f = \det\nabla\boldsymbol{\phi}, \\ \boldsymbol{g} = \nabla\times\boldsymbol{\phi}. \end{cases} \tag{6}$$

Secondly,

$$\forall \delta f(\boldsymbol{\omega}) \text{ on } \Omega, \quad \delta\mathcal{L} = \frac{d}{d\epsilon}\Big|_{\epsilon=0}\mathcal{L}(f + \epsilon\delta f) = \int_\Omega [(\delta f)(-\lambda_f + f)]d\boldsymbol{\omega}, \quad \Rightarrow \frac{\partial\mathcal{L}}{\partial f} = -\lambda_f + f,$$

$$\forall \delta\boldsymbol{g}(\boldsymbol{\omega}) \text{ on } \Omega, \quad \delta\mathcal{L} = \frac{d}{d\epsilon}\Big|_{\epsilon=0}\mathcal{L}(\boldsymbol{g} + \epsilon\delta\boldsymbol{g}) = \int_\Omega [\delta\boldsymbol{g}\cdot(-\boldsymbol{\lambda_g} + \boldsymbol{g})]d\boldsymbol{\omega},$$
$$\Rightarrow \frac{\partial\mathcal{L}}{\partial\boldsymbol{g}} =< \frac{\partial\mathcal{L}}{\partial g_1}, \frac{\partial\mathcal{L}}{\partial g_2}, \frac{\partial\mathcal{L}}{\partial g_3} >=< -\lambda_1 + g_1, -\lambda_2 + g_2, -\lambda_3 + g_3 >= -\boldsymbol{\lambda_g} + \boldsymbol{g},$$

then, the Optimality Conditions are obtained by

$$\frac{\partial\mathcal{L}}{\partial f} = 0 \quad \text{and} \quad \frac{\partial\mathcal{L}}{\partial\boldsymbol{g}} = \boldsymbol{0}, \quad \Rightarrow \begin{cases} \lambda_f = f, \\ \boldsymbol{\lambda_g} = \boldsymbol{g}. \end{cases} \tag{7}$$

Lastly,

$$\forall \delta\boldsymbol{\phi_m}(\boldsymbol{\omega}) \in H_0^2(\Omega), \quad \delta\mathcal{L} = \frac{d}{d\epsilon}\Big|_{\epsilon=0}\mathcal{L}(\boldsymbol{\phi_m} + \epsilon\delta\boldsymbol{\phi_m}) = \int_\Omega [(\boldsymbol{\phi_m}\circ\boldsymbol{\phi}_o - \boldsymbol{\phi}_t)\cdot\delta\boldsymbol{\phi_m}]d\boldsymbol{\omega}$$
$$+ \int_\Omega [\lambda_f\delta\det\nabla(\boldsymbol{\phi_m}\circ\boldsymbol{\phi}_o)]d\boldsymbol{\omega} + \int_\Omega [\boldsymbol{\lambda_g}\cdot\delta\nabla\times(\boldsymbol{\phi_m}\circ\boldsymbol{\phi}_o)]d\boldsymbol{\omega} = \boldsymbol{I} + \boldsymbol{II} + \boldsymbol{III},$$

$$\boldsymbol{I} = \int_{\Omega} [(\boldsymbol{\phi_m} \circ \boldsymbol{\phi}_o - \boldsymbol{\phi}_t) \cdot \delta \boldsymbol{\phi_m}] d\boldsymbol{\omega}; \tag{8}$$

