Fast 4D Elastic Group-Wise Image Registration. Convolutional Interpolation Revisited

Rosa-María Menchón-Lara[1 a], Javier Royuela-del-Val[b], Federico Simmross-Wattenberg[a], Pablo Casaseca-de-la-Higuera[a], Marcos Antonio Martín-Fernández[a], Carlos Alberola-López[a]

[a]*Laboratorio de Procesado de Imagen. ETSI de Telecomunicación. Universidad de Valladolid. Valladolid. Spain.*
[b]*Health time Corporation. Córdoba.Spain.*

Abstract

Background and Objective: This paper proposes a new highly efficient implementation of a 3D+t groupwise registration problem based on the free-form deformation paradigm. *Methods*: The deformation operation is posed as a cascade of 1D convolutions, achieving a great reduction in the execution time for the evaluation of transformations and gradients. *Results*: The proposed registration method has been applied to 4D cardiac MRI and 4D thoracic CT dataset. Results show an average runtime reduction above 90%, both in CPU and GPU executions, compared with classical tensor product formulation. *Conclusions*: Our implementation is independent of the registration metric used and its adaptation to multiresolution strategies is straightforward. Therefore, it can be extremely useful for speeding up image registration procedures in different applications in which high dimensional data are involved.

Keywords: free-form deformation, b-splines, convolution, non-rigid registration, groupwise registration, efficient implementation

---

[1]*Corresponding Author*: Rosa M. Menchón-Lara, PhD. Laboratorio de Procesado de Imagen, Universidad de Valladolid, ETSI de Telecomunicación, Departamento de Teoría de la Señal y Comunicaciones e Ingeniería Telemática. Campus Miguel Delibes sn., 47011 Valladolid, Spain. e-mail: rmenchon@lpi.tel.uva.es. Phone: (+34) 983 423660.

1. Introduction

Image registration is the procedure that pursues to spatially align a set of images for subsequent processing. Many applications rely on an accurate registration procedure (see, for instance, [1, 2] and references therein). Image registration methods may also be applied for motion estimation in dynamic images, where the goal is to quantify the function of moving organs or the elasticity of vessels [3], or as a means to give rise to efficient image acquisition procedures [4, 5].

When a set of images are to be registered, pairwirse or groupwise approaches may be adopted; the former may be carried out sequentially, i.e., ordering the images and registering pairs of consecutive images to each other, or may be based on selecting one of the images as a common reference and registering the rest of the set to that reference. Pairwise approaches may not be the best choice since the procedure is not executed globally but as a partition of isolated problems. On the other hand, Groupwise (GW) approaches consist in a single joint procedure [6] and have shown advantages over the pairwise approaches (as this is the case in [4] and [5] in a dynamic MR reconstruction procedure).

In general terms, image registration involves three main steps: 1) selection of a deformation model; 2) definition of a cost function, and 3) adoption of an optimization strategy [3]. In this paper we focus on the deformation model and, specifically, on the free-form deformations (FFD) for non-rigid registration [7, 8]. An FFD model inherently gives rise to smooth deformation fields with the appropriate selection of its basis functions. However, these models also have limitations [9]. In particular, it is important to highlight limitations on memory space and execution time with large-scale 3D+t images. This issue has been analyzed in previous works, in which different solutions have been proposed to improve their efficiency.

A multi-level approach was proposed by Schnabel *et al.* in [10], which generalized Rueckert's method [7] by simulating a non-uniform control point distribution. The multi-resolution registration is the sum of a hierarchy of deformations at different mesh resolutions. Successive deformation refinements are evaluated only in a subset of active control points at each level, which involves a lower run-time in the high resolution levels. In [11], Sun *et al.* propose the use of lower-order basis functions combined with stochastic perturbation and smoothing techniques. Other works suggest parallel implementations of the FFD based registration. Rohlfing and Maurer [12] propose an efficient parallel implementation using 64 CPUs of a supercomputer. The proposed approach by Ino *et al.* in [13] incorporates data distribution, data-parallel processing, and load balancing techniques into the aforementioned Schnabel's registration algorithm. In [14], Rohrer *et al.* propose a multicore implementation of the original Rueckert's method for a Cell Broadband Engine platform. In addition, several works propose efficient implementations using graphics processing units (GPU), reporting significant speedups over CPU implementations [15, 16, 17, 18, 19, 20].

Recently, some Deep Learning approaches have been applied to the registration problem. These solutions promise fast registrations in operation mode, once the networks have been trained. For pairwise image registration, the VoxelMorph learning framework [21, 22] parameterizes the deformations via a convolutional neural network (CNN). In [23], the authors use a very involved architecture with a large number of parameters as well as several skip connections for GW registration of multimodal static images. In these approaches, only 2D solutions are reported.

Interpolation by means of convolutions is well-known [24, 25] for over four decades, although examples of recent contributions can be found [26]; however, a lukewarm opinion about their efficiency has been recently reported [27]. As for FFDs, their implementation based on convolutions has not gained popularity since its onset [7, 28] and, to the best of our knowledge, reported implementations are not based on convolutions either. Our contribution consists in a new and highly efficient implementation of a 4D (3D+t) GW registration approach with FFDs based on simple convolutional operations, which leads to a great reduction in the execution time. Results will be shown with monomodal images of both MR and CT images, although our approach is independent of the metric used since the core of the proposal relies on how the transformation is tackled. The proposed methodology has been compared with the classical FFD implementation based on tensor products, both in CPU and GPU. Results show a mean runtime reduction of 91.5% for CPU and of 93.2% for GPU executions in Matlab. It is worth mentioning that the proposed approach could be adapted to multi-resolution scenarios and may further benefit from parallelization strategies and sparse convolution optimizations.