$$\begin{aligned}
\boldsymbol{II} &= \int_{\Omega} [\lambda_f \delta \det\nabla(\boldsymbol{\phi_m} \circ \boldsymbol{\phi}_o)] d\boldsymbol{\omega} \\
&= \int_{\Omega} [\lambda_f \delta(\det\nabla \boldsymbol{\phi_m} \det\nabla \boldsymbol{\phi}_o)] d\boldsymbol{\omega} = \int_{\Omega} [\lambda_f \det\nabla \boldsymbol{\phi}_o (\delta \det\nabla \boldsymbol{\phi_m})] d\boldsymbol{\omega} \\
&= \int_{\Omega} [\lambda_f \det\nabla \boldsymbol{\phi}_o \delta(\phi_{\boldsymbol{m}1x}(\phi_{\boldsymbol{m}2y}\phi_{\boldsymbol{m}3z} - \phi_{\boldsymbol{m}2z}\phi_{\boldsymbol{m}3y}) - \phi_{\boldsymbol{m}1y}(\phi_{\boldsymbol{m}2x}\phi_{\boldsymbol{m}3z} - \phi_{\boldsymbol{m}2z}\phi_{\boldsymbol{m}3x}) \\
&+ \phi_{\boldsymbol{m}1z}(\phi_{\boldsymbol{m}2x}\phi_{\boldsymbol{m}3y} - \phi_{\boldsymbol{m}2y}\phi_{\boldsymbol{m}3x}))] d\boldsymbol{\omega} \\
&= \int_{\Omega} [\lambda_f \det\nabla \boldsymbol{\phi}_o (\delta\phi_{\boldsymbol{m}1x}\phi_{\boldsymbol{m}2y}\phi_{\boldsymbol{m}3z} + \phi_{\boldsymbol{m}1x}\delta\phi_{\boldsymbol{m}2y}\phi_{\boldsymbol{m}3z} + \phi_{\boldsymbol{m}1x}\phi_{\boldsymbol{m}2y}\delta\phi_{\boldsymbol{m}3z} - \delta\phi_{\boldsymbol{m}1x}\phi_{\boldsymbol{m}2z}\phi_{\boldsymbol{m}3y} \\
&- \phi_{\boldsymbol{m}1x}\delta\phi_{\boldsymbol{m}2z}\phi_{\boldsymbol{m}3y} - \phi_{\boldsymbol{m}1x}\phi_{\boldsymbol{m}2z}\delta\phi_{\boldsymbol{m}3y} - \delta\phi_{\boldsymbol{m}1y}\phi_{\boldsymbol{m}3z}\phi_{\boldsymbol{m}2x} - \phi_{\boldsymbol{m}1y}\delta\phi_{\boldsymbol{m}3z}\phi_{\boldsymbol{m}2x} - \phi_{\boldsymbol{m}1y}\phi_{\boldsymbol{m}3z}\delta\phi_{\boldsymbol{m}2x} \\
&+ \delta\phi_{\boldsymbol{m}1y}\phi_{\boldsymbol{m}3x}\phi_{\boldsymbol{m}2z} + \phi_{\boldsymbol{m}1y}\delta\phi_{\boldsymbol{m}3x}\phi_{\boldsymbol{m}2z} + \phi_{\boldsymbol{m}1y}\phi_{\boldsymbol{m}3x}\delta\phi_{\boldsymbol{m}2z} + \delta\phi_{\boldsymbol{m}1z}\phi_{\boldsymbol{m}2x}\phi_{\boldsymbol{m}3y} + \phi_{\boldsymbol{m}1z}\delta\phi_{\boldsymbol{m}2x}\phi_{\boldsymbol{m}3y} \\
&+ \phi_{\boldsymbol{m}1z}\phi_{\boldsymbol{m}2x}\delta\phi_{\boldsymbol{m}3y} - \delta\phi_{\boldsymbol{m}1z}\phi_{\boldsymbol{m}2y}\phi_{\boldsymbol{m}3x} - \phi_{\boldsymbol{m}1z}\delta\phi_{\boldsymbol{m}2y}\phi_{\boldsymbol{m}3x} - \phi_{\boldsymbol{m}1z}\phi_{\boldsymbol{m}2y}\delta\phi_{\boldsymbol{m}3x})] d\boldsymbol{\omega} \\
&= \int_{\Omega} [(\lambda_f \det\nabla \boldsymbol{\phi}_o \begin{pmatrix} \phi_{\boldsymbol{m}2y}\phi_{\boldsymbol{m}3z} - \phi_{\boldsymbol{m}3y}\phi_{\boldsymbol{m}2z} \\ \phi_{\boldsymbol{m}3x}\phi_{\boldsymbol{m}2z} - \phi_{\boldsymbol{m}2x}\phi_{\boldsymbol{m}3z} \\ \phi_{\boldsymbol{m}2x}\phi_{\boldsymbol{m}3y} - \phi_{\boldsymbol{m}2y}\phi_{\boldsymbol{m}3x} \end{pmatrix}) \cdot \nabla\delta\phi_{\boldsymbol{m}1} \\
&\quad + (\lambda_f \det\nabla \boldsymbol{\phi}_o \begin{pmatrix} \phi_{\boldsymbol{m}3y}\phi_{\boldsymbol{m}1z} - \phi_{\boldsymbol{m}1y}\phi_{\boldsymbol{m}3z} \\ \phi_{\boldsymbol{m}1x}\phi_{\boldsymbol{m}3z} - \phi_{\boldsymbol{m}1z}\phi_{\boldsymbol{m}3x} \\ \phi_{\boldsymbol{m}3x}\phi_{\boldsymbol{m}1y} - \phi_{\boldsymbol{m}1x}\phi_{\boldsymbol{m}3y} \end{pmatrix}) \cdot \nabla\delta\phi_{\boldsymbol{m}2} \\
&\quad + (\lambda_f \det\nabla \boldsymbol{\phi}_o \begin{pmatrix} \phi_{\boldsymbol{m}1y}\phi_{\boldsymbol{m}2z} - \phi_{\boldsymbol{m}2y}\phi_{\boldsymbol{m}1z} \\ \phi_{\boldsymbol{m}2x}\phi_{\boldsymbol{m}1z} - \phi_{\boldsymbol{m}1x}\phi_{\boldsymbol{m}2z} \\ \phi_{\boldsymbol{m}1x}\phi_{\boldsymbol{m}2y} - \phi_{\boldsymbol{m}2x}\phi_{\boldsymbol{m}1y} \end{pmatrix}) \cdot \nabla\delta\phi_{\boldsymbol{m}3}] d\boldsymbol{\omega} \\
&= \int_{\Omega} [\boldsymbol{A}_1 \cdot \nabla\delta\phi_{\boldsymbol{m}1} + \boldsymbol{A}_2 \cdot \nabla\delta\phi_{\boldsymbol{m}2} + \boldsymbol{A}_3 \cdot \nabla\delta\phi_{\boldsymbol{m}3}] d\boldsymbol{\omega}, \text{ where } \boldsymbol{A}_i\text{'s denote the big vectors.}
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{III} &= \int_{\Omega} [\boldsymbol{\lambda_g} \cdot \delta\nabla \times (\boldsymbol{\phi_m} \circ \boldsymbol{\phi}_o)] d\boldsymbol{\omega} = \int_{\Omega} [\boldsymbol{\lambda_g} \cdot \delta \begin{pmatrix} \nabla\phi_{\boldsymbol{m}3} \cdot (\boldsymbol{\phi}_o)_y - \nabla\phi_{\boldsymbol{m}2} \cdot (\boldsymbol{\phi}_o)_z \\ -\nabla\phi_{\boldsymbol{m}3} \cdot (\boldsymbol{\phi}_o)_x + \nabla\phi_{\boldsymbol{m}1} \cdot (\boldsymbol{\phi}_o)_z \\ \nabla\phi_{\boldsymbol{m}2} \cdot (\boldsymbol{\phi}_o)_x - \nabla\phi_{\boldsymbol{m}1} \cdot (\boldsymbol{\phi}_o)_y \end{pmatrix}] d\boldsymbol{\omega} \\
&= \int_{\Omega} [\boldsymbol{\lambda_g} \cdot \begin{pmatrix} \delta\phi_{\boldsymbol{m}3x}\phi_{o1y} + \delta\phi_{\boldsymbol{m}3y}\phi_{o2y} + \delta\phi_{\boldsymbol{m}3z}\phi_{o3y} - \delta\phi_{\boldsymbol{m}2x}\phi_{o1z} - \delta\phi_{\boldsymbol{m}2y}\phi_{o2z} - \delta\phi_{\boldsymbol{m}2z}\phi_{o3z} \\ -\delta\phi_{\boldsymbol{m}3x}\phi_{o1x} - \delta\phi_{\boldsymbol{m}3y}\phi_{o2x} - \delta\phi_{\boldsymbol{m}3z}\phi_{o3x} + \delta\phi_{\boldsymbol{m}1x}\phi_{o1z} + \delta\phi_{\boldsymbol{m}1y}\phi_{o2z} + \delta\phi_{\boldsymbol{m}1z}\phi_{o3z} \\ \delta\phi_{\boldsymbol{m}2x}\phi_{o1x} + \delta\phi_{\boldsymbol{m}2y}\phi_{o2x} + \delta\phi_{\boldsymbol{m}2z}\phi_{o3x} - \delta\phi_{\boldsymbol{m}1x}\phi_{o1y} - \delta\phi_{\boldsymbol{m}1y}\phi_{o2y} - \delta\phi_{\boldsymbol{m}1z}\phi_{o3y} \end{pmatrix}] d\boldsymbol{\omega} \\
&= \int_{\Omega} [\begin{pmatrix} \lambda_{\boldsymbol{g}2}\phi_{o1z} - \lambda_{\boldsymbol{g}3}\phi_{o1y} \\ \lambda_{\boldsymbol{g}2}\phi_{o2z} - \lambda_{\boldsymbol{g}3}\phi_{o2y} \\ \lambda_{\boldsymbol{g}2}\phi_{o3z} - \lambda_{\boldsymbol{g}3}\phi_{o3y} \end{pmatrix} \cdot \begin{pmatrix} \delta(\phi_{\boldsymbol{m}1})_x \\ \delta(\phi_{\boldsymbol{m}1})_y \\ \delta(\phi_{\boldsymbol{m}1})_z \end{pmatrix} + \begin{pmatrix} -\lambda_{\boldsymbol{g}1}\phi_{o1z} + \lambda_{\boldsymbol{g}3}\phi_{o1x} \\ -\lambda_{\boldsymbol{g}1}\phi_{o2z} + \lambda_{\boldsymbol{g}3}\phi_{o2x} \\ -\lambda_{\boldsymbol{g}1}\phi_{o3z} + \lambda_{\boldsymbol{g}3}\phi_{o3x} \end{pmatrix} \cdot \begin{pmatrix} \delta(\phi_{\boldsymbol{m}2})_x \\ \delta(\phi_{\boldsymbol{m}2})_y \\ \delta(\phi_{\boldsymbol{m}2})_z \end{pmatrix} \\
&+ \begin{pmatrix} \lambda_{\boldsymbol{g}1}\phi_{o1y} - \lambda_{\boldsymbol{g}2}\phi_{o1x} \\ \lambda_{\boldsymbol{g}1}\phi_{o2y} - \lambda_{\boldsymbol{g}2}\phi_{o2x} \\ \lambda_{\boldsymbol{g}1}\phi_{o3y} - \lambda_{\boldsymbol{g}2}\phi_{o3x} \end{pmatrix} \cdot \begin{pmatrix} \delta(\phi_{\boldsymbol{m}3})_x \\ \delta(\phi_{\boldsymbol{m}3})_y \\ \delta(\phi_{\boldsymbol{m}3})_z \end{pmatrix}] d\boldsymbol{\omega} = \int_{\Omega} [\boldsymbol{B}_1 \cdot \nabla\delta\phi_{\boldsymbol{m}1} + \boldsymbol{B}_2 \cdot \nabla\delta\phi_{\boldsymbol{m}2} + \boldsymbol{B}_3 \cdot \nabla\delta\phi_{\boldsymbol{m}3}] d\boldsymbol{\omega}.
\end{aligned}$$

By Green's Identities, it is carried to,

$$\begin{aligned}
\boldsymbol{II} + \boldsymbol{III} &= \int_{\Omega} [\boldsymbol{A}_1 \cdot \nabla \delta\phi_{\boldsymbol{m}1} + \boldsymbol{A}_2 \cdot \nabla \delta\phi_{\boldsymbol{m}2} + \boldsymbol{A}_3 \cdot \nabla \delta\phi_{\boldsymbol{m}3} \\
&\quad + \boldsymbol{B}_1 \cdot \nabla \delta\phi_{\boldsymbol{m}1} + \boldsymbol{B}_2 \cdot \nabla \delta\phi_{\boldsymbol{m}2} + \boldsymbol{B}_3 \cdot \nabla \delta\phi_{\boldsymbol{m}3}] d\boldsymbol{\omega} \\
&= \int_{\Omega} [(\boldsymbol{A}_1 + \boldsymbol{B}_1) \cdot \nabla \delta\phi_{\boldsymbol{m}1} + (\boldsymbol{A}_2 + \boldsymbol{B}_2) \cdot \nabla \delta\phi_{\boldsymbol{m}2} + (\boldsymbol{A}_3 + \boldsymbol{B}_3) \cdot \nabla \delta\phi_{\boldsymbol{m}3}] d\boldsymbol{\omega} \\
&= \int_{\Omega} [-\nabla \cdot (\boldsymbol{A}_1 + \boldsymbol{B}_1) \delta\phi_{\boldsymbol{m}1} - \nabla \cdot (\boldsymbol{A}_2 + \boldsymbol{B}_2) \delta\phi_{\boldsymbol{m}2} - \nabla \cdot (\boldsymbol{A}_3 + \boldsymbol{B}_3) \delta\phi_{\boldsymbol{m}3}] d\boldsymbol{\omega} \\
&= \int_{\Omega} [-\begin{pmatrix} \nabla \cdot (\boldsymbol{A}_1 + \boldsymbol{B}_1) \\ \nabla \cdot (\boldsymbol{A}_2 + \boldsymbol{B}_2) \\ \nabla \cdot (\boldsymbol{A}_3 + \boldsymbol{B}_3) \end{pmatrix} \cdot \delta\boldsymbol{\phi}_{\boldsymbol{m}}] d\boldsymbol{\omega},
\end{aligned}$$

where $\boldsymbol{B}_i$, $i = 1, 2, 3$, denote the long vectors of $\boldsymbol{III}$. By (8) and (2), it goes to,