The remainder of this paper is structured as follows. Section 2 describes the methods; we first revise the FFD concepts and then we analytically show its 3D convolutional implementation. We then calculate the best-case complexity of the method and compare it with the implementation of the spatial transformation using tensor products. This section ends with a description of the 4D GW registration approach. Results are shown in Section 3 and the discussion is carried out in Section 4. Section 5 gathers the main conclusions of the paper. Finally, we also include an appendix to give insight into the gradient calculations and its implementation with convolutions.

2. Methods

2.1. Background

Free-form deformations (FFDs) are a powerful geometric modeling technique which can be used to represent arbitrary deformations applied to objects [29]; hence, they have become a popular approach for non-rigid registration algorithms [8], specially for medical image applications. The basic idea consists in locally deforming a given image by manipulating a grid of control points distributed across the image at an arbitrary mesh resolution [7]. Image registration based on FFD commonly uses B-spline functions to define the transformations; specifically, third-order B-spline basis functions are selected because of their good balance between function smoothness and support region [11].

The goal of any image registration method is to find the spatial transformation $\mathbf{T}$ that maps each point in the source image into the corresponding point in the target image. Therefore, for the particular 3D case, $\mathbf{T} : \mathbf{x} \mapsto \mathbf{x}'$, such that $\mathbf{x}' = \mathbf{T}(\mathbf{x})$, where $\mathbf{x} = (x_1, x_2, x_3) \in \mathcal{X} \subset \mathbb{R}^3$ represents the *fixed* image domain and $\mathbf{x}' = (x'_1, x'_2, x'_3) \in \mathcal{X}' \subset \mathbb{R}^3$ stands for the moving image domain. According to a FFD-based registration [7, 28], $\mathbf{T}$ is defined through a cubic B-spline interpolation from a lattice of control points $\mathbf{u} = (u_1, u_2, u_3)$, taking integer values $-\lfloor K_l/2 \rfloor \leq u_l \leq \lfloor (K_l - 1)/2 \rfloor$ (where K_l is the number of control points along each dimension l, $l \in \{1, 2, 3\}$). The resolution of the control point grid, i.e., the spacing in pixels between control points along each dimension, is given by $\boldsymbol{\Delta} = (\Delta_1, \Delta_2, \Delta_3)$. Then, if we denote the center of the control point mesh in $\mathcal{X}$ as $\mathbf{c} = (c_1, c_2, c_3) = (\lceil N_1/2 \rceil, \lceil N_2/2 \rceil, \lceil N_3/2 \rceil)$ —with N_l the volume size along dimension l—, the location of each control point $\mathbf{u}$ in $\mathcal{X}$ can be expressed as $\mathbf{p_u} = (p_{u_1}, p_{u_2}, p_{u_3}) = \mathbf{c} + \boldsymbol{\Delta} \circ \mathbf{u}$, where '$\circ$' symbolizes the Hadamard product. Note that $\boldsymbol{\Delta}$ is defined such that $\Delta_l \cdot K_l \leq N_l$, with $\Delta_l > 0$. With these previous considerations, the B-spline based 3D transformation is defined as

$$\mathbf{T}(\mathbf{x}) = \mathbf{x} + \sum_{\mathbf{u} \in \mathcal{N}(\mathbf{x})} \left(\prod_{l=1}^{3} B_3 \left(\frac{x_l - p_{u_l}}{\Delta_l} \right) \right) \cdot \boldsymbol{\theta}_{\mathbf{u}} \quad (1)$$

where $\mathcal{N}(\mathbf{x})$ is a certain vicinity of the voxel, $\boldsymbol{\theta}_{\mathbf{u}} = (\theta_{u_1}, \theta_{u_2}, \theta_{u_3})$ represents the control point displacements, and B_3 stands for the third order B-spline function obtained through the Cox-DeBoor recursion formula as defined in [30].

B-spline functions have a compact support. Therefore, the displacement of a control point affects the transformation only in a local neighborhood of that control point. In other words, a given control point has an influence radius R_l. Thus, Eq. (1) can be expressed as

$$\mathbf{T}(\mathbf{x}) = \mathbf{x} + \sum_{u_1=C_1^1}^{C_1^2} \sum_{u_2=C_2^1}^{C_2^2} \sum_{u_3=C_3^1}^{C_3^2} \left(\prod_{l=1}^{3} B_3 \left(\frac{x_l - p_{u_l}}{\Delta_l} \right) \right) \cdot \boldsymbol{\theta}_{\mathbf{u}} \tag{2}$$

where $C_l^1 = -\lfloor (c_l - x_l + R_l) \rfloor / \Delta_l$, and $C_l^2 = \lfloor (x_l - c_l + R_l)/\Delta_l \rfloor$.

Therefore, the deformation at each point is given by the 3D tensor product of 1D functions [7, 28]. Displacements of the control points ($\boldsymbol{\theta}_{\mathbf{u}}$) act as parameters of the transformation and the resolution of the control point mesh defines the number of degrees of freedom, and consequently the computational complexity [8]. Moreover, the spacing between control points restricts the performance of the registration [10]; a coarse resolution of the grid of control points gives rise to more global and smoother deformations, whereas a finer resolution allows bring forth local and less smooth deformations.