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}_{\boldsymbol{m}}} = (\boldsymbol{\phi}_{\boldsymbol{m}} \circ \boldsymbol{\phi}_o - \boldsymbol{\phi}_t) - \begin{pmatrix} \nabla \cdot (\boldsymbol{A}_1 + \boldsymbol{B}_1) \\ \nabla \cdot (\boldsymbol{A}_2 + \boldsymbol{B}_2) \\ \nabla \cdot (\boldsymbol{A}_3 + \boldsymbol{B}_3) \end{pmatrix}, \tag{9}$$

therefore, the Adjoin Equations are obtained through

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}_{\boldsymbol{m}}} = \boldsymbol{0} \quad \Rightarrow \boldsymbol{\phi}_{\boldsymbol{m}} \circ \boldsymbol{\phi}_o - \boldsymbol{\phi}_t - \begin{pmatrix} \nabla \cdot (\boldsymbol{A}_1 + \boldsymbol{B}_1) \\ \nabla \cdot (\boldsymbol{A}_2 + \boldsymbol{B}_2) \\ \nabla \cdot (\boldsymbol{A}_3 + \boldsymbol{B}_3) \end{pmatrix} = \boldsymbol{0}. \tag{10}$$

Gradient-Descent Based Algorithm for LM-VP A gradient descent pseudo-code is summarized below. $\triangle t$ is the step-size of gradient. Major computational costs occur in solving *Poisson* equations by a Fast Fourier Transform *Poisson* solver (FFT). Minor incremental computations happen in the function composition by interpolation of step 7. Define $ratio = \mathrm{SSD}_{final}/\mathrm{SSD}_{initial}$ and let $\boldsymbol{F}$ be the control function to be determined.

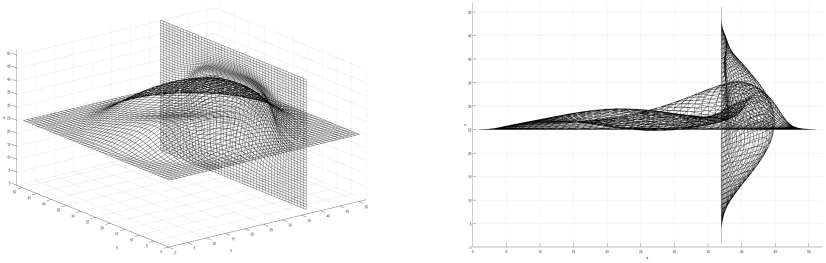
Algorithm 1 $[\boldsymbol{\phi}, \boldsymbol{\phi}_{\boldsymbol{m}}] = \text{LM-VP}(\boldsymbol{\phi}_o, \boldsymbol{\phi}_t)$

- 1: initialize $\boldsymbol{\phi} = \boldsymbol{\phi}_o$ and the gradient step-size $\triangle t$;
- 2: while stopping criteria is unmet
  - 3: if *better*
    - 4: compute $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}_{\boldsymbol{m}}}$ from (9);
  - 5: update $\boldsymbol{\phi}_{\boldsymbol{m}\_new} = \boldsymbol{\phi}_{\boldsymbol{m}} - \triangle t * \frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}_{\boldsymbol{m}}}$;
  - 6: update $\boldsymbol{\phi} = \boldsymbol{\phi}_{\boldsymbol{m}\_new}(\boldsymbol{\phi}_o)$ by interpolation;
  - 7: compute $\mathrm{SSD}_3(\boldsymbol{\phi})$;
  - 8: if $\mathrm{SSD}_3(\boldsymbol{\phi})$ decrease,
    - 9: *better*;
    - 10: $\triangle t = \triangle t * t_{up}$ (For example, set $t_{up} = 1.01$ and $t_{down} = 0.99$);
    - 11: $\boldsymbol{\phi}_{\boldsymbol{m}} = \boldsymbol{\phi}_{\boldsymbol{m}\_new}$;

    else
    - 12: *better* false;
    - 13: $\triangle t = \triangle t * t_{down}$.

2.1 3D Grid Reconstruction Comparison between the Revised and Lagrange-Multiplier VPs

All the experiments are implemented with MatLab codes on a laptop with Intel Core i5-7300HQ Processor, 16 GB RAM and NVIDA GeForce GTX 1050Ti GPU, except indicated otherwise. In order to see a comparable test between the revised-VP and the proposed LM-VP, we repeated the same 3D simulation in [ICGG, Zhou], which it shows that LM-VP is capable of finding the desired solutions as the revised-VP does. The 3D example goes as the following, given a 3D grid $\boldsymbol{\Phi}$ in Black, in Fig. 1, $\boldsymbol{\Phi}$ is manually built by multiple times of applying cutoff rotation and partial displacement over Ω such that $\det\nabla\boldsymbol{\Phi} > 0.1734$ holds which $\boldsymbol{\Phi}$ is a non-folding grid. Due to visualization challenges in 3D grids, all the example grids are plotted with only the 25-th frame on z-axis and the 37-th frame on x-axis. Define $f_t = \det\nabla\boldsymbol{\Phi}$ and $\boldsymbol{g}_t = \nabla \times \boldsymbol{\Phi}$ as the prescribed JD and curl.



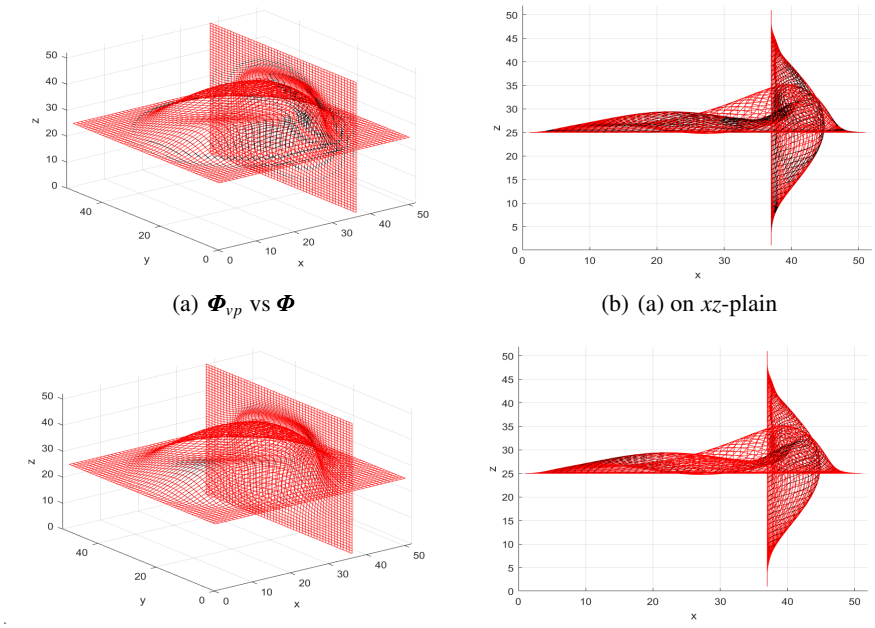
(a) $\boldsymbol{\Phi}$ (b) (a) on xz-plain

Fig. 1: Given ground truth (GT)

Fig.2(a-b) is the solution by VP, $\boldsymbol{\Phi}_{vp}$, and is superposed on the GT, $\boldsymbol{\Phi}$. It can be visually seen in Fig.2(b) that Red grid lines of $\boldsymbol{\Phi}_{vp}$ do not line-up well with $\boldsymbol{\Phi}$ in Black grid, i.e., a more accurate solution would have covered more Black grid. Comparing to the solution of revised VP, $\boldsymbol{\Phi}_{lm\text{-}vp}$, in Fig.2(c-d), that also is superposed on $\boldsymbol{\Phi}$, in Fig.2(c-d), there is only a small portion of the Black grid stay uncovered by $\boldsymbol{\Phi}_{lm\text{-}vp}$ which indicates revised VP provides much better solutions over the original VP. This observation is confirmed by measurements in the following Table.1. It also recorded the revised VP had reached the *ratio*-tolerance with less iterations and computational time. SSD *ratio* = $\text{SSD}_{final}/\text{SSD}_{initial}$ is defined in the table.

Table 1: Performance of Fig.2

Solution	Ω	SSD *ratio*	sec	iteration	max differences of Fig. 2(a,c)		
					$\lvert\det\nabla\boldsymbol{\phi} - f_t\rvert$	$\lVert\nabla\times\boldsymbol{\phi} - \boldsymbol{g}_t\rVert_2$	$\lVert\boldsymbol{\phi} - \boldsymbol{\Phi}\rVert_2$
$\boldsymbol{\phi} = \boldsymbol{\Phi}_{vp}$	$[1,51]^3$	$2.0 * 10^{-4}$	788.21	576	0.0104	0.0136	0.0403
$\boldsymbol{\phi} = \boldsymbol{\Phi}_{lm\text{-}vp}$	$[1,51]^3$	$9.0 * 10^{-6}$	107.02	127	0.0013	$5.9 * 10^{-4}$	0.0017



(a) $\boldsymbol{\Phi}_{vp}$ vs $\boldsymbol{\Phi}$ (b) (a) on xz-plain

(c) $\boldsymbol{\Phi}_{lm\_vp}$ vs $\boldsymbol{\Phi}$ (d) (c) on xz-plain

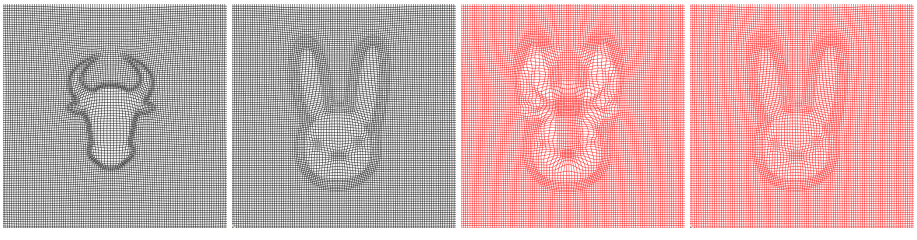
Fig. 2: Solutions by revised VP, $\boldsymbol{\Phi}_{vp}$, and LM-VP, $\boldsymbol{\Phi}_{lm\_vp}$, vs GT, $\boldsymbol{\Phi}$

This example demonstrates a three-fold improvement of the revised VP: (i) the revised VP is effective and capable in generating 3D grids with prescribed JD and curl; (ii) the revised VP finds more accurate solutions in terms of the prescription of JD and curl (it is more obvious in 3D examples) compared to the original VP; (ii) the revised VP reaches the desired tolerance faster than the original VP (in fact, the computational cost of the revised VP for an effective iteration is very close to the original VP, where main extra computations occur on the interpolation of step 7 in the provided Alg.1).

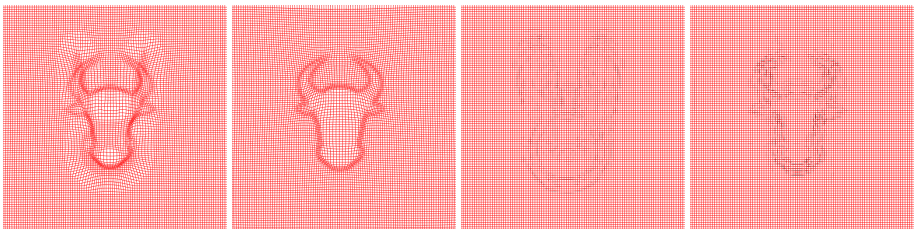
2.2 Computational Inverse Consistency and Transitivity

In this example, given another bull grid, $\boldsymbol{B}$, the inverse consistency between $\boldsymbol{B}$ and $\boldsymbol{R}$ are demonstrated in Figure 3. Moreover, if one more cat grid, $\boldsymbol{C}$, is inserted between $\boldsymbol{B}$ and $\boldsymbol{R}$, the transitivity of the path $\boldsymbol{B} \rightarrow \boldsymbol{R}$ and the path from $\boldsymbol{B} \rightarrow \boldsymbol{C}$ then $\boldsymbol{C} \rightarrow \boldsymbol{R}$ is demonstrated in Figure 4.