In order to determine the optimal transformation, a registration cost function must be defined and minimized. Typically, the cost function consists of two terms [7]:

$$\mathcal{C}(\Theta) = \int_{\mathcal{X}} \left[-\mathcal{C}_{similarity}(I_F(\mathbf{x}), I_M(\mathbf{T}(\mathbf{x}))) + \lambda \mathcal{C}_{smooth}(\mathbf{T}(\mathbf{x})) \right] d\mathbf{x} \tag{3}$$

where I_F stands for the *fixed* image and I_M for the *moving* image. The first term in Eq. (3) represents the cost associated with the image similarity, which measures the accuracy of the registration, i.e. the degree of alignment between the two images. In addition, to constrain the deformation to be smooth, a penalty regularization term weighted by a factor λ is included in the registration cost function ($\mathcal{C}_{smooth}(\mathbf{T})$).

2.2. Convolutional Implementation

In this work, the well-known idea of convolution-based interpolation is revisited [24, 25]. The goal of this Section is to establish the link between the B-spline FFD formulation based on tensor products [7, 28] and the convolutional implementation.

Due to the compact support property of B-splines referred to above, the summations in Eq. (2) can be extended to the whole control point mesh. With this idea in mind, we now assume that the FFD is defined on a Cartesian coordinate system. We first consider, for simplicity, a 1D scenario with point positions taking on integer values $1 \leq i \leq N_1$ and a set of K_1 control points with Δ_1 spacing, located on a subset of the N_1 points. We use the running index $1 \leq u_1 \leq K_1$ to refer to each control point and assume p_{u_1} denotes the location of the control point with index u_1 in the N_1-point grid, i.e., $p_{u_1} \in \{1, \ldots, N_1\}$. Then, we can write the 1D transformation as:

$$T(i) = i + \sum_{u_1=1}^{K_1} B_3\left(\frac{i - p_{u_1}}{\Delta_1}\right) \cdot \theta_{u_1} = i + \sum_{u_1=1}^{K_1} B^{\Delta_1}(i - p_{u_1}) \cdot \theta_{u_1} \tag{4}$$

with $B^{\Delta_l}(m) := B_3(m/\Delta_l)$. Now, for convenience we rewrite this expression above using a new index $1 \leq q_1 \leq N_1$, such that

$$T(i) = i + \sum_{q_1=1}^{N_1} B^{\Delta_1}(i - i_1(q_1)) \cdot \underbrace{[\delta(q_1 - i_1(q_1)) \cdot \theta_{i_2(q_1)}]}_{\Pi(q_1)} \tag{5}$$

where δ is the Kronecker delta, i.e. $\delta(t) = 1$ if $t = 0$ and $\delta(t) = 0$ if $t \neq 0$, and $i_1(q_1)$ is a function defined to cancel the contribution of any point q_1 that it is not a control point,

$$i_1(q_1) = \begin{cases} q_1 & \text{if } \exists u_1 : q_1 = p_{u_1} \\ 1/2 & \text{otherwise} \end{cases} \tag{6}$$

and $i_2(q_1)$ is a function defined to select the appropriate displacement:

$$i_2(q_1) = \begin{cases} u_1 & \text{if } q_1 = p_{u_1} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

and we set $\theta_0 = 0$ arbitrarily. Therefore, function $\Pi(q_1)$ in Eq. (5) is null on those points q_1 on which a control point is not located (which, in turn, makes the value of θ_0 irrelevant). Then, Eq. (5) can be extended and reformulated as a convolution operation[2]:

$$T(i) = i + \sum_{q_1=1}^{N_1} B^{\Delta_1}(i - q_1) \cdot \Pi(q_1) = i + B^{\Delta_1}(i) * \Pi(i) \tag{8}$$

[2] Following [31], $y[n] = x[n] * h[n]$ denotes the convolution of the signals $x[n]$ and $h[n]$ evaluated at point n.

The extension to a 3D scenario is straightforward; assume a Cartesian grid where voxel positions take on integer values $\mathbf{x} = (i, j, k)$ (with $1 \le i \le N_1$; $1 \le j \le N_2$; $1 \le k \le N_3$) and a control point mesh with cardinality $K = K_1 \times K_2 \times K_3$, located on a subset of the grid points. Then,

$$\mathbf{T}(\mathbf{x}) = \mathbf{x} + \sum_{q_1=1}^{N_1} B^{\Delta_1}(i - i_1(q_1)) \sum_{q_2=1}^{N_2} B^{\Delta_2}(j - j_1(q_2)) \sum_{q_3=1}^{N_3} B^{\Delta_3}(k - k_1(q_3)) \cdot \mathbf{\Pi}(q_1, q_2, q_3) \tag{9}$$

where

$$\mathbf{\Pi}(q_1, q_2, q_3) = \delta(q_1 - i_1(q_1))\delta(q_2 - j_1(q_2))\delta(q_3 - k_1(q_3))\boldsymbol{\theta}_{i_2(q_1), j_2(q_2), k_2(q_3)} \tag{10}$$

with the functions j_1 and k_1 defined according to Eq. (6) and functions j_2 and k_2 defined as Eq. (7), respectively, to address the two other spatial dimensions (and, accordingly using u_2 and u_3 instead of u_1). Similarly, $\mathbf{T}(\mathbf{x})$ can be reformulated as:

$$\mathbf{T}(\mathbf{x}) = \mathbf{x} + \underbrace{\sum_{q_1=1}^{N_1} B^{\Delta_1}(i - q_1) \underbrace{\sum_{q_2=1}^{N_2} B^{\Delta_2}(j - q_2) \underbrace{\sum_{q_3=1}^{N_3} B^{\Delta_3}(k - q_3) \cdot \mathbf{\Pi}(q_1, q_2, q_3)}_{\mathbf{\Psi}(q_1, q_2, k)}}_{\mathbf{\Phi}(q_1, j, k)}}_{\mathbf{\Gamma}(i, j, k)} \tag{11}$$

Therefore, the 3D tensor product in the original formulation for the deformations (in Eq. (2)) is reduced to the evaluation of simple 1D discrete convolutions along each coordinate axis, according to the following steps:

1. Evaluation of $\mathbf{\Psi}(i, j, k)$: 1D convolution along k-axis evaluated only in the subset of (p_{u_1}, p_{u_2}) corresponding to control point positions (see Figs. 1a and 1b) as follows:

$$\mathbf{\Psi}(p_{u_1}, p_{u_2}, k) = B^{\Delta_3}(k) * \mathbf{\Pi}(p_{u_1}, p_{u_2}, k) \tag{12}$$

2. Evaluation of $\mathbf{\Phi}(i, j, k)$: 1D convolution along j-axis evaluated in p_{u_1} points corresponding to control point locations (see Fig. 1c) as:

$$\mathbf{\Phi}(p_{u_1}, j, k) = B^{\Delta_2}(j) * \mathbf{\Psi}(p_{u_1}, j, k) \tag{13}$$

3. Evaluation of $\mathbf{\Gamma}(i, j, k)$: 1D convolution along i-axis evaluated in the whole Cartesian grid (see Fig. 1d):

$$\mathbf{\Gamma}(i, j, k) = B^{\Delta_1}(i) * \mathbf{\Phi}(i, j, k) \tag{14}$$

4. Finally, the transformation for each voxel is

$$\mathbf{T}(\mathbf{x}) = \mathbf{x} + \mathbf{\Gamma}(i, j, k) \tag{15}$$

The convolutional implementation of the B-spline based FFD model entails a substantial improvement in the computational efficiency of the registration process, since the operators are limited to a number of points and, in addition, the convolution operator has been highly optimized in different development environments, e.g. Matlab, since it is commonly used in signal processing. Note that this new interpretation in terms of 1D convolutions also affects the gradient evaluation needed in the optimization process of the 3D transformation, which contributes to a greater reduction of global execution time. This is now explored.

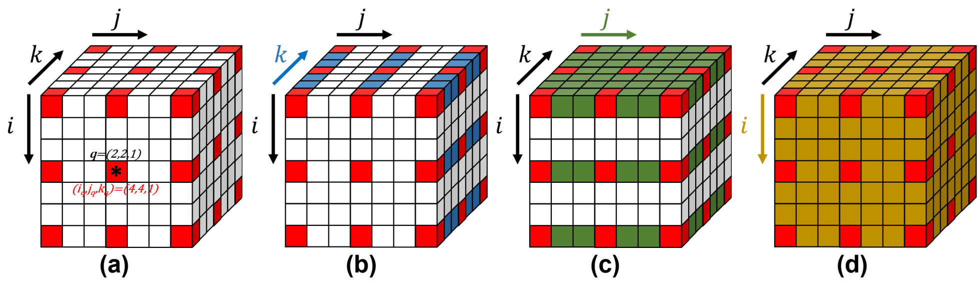


Figure 1: Representation of the efficient 3D FFD model evaluation: (a) regular set of control points (red boxes) distributed across the Cartesian grid; (b) 1D convolution along k-axis evaluated in those rows and columns containing control points; (c) 1D convolution along j-axis evaluated in all slices, but only in those rows containing control points; (d) 1D convolution along i-axis evaluated in the whole mesh.

2.3. Computational Complexity

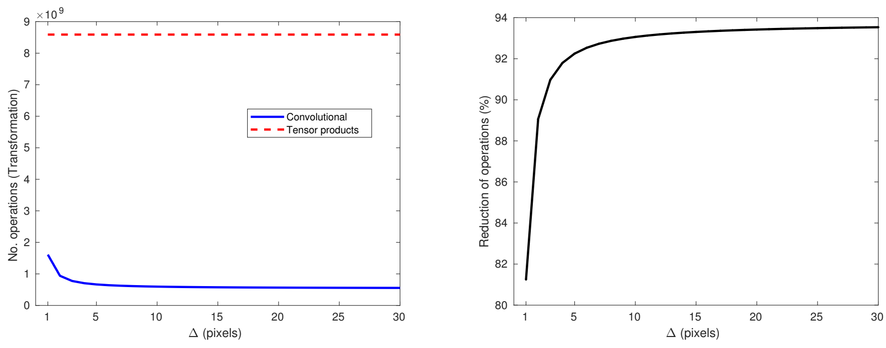
In order to quantify the computational benefits of the convolutional approach we evaluate the computational cost of both approaches in terms of number of operations. For simplicity, we consider M as a shortcut for any of the volume spatial dimensions, since we assume the three dimensions will be comparable.

Similarly, the parameter Δ will be used as the distance (in pixels) between control points in any of the dimensions and the parameter K will be the number of control points along any dimension. Hence, $K \approx M/\Delta$. In addition, the compact support of the B-spline function (B^{Δ}) in each dimension is $S = 2R_l - 1 = (E+1)\Delta - 1$.