Figure 3. (d) and (f) are presenting the solutions, $\boldsymbol{D}_1 \circ \boldsymbol{B}$ and $\boldsymbol{D}_2 \circ \boldsymbol{R}$, by LM-VP, in red grid lines, are very close to the targets grids, $\boldsymbol{R}$ and $\boldsymbol{B}$, in black grids lines, respectively. Figure 3.(g) and (h) are showing that the forward path $\boldsymbol{D}_1 : \boldsymbol{B} \rightarrow \boldsymbol{R}$ and the backward path $\boldsymbol{D}_2 : \boldsymbol{R} \rightarrow \boldsymbol{B}$ indeed compensate each other closely to the identity map. $\boldsymbol{D}_2 \circ \boldsymbol{D}_1$ and $\boldsymbol{D}_1 \circ \boldsymbol{D}_2$, in red grid lines, are superimposed on $\boldsymbol{id}$, in black grid lines.



(a) $\boldsymbol{B}$ (b) $\boldsymbol{R}$ (c) $\boldsymbol{D}_1 : \boldsymbol{B} \to \boldsymbol{R}$ (d) $\boldsymbol{D}_1 \circ \boldsymbol{B}$ vs $\boldsymbol{R}$



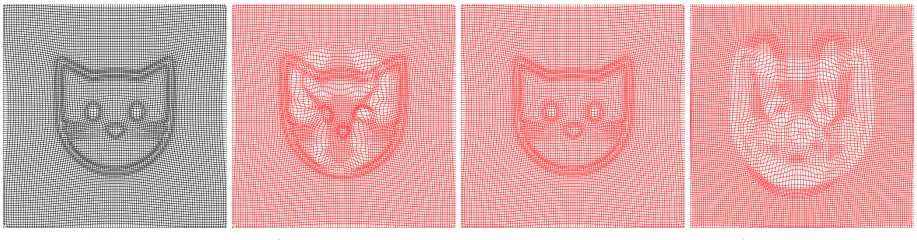
(e) $\boldsymbol{D}_2 : \boldsymbol{R} \to \boldsymbol{B}$ (f) $\boldsymbol{D}_2 \circ \boldsymbol{R}$ vs $\boldsymbol{B}$ (g) $\boldsymbol{D}_2 \circ \boldsymbol{D}_1$ vs $\boldsymbol{id}$ (h) $\boldsymbol{D}_1 \circ \boldsymbol{D}_1$ vs $\boldsymbol{id}$

Fig. 3: Inverse Consistency

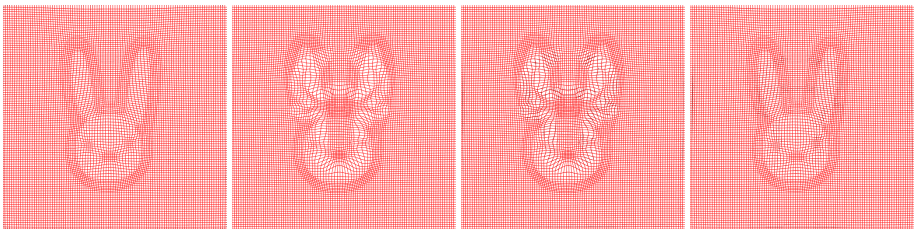
Next, in Figure 4.(b) and (d), are showing the paths of $\boldsymbol{D}_3 : \boldsymbol{B} \to \boldsymbol{C}$ and $\boldsymbol{D}_4 : \boldsymbol{C} \to \boldsymbol{R}$, whose combination $\boldsymbol{D}_{4\circ 3}$ is presented in (f). Figure 4.(g) indicates that the combined path, $\boldsymbol{D}_{4\circ 3}$, in red grid lines, did the same job as $\boldsymbol{D}_1$ of Figure 3.(c), in black grid lines. Finally, Figure 4.(h) shows that both $\boldsymbol{D}_{4\circ 3}$ and $\boldsymbol{D}_1$ leads $\boldsymbol{B}$ to the same end that is very close to $\boldsymbol{R}$. This shows the solutions by VP have the transitivity property.

Table 2: Performance of Fig.3

Orientation	Ω	*ratio*	second	iteration	max differences of Fig.3(d, f)		
					$\lvert\det\nabla\boldsymbol{\phi} - f_t\rvert$	$\lvert\nabla\times\boldsymbol{\phi} - \boldsymbol{g}_t\rvert$	$\lVert\boldsymbol{\phi} - \boldsymbol{\phi}_t\rVert_2$
$\boldsymbol{D}_1 : \boldsymbol{B} \to \boldsymbol{R}$	$[1, 101]^2$	$4.1916 * 10^{-5}$	21.28	1350	0.0460	0.0255	0.0249
$\boldsymbol{D}_2 : \boldsymbol{R} \to \boldsymbol{B}$	$[1, 101]^2$	$3.0864 * 10^{-5}$	21.97	1256	0.0460	0.0241	0.0249



(a) $\boldsymbol{C}$ (b) $\boldsymbol{D}_3 : \boldsymbol{B} \to \boldsymbol{C}$ (c) $\boldsymbol{D}_3 \circ \boldsymbol{B}$ vs $\boldsymbol{C}$ (d) $\boldsymbol{D}_4 : \boldsymbol{C} \to \boldsymbol{R}$