The computational cost for a dense 1D discrete convolution along one line of a given dimension, e.g. $B^{\Delta}(k) * \mathbf{\Pi}(1, 1, k)$, is $[M \times S]$. However, our approach intrinsically involves the evaluation of highly sparse convolutions; specifically the convolution computation at each point only consists in $(E+1)$ effective products (i.e., those not known beforehand to be null). Therefore, if we consider an optimized operator for sparse convolutions, it is possible to conclude that: (i) Eq. (12) takes $[M \times (E+1) \times K \times K]$ operations; (ii) Eq. (13) takes $[M \times (E+1) \times K \times M]$ operations; (iii) Eq. (14) takes $[M \times (E+1) \times M \times M]$ operations. Thus, the total number of operations to compute the transformation for the whole image volume following the convolutional formulation can be expressed as: $[M^3 \times (E+1) \times (1/\Delta^2 + 1/\Delta + 1)]$.

As for the evaluation of the transformation using the classical tensor product approach, it is possible to precompute the 3D B-spline product matrix, i.e. $\prod_{l=1}^{3} B^{\Delta}(x_l - p_{u_l})$, during the algorithm initialization. Therefore, the computational cost of Eq. (2) is reduced to $[M^3 \times (E+1)^3]$.

As can be seen, the computational complexity of the convolutional formulation depends on the resolution of the control point grid and, consequently, on the number of control points along each dimension. Figure 2 shows the theoretical



(a) (b)

Figure 2: Computational complexity for transformation evaluation: (a) Number of operations needed for volume size $M = 512$; (b) Reduction of operations (%) with the proposed convolutional formulation.

number of operations for a range of Δ values and the reduction percentage for the proposed efficient implementation. Note that for $\Delta > 2$ the number of operations is reduced above 90% with the convolutional approach.

2.4. Group-Wise Registration

The convolutional approach for B-spline FFD implementation presented here can be applied to pairwise image registration procedures, both in 3D and 2D domains, as well as to group-wise (GW) registration. This work focuses on the latter. Previous works have successfully applied elastic GW registration to the alignment of cardiac MR perfusion images [32], to motion estimation in cardiac cine MRI [33, 34, 35, 36], and to abdominal diffusion-weighted MRI [37, 38]. Many other uses of GW registration have been described elsewhere [23, 39].

Now, as an application example, we focus on the non-rigid GW registration of 4D (3D+t) cardiac MR images, with the aim of providing a robust estimation of the cardiac motion during cardiac cycle. Consider a dynamic MRI sequence $\mathbf{m} = \{\mathbf{m}_1(\mathbf{x}_1), \mathbf{m}_2(\mathbf{x}_2), ..., \mathbf{m}_N(\mathbf{x}_N), \}$, with temporal index $1 \leq n \leq N$ and $\mathbf{m}_n(\mathbf{x}_n)$ defined over the 3D image domain $\mathbf{x}_n = (i_n, j_n, k_n) \in \mathcal{X}_n \subset \mathcal{R}^3$. The images are originally defined at grid positions (integer coordinates) albeit during the registration process the images may be resampled at non-grid coordinates. See Section 3 for further details on interpolation.

Now, the goal is to find the optimal set of spatial transformations $\mathcal{T}_{\mathbf{\Theta}} = \{\mathbf{T}_{n,\mathbf{\Theta}} : \mathbf{x}'_n = \mathbf{T}_{n,\mathbf{\Theta}}(\mathbf{x}) \in \mathcal{X}_n\}$ which maps the coordinates of each material point in a common reference image (say, $\mathbf{x} \in \mathcal{X}_{ref}$) into its corresponding coordinates in $\mathcal{X}_n$, $1 \leq n \leq N$ (see Fig. 3).

As previously stated, optimal parameters of the deformations ($\mathbf{\Theta} = \{\boldsymbol{\theta}_{n\mathbf{u}}\}$) are found by minimizing a cost function

$$\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{\operatorname{argmin}}\, \mathcal{C}(\mathbf{\Theta}) = \underset{\mathbf{\Theta}}{\operatorname{argmin}} \left(\int_{\mathcal{X}} (\mathcal{V}_{\mathbf{\Theta}}(\mathbf{x}) + \mathcal{R}_{\mathbf{\Theta}}(\mathbf{x}) d\mathbf{x} \right) \tag{16}$$

As for the second term $\mathcal{R}_{\mathbf{\Theta}}$, we have used the the simple regularizer proposed in [40]. This is an alternative to traditional Jacobian penalty methods that relaxes the invertibility condition by using a piecewise quadratic penalty function directly on the deformation coefficients that encourages diffeomorphic transformations and requires less computation time. As for the first term, the GW registration metric is based on the variance of the intensity along time. Specifically, the function $\mathcal{V}_{\mathbf{\Theta}}$ is defined as

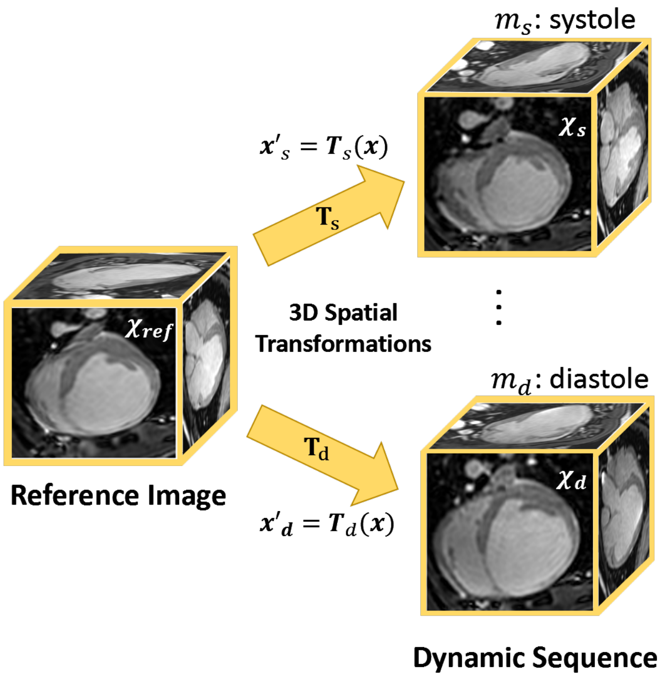


Figure 3: Scheme of spatial transformations in GW registration for 3D cine cardiac MRI.

$$\begin{aligned}\mathcal{V}_{\boldsymbol{\Theta}}(\mathbf{x}) &= \frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{m}_n(\mathbf{T}_{n,\boldsymbol{\Theta}}(\mathbf{x})) - \frac{1}{N}\sum_{k=1}^{N}\mathbf{m}_k(\mathbf{T}_{k,\boldsymbol{\Theta}}(\mathbf{x}))\right)^2 \\ &= \frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{m}_n(\mathbf{x}'_n) - \overline{\mathbf{m}_{\boldsymbol{\Theta}}(\mathbf{x})}\right)^2 \end{aligned} \quad (17)$$

where $\overline{\mathbf{m}_{\boldsymbol{\Theta}}(\mathbf{x})}$ is the image average over time after applying $\mathcal{T}_{\boldsymbol{\Theta}}$.

In the optimization process, the gradient of the function $\mathcal{C}(\boldsymbol{\Theta})$ must be evaluated at each iteration. This operation represents the bottleneck of the FFD based registration algorithms. In our case, the gradient evaluation is also performed by simple convolution operations (see Appendix A), which leads to a greater reduction in the execution time of the proposed registration method.

3. Results

3.1. 3D Cine cardiac MRI

The GW registration algorithm has been applied to eight isotropic 3D+t cardiac MR scans of different swine in order not only to test the ability of this method to capture the non-rigid motion of the heart, but also to analyze the computational benefits and efficiency of the convolutional implementation proposed in this work. All images have a high spatial resolution (voxel size = 1 mm^3) with field of view equal to $183\times183\times183$ mm^3 and show a short-axis (SA) view of the heart. Each MRI sequence consists of $N = 20$ temporal frames, where each of them represents a different phase of the cardiac cycle.

The 4D elastic GW registration has been carried out four times for each cardiac cine MRI, following two different approaches, both on CPU and GPU, to find the optimal set of transformations: (a) the classical implementation based on 3D tensor products, and (b) the proposed convolutional implementation. As images have similar geometry, some parameters of the FFD model were fixed for all cases. The spacing of control points is $8\times8\times8$ mm^3, which led to a $23\times23\times23$ control point mesh for each image frame. Therefore, this means that a total of 730,020 $(= 23^3 \times 3 \times 20)$ parameters —components of $\mathbf{\Theta}$— must be optimized in the GW registration process. As discussed in Sect. 2, cubic B-spline functions are used to model the deformations. Therefore, each control point affects the transformation of voxels within a neighborhood with 15 mm of radius, a total of $31\times31\times31$ voxels in this case.

The registration process is accompanied by a linear interpolation that allows us to apply the transformation described in Section 2.2. Specifically, this process consists in looping over all voxels in the common reference image $\mathbf{x} \in \mathcal{X}_{ref}$, and interpolating the moving image at the transformed coordinates ($\mathbf{x}'_n$, mapped position), to fill in this value at position $\mathbf{x}$ in the registered image.

The optimization problem in Eq. (16) is solved by means of an iterative nonlinear conjugate gradient algorithm. In particular, we use Polak-Ribière [41] constrained by Fletcher-Reeves [42], based on strong Wolfe line search. A maximum number of 100 iterations is set for the optimizer.

Figure 4 illustrates the GW registration results. The optimization processes for the two implementations are virtually identical but for irrelevant numerical precision; specifically, the difference between the accumulated squared moduli of displacements with both methods is on the order of 10^{-6} pixels in both CPU and GPU.

Binary masks of the myocardium in systole and diastole from an expert manual segmentation are available for each cardiac MRI sequence. In order to analyze

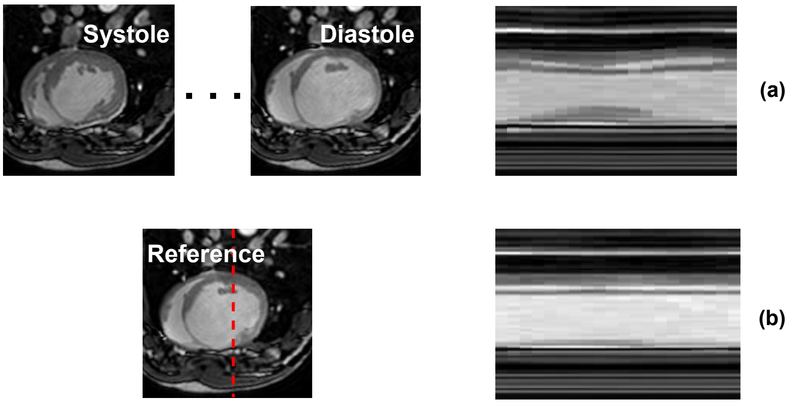


Figure 4: GW registration results. (a) (left) Dynamic MRI sequence before registration (systolic and diastolic cardiac phases); (b) (left) Reference image after registration. Images in right column of the figure show the temporal evolution of the intensity profile indicated in the reference image by the red dashed line.