(e) $\boldsymbol{D}_4 \circ \boldsymbol{C}$ vs $\boldsymbol{R}$ (f) $\boldsymbol{D}_{4\circ 3} \equiv \boldsymbol{D}_4 \circ \boldsymbol{D}_3$ (g) $\boldsymbol{D}_{4\circ 3}$ vs $\boldsymbol{D}_1$ (h) $\boldsymbol{D}_{4\circ 3} \circ \boldsymbol{B}$ vs $\boldsymbol{D}_1 \circ \boldsymbol{B}$

Fig. 4: Transitivity

Table 3: Performance of Fig.4

Orientation	Ω	*ratio*	second	iteration	max differences of Fig.4(c, e)		
					$\lvert\det\nabla\boldsymbol{\phi} - f_t\rvert$	$\lvert\nabla\times\boldsymbol{\phi} - \boldsymbol{g}_t\rvert$	$\lVert\boldsymbol{\phi} - \boldsymbol{\phi}_t\rVert_2$
$\boldsymbol{D}_3 : \boldsymbol{C} \to \boldsymbol{P}$	$[1, 97]^2$	$1.42 * 10^{-5}$	22.99	1339	0.0332	0.0222	0.0848
$\boldsymbol{D}_4 : \boldsymbol{P} \to \boldsymbol{R}$	$[1, 97]^2$	$1.30 * 10^{-5}$	22.16	1352	0.1967	0.1049	0.1046

3 Discussion: LM-VP inspired Loss for Deep-Learning Nonrigid Image Registration

The above are the detailed analytical works for grid generation problem, it connects to the resulting image registration deformations. Here, the image registration method in [17], VP-control method is described as follows: let I_m be a ***moving*** image is to be registered to a ***fixed*** image I_f on the fixed and bounded domain $(\boldsymbol{\omega} =< x, y, z >\in)\Omega \subset \mathbb{R}^3$, the energy function *Loss* function, SSD_3, is minimized over the form $\boldsymbol{\phi} = \boldsymbol{id} + \boldsymbol{u}$ on Ω with $\boldsymbol{u} = \boldsymbol{0}$ on $\partial\Omega$,

$$\text{SSD}_3(\boldsymbol{\phi}) = \frac{1}{2}\int_\Omega [I_m(\boldsymbol{\phi}) - I_f]^2 d\boldsymbol{\omega} \text{ subjects to } \Delta\boldsymbol{\phi} = \boldsymbol{F}(f, \boldsymbol{g}) \text{ in } \Omega, \tag{11}$$

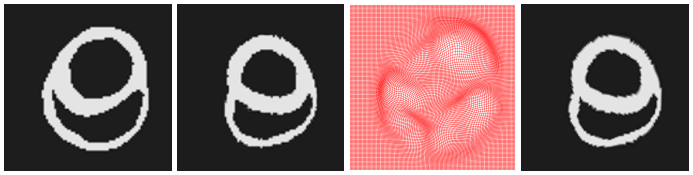
where f and $\boldsymbol{g}$ are the control functions in the sense of VP that mimic the prescribed JD and curl, respectively. In [19], to ensure (11) producing diffeomorphic deformations that is controlled by $J_{min} \in (0, 1)$, it optimizes along $\hat{f}$ where $f := J_{min} + \hat{f}^2$ was defined.

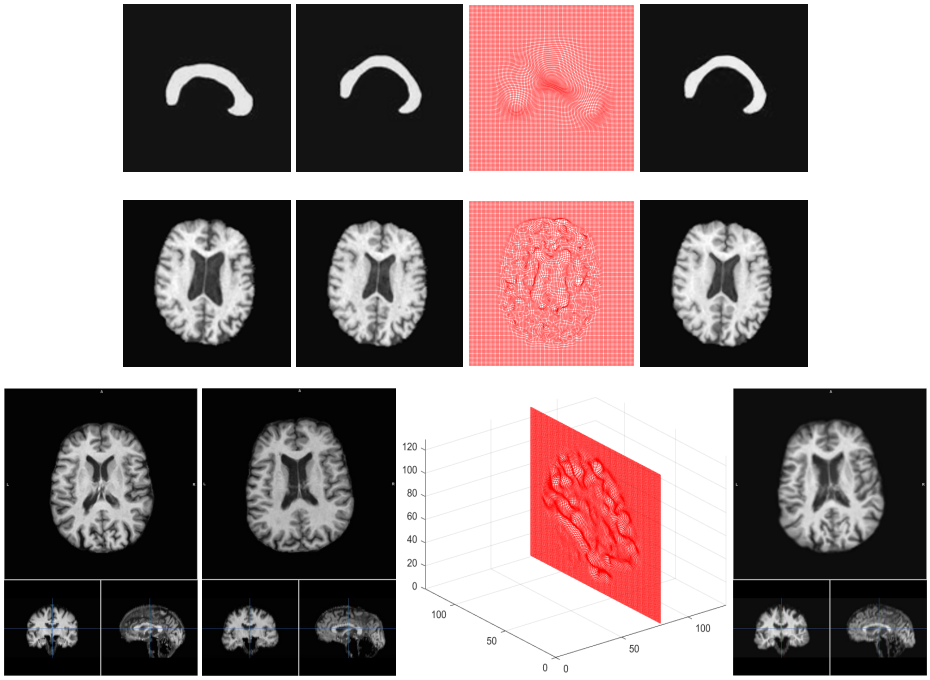
$$\begin{aligned}\mathcal{L}(\boldsymbol{\phi}, \lambda_{\hat{f}}, \boldsymbol{\lambda_g}, \hat{f}, \boldsymbol{g}) = \frac{1}{2}\int_\Omega [I_m(\boldsymbol{\phi}) - I_f]^2 d\boldsymbol{\omega} + \int_\Omega [\lambda_{\hat{f}}(\det\nabla\boldsymbol{\phi} - (J_{min} + \hat{f}^2))] d\boldsymbol{\omega} + \frac{1}{2}\int_\Omega \hat{f}^4 d\boldsymbol{\omega} \\ + \int_\Omega [\boldsymbol{\lambda_g} \cdot (\nabla\times\boldsymbol{\phi} - \boldsymbol{g})] d\boldsymbol{\omega} + \frac{1}{2}\int_\Omega |\boldsymbol{g}|^2 d\boldsymbol{\omega},\end{aligned} \tag{12}$$

where $\hat{f}, \boldsymbol{g}$ are control functions on Ω. This is the proposed loss function for the parallel deep-learning work. So, the detailed analysis is omitted here. Next, some preliminary examples are presented. These are results achieved by traditional gradient descent scheme, no artificial neuro-networks involved.