the registration accuracy, the mask at systole is transformed to the diastolic phase and compared to the expert mask at diastole. The transformation that maps coordinates from diastole to systole is defined as the composition $\mathbf{T}_{dia,sys} = \mathbf{T}_{sys} \circ \mathbf{T}_{dia}^{-1}$, where the inverse transformation ($\mathbf{T}_{dia}^{-1}$) maps the coordinates from the diastolic phase to the common reference space; this second transformation is approximated iteratively by an additional optimization procedure, as described in [3]. The diastolic manual mask is transformed similarly to systolic phase. The Dice coefficient was evaluated in each case; we also evaluated the end-diastolic and end-systolic volumes (see Table 1). In addition, the dynamic cardiac MRI sequences were also registered using the *Elastix*[3] software, with the parameter files used in [3] for GW registration.

The overall execution times in each case for our Matlab implementations are included in Table 2. In addition, specific runtimes per iteration for the evaluation of transformations and gradients are shown in Table 3. As stated in Sect. 2.3, the computational complexity depends on $\mathbf{\Delta}$. For this reason, CPU runtimes have been also analyzed experimentally for different resolutions of the control point grid (see Fig. 5).

[3] http://elastix.isi.uu.nl/

ID	DICE COEFFICIENT (%)				EDV (mL)			ESV (mL)		
	Diastole		Systole							
	Conv.	Elastix	**Conv.**	Elastix	**Conv.**	Elastix	GT	**Conv.**	Elastix	GT
1	**83.2**	81.9	**81.2**	80.8	**119**	121	128	**94**	93	89
2	**80.8**	78.3	**78.0**	78.3	**86**	90	99	**40**	37	37
3	**81.8**	77.7	**81.6**	79.2	**145**	151	170	**103**	97	99
4	**87.9**	82.4	**87.5**	82.7	**68**	66	70	**31**	32	31
5	**85.7**	83.0	**85.1**	82.3	**71**	66	69	**24**	27	25
6	**85.7**	81.0	**82.9**	78.2	**133**	130	154	**91**	89	81
7	**85.7**	72.6	**84.9**	72.1	**142**	139	146	**83**	88	89
8	**85.7**	79.0	**84.6**	78.7	**208**	191	212	**151**	161	156

Table 1: Registration results for 4D cardiac MRI dataset. Several metrics for both the proposed GW convolutional approach and the Elastix registration are shown for comparison, namely, Dice coefficient between manually segmented myocardial mask for diastolic phase and the corresponding transformed mask from the systolic phase, and vice versa (left), End-diastolic volume (EDV, three columns in the middle of the table) and end-systolic volume (ESV, three left-most columns). Volumes are measured in mL on both the transformed masks and on the manually segmented masks (ground-truth, GT).

GW registration approach	CPU Time (min.)	GPU Time (min.)
3D tensor products	674.16 ± 56.98	33.15 ± 2.99
1D convolutions	57.15 ± 3.11	2.24 ± 0.13
Convolutional improvement	91.5 %	93.2 %

Table 2: Computational times (minutes) of the 4D elastic GW registration method in CPU and GPU executions for 3D tensor product and convolutional implementation of the FFD (mean $\pm$ standard deviation from eight different dynamic cardiac MRI sequences) and improvement (time reduction) of the convolutional approach.

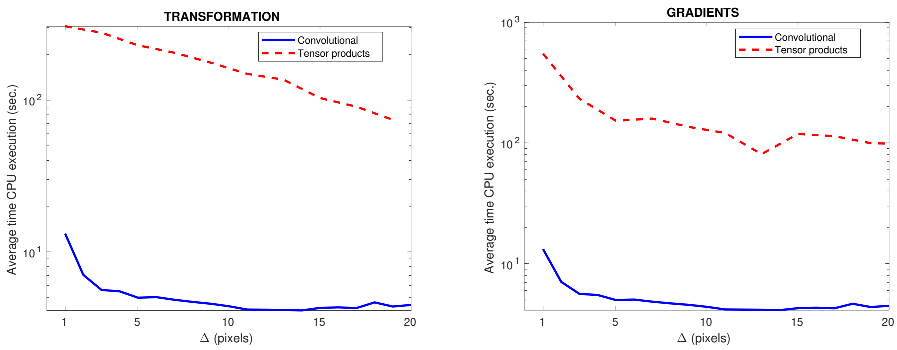
3.2. 3D+t Thoracic CT

4D CT data of the lungs was taken from the publicly available POPI-model [43]. This dataset includes a total of six sequences with 10 respiratory phases each. Moreover, the POPI dataset provides 100 manually annotated landmarks in the end-of-inspiration and end-of-expiration phases. These 100 anatomical landmarks are also available in all frames for 3 CT sequences.