Table 4: Evaluation of the Proposed Image Registration

e.g.	Ω	SSD_3 *ratio*	second	min($\det\nabla\boldsymbol{\phi}$)	JSC	DICE
`Synthetic holes`	$[1, 128]^2$	0.0279	15.61	0.4076	0.9115	0.9537
`Hippocampus`	$[1, 128]^2$	0.0052	13.50	0.3628	0.9678	0.9836
`2D MRI`	$[1, 128]^2$	0.0605	10.77	0.2540	0.9836	0.9918
`3D MRI`	$[1, 128]^3$	0.2892	535.77	0.0103	0.8666	0.9285





(e) I_m (f) I_f (g) ϕ: 82-th on y-axis (h) $I_m(\phi)$

Fig. 13: Resulting Registration Deformations by the proposed Loss Function

The figure shows that the idea of LM-VP can be transferred to the problem of nonrigid image registration problem. It performs quite well on 2D problems and has potentials to be improved for the 3D cases. Yet, the effectiveness of the proposed method is visualized. Traditional optimization schemes may result in endless tuning of different parameters in balancing each corresponding terms of the loss function, so for the future study, this loss function should be implemented with artificial neuro networks for deep-learning methods.

4 Conclusions

In sum, the LM-VP confirmed that a desired un-entangled grid can be attained with a given un-entangled grid that traces by the Lagrange Multipliers of the control functions of JD and curl. Compared to the earlier version of VP, this LM-VP computes faster and more accurate without the computational burden of solving its embedded PDE system. Based on its idea, we proposed a new training loss function (12) for the problem of nonrigid image registration. The preliminary results based on a gradient descent scheme showed that it is legitimate and promising for further investigation with deep-learning methods.

References

1. M. Bauer, S. Joshi and K. Modin, Diffeomorphic density matching by optimal information transport, SIAM Journal on Imaging Sciences, **8**3 1718âĂŞ1751 (2015)
2. X. Cai, D. Fleitas, B. Jiang and G. Liao, Adaptive grid generation based on the least-squares finite elements method, Computers and Mathematics with Applications, **48**, 1007-1085 (2004)
3. J. Castillo, S. Steinberg, and P. Roach, Parameter estimation in variational grid generation, Appl. Math. Comput., **155**, 155-177 (1988)
4. Y. Chen and X. Ye, Inverse Consistent Deformable Image Registraition. Springer Verlag, 419-440 (2010).
5. X. Chen and G. Liao, New Variational Method of Grid Generation with Prescribed Jacobian determinant and Prescribed Curl, arxiv.org/pdf/1507.03715 (2015)
6. X. Chen and G. Liao, New method of averaging diffeomorphisms based on Jacobian determinant and curl vector, arxiv.org/abs/1611.03946 (2016)
7. B. Dacoragna and J. Moser, On A Partial Differential Equation Involving the Jacobian determinant, Ann.Inst H PoincarÃl', **7**, 1-26 (1990)
8. A. Gruber, Planar Immersions with prescribed Curl and Jacobian Determinant are Unique. Bulletin of the Australian Mathematical Society, 1-6 (2021)
9. M. Grajewski, M. Koster and S. Turek, Mathematical and Numerical Analysis of a Robust and Efficient Grid Deformation Method in the Finite Element Context, *SIAM Journal on Scientific Computing*, **31**2, 1539-1557 (2009)
10. W. Huang and W. Sun, Variational mesh adaptation II: Error estimates and monitor functions, *J. Comput. Phys*, **184**, 619-648 (2003)
11. S. Joshi, B. Davis, M. Jomier and G. Gerig, Unbiased Diffeomorphic Atlas Construction for Computational Anatomy, NeuroImage, **23**, 151-160 (2004)
12. G. Liao, X. Cai, J. Liu, X. Luo, J. Wang, J. Xue, Construction of differentiable transformations, Applied Math.Letters,**22**, 543-1548 (2009)
13. V. Liseikin, Grid Generation Method, Springer Press. (1999)
14. Z. Zhou, X. Chen, X. Cai and G. Liao, Uniqueness of Transformation based on Jacobian Determinant and curl-Vector, arxiv.org/abs/1712.03443 (2017)
15. Zhou, Z., Liao, G.:Recent insights on the Uniqueness Problem of Diffeomorphisms determined by Prescribed Jacobian Determinant and Curl, https://arxiv.org/abs/2207.13053
16. Zhou, Z., Liao, G.: Construction of diffeomorphisms with prescribed Jacobian determinant and curl. In: International Conference on Geometry and Graphics, Proceedings (2022)
17. Zhou, Z., Liao, G.: A novel approach to form Normal Distribution of Medical Image Segmentation based on multiple doctors annotations. In: Proceedings of SPIE 12032, Medical Imaging 2022: Image Processing, p. 1203237 (2022). https://doi.org/10.1117/12.2611973
18. Zhou, Z., Liao, G.: Recent Developments of an Optimal Control Approach to Nonrigid Image Registration. Biomedical Image Registration. WBIR 2022. Lecture Notes in Computer Science, vol 13386. Springer, Cham. https://doi.org/10.1007/978-3-031-11203-4_22
19. Z. Zhou, Image Analysis Based on Differential Operators with Applications to Brain MRIs, *Ph.D. Dissertation*, University of Texas at Arlington (2019)
20. Y. Zhu, Z. Zhou, G Liao, K. Yuan: New loss functions for medical image registration based on Voxelmorph, Medical Imaging 2020: Image Processing 11313, 596-603