In this case, each CT sequence was registered using the proposed convolutional approach with a spacing between control points of $\boldsymbol{\Delta} = (12, 12, 12)$ mm.

Operation	**CPU execution (sec.)**			**GPU execution (sec.)**		
	T. Prod.	Conv.	Reduction	T. Prod.	Conv.	Reduction
Transformations	117.51	4.57	96.1 %	2.15	0.33	84.7 %
Gradients	172.67	11.45	93.4 %	11.99	0.44	96.3 %

Table 3: Mean execution times (seconds) for the critical operations in the GW registration procedure: evaluation of the set of transformations and gradient calculation during the optimization process.



(a) (b)

Figure 5: Empirical Matlab runtimes in CPU (seconds): (a) Transformation evaluation; (b) Gradient computation.

Images were decimated in the two first dimensions by a factor of 2. After registration, the obtained transformations were scaled up to the original spatial resolution. More details about the CT sequences, control point grid and runtimes in CPU and GPU are included in Table 4.

In order to validate the accuracy of the proposed registration method, the group-wise target registration error (gwTRE) [1, 3] was evaluated using the aforementioned collection of landmarks $\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_{N_t}\}$, with N_t frames for which a N_p number of landmarks are available, $\mathbf{P}_i = \{\mathbf{p}_{i,1}, \mathbf{p}_{i,2}, \cdots, \mathbf{p}_{i,N_p}\}$:

$$\text{gwTRE} = \frac{1}{N_t}\frac{1}{N_p}\sum_{i\neq r}^{N_t}\sum_{j=1}^{N_p} \parallel \mathbf{T}_{i,r}(\mathbf{p}_{i,j}) - \mathbf{p}_{r,j} \parallel \quad (18)$$

where $\mathbf{T}_{i,r} = \mathbf{T}_r \circ \mathbf{T}_i^{-1}$ stands for the transformation from the ith frame to a reference frame (end-of-inspiration, in this case). As in the case of cardiac MRI dataset, the CT sequences were also registered using the *Elastix* software to validate our registration. Results are shown in Table 5.

ID	Volume Size	Spatial Res. (mm)	Grid	Iter.	Time (min.)	
					CPU	GPU
1	256×256×141×10	1.95×1.95×2	42×42×27	34	17.1	0.7
2	256×256×169×10	1.95×1.95×2	42×42×33	26	17.1	0.7
3	256×256×170×10	1.76×1.76×2	42×42×33	50	31.5	1.1
4	256×256×187×10	1.56×1.56×2	42×42×35	22	18.4	0.6
5	256×256×139×10	2.34×2.34×2	42×42×27	37	21.1	0.9
6	256×256×161×10	2.34×2.34×2	42×42×31	16	15.0	0.6

Table 4: Registration of the POPI dataset. Volume size, spatial resolution and control point grid are specified for each CT sequence. CPU and GPU runtimes are expressed in minutes; the number of iterations of the optimizer in each case for GW registration with the proposed convolutional formulation is also included.

ID	N_t	gwTRE$_{\text{conv}}$		gwTRE$_{\text{Elastix}}$	
		mm.	pixels	mm.	pixels
1	10	**0.918**	**0.70**	1.740	1.28
2	10	**1.810**	**1.30**	3.480	2.46
3	10	**1.090**	**0.88**	1.870	1.46
4	2	**1.312**	**0.93**	2.720	1.67
5	2	**0.806**	**0.53**	2.483	1.53
6	2	**0.982**	**0.64**	2.331	1.30
Mean ± Std. dev.		**1.15±0.33**	**0.83±0.25**	2.44±0.58	1.62±0.40

Table 5: Groupwise target registration error for POPI dataset in millimeters and pixels. We show the results for both the proposed method and for Elastix registration.

4. Discussion

All experiments non based on Elastix were run using MATLAB R2019a on a VM with two processors (Intel Xeon E5-2697 v4 @ 2.30 GHz), with a total of 35 cores (2 threads per core) and 500 GB of RAM. The GPU executions were carried out in a nVIDIA Quadro RTX5000 device by using the CUDA capabilities of MATLAB.

The GW registration approach proposed in this work has been tested on two different datasets: 4D cardiac MRI sequences (Sect. 3.1) and 4D thoracic CT sequences (Sect. 3.2). Although we focus on monomodal image registration, its adaptation to multimodal registration is straightforward and only requires the appropriate change of the registration metric. As previously stated, the method relies on how that transformation is tackled.

As for the registration accuracy, manual and transformed segmentation masks of the myocardium have been compared for the MRI dataset. Table 1 shows the corresponding Dice coefficient in each case, and it also includes the results obtained from *Elastix*. The average values are 84.6% and 83.2% for the convolutional GW registration in diastole and systole, respectively. As can be seen, our registration slightly outperforms the Elastix results (Dice coeff. of 79.5% for diastole and 79% for systole). For the CT dataset, registration accuracy has been evaluated in terms of gwTRE on the available landmarks (Table 5). Results of our convolutional approach are also compared with the Elastix GW registration, where the latter shows less precision. We stress, however, that the main goal of this work is not to achieve a better registration accuracy but to propose an efficient alternative to computing the core of FFD transformations, as well as the gradients.

In terms of computational efficiency, the improvement of the convolution-based FFD has been quantified for the best-case implementation (Sect. 2.3) and experimental results have also been reported for ours (see Tables 2-3 and Fig. 5). Experiments carried out on cardiac MRI in Matlab with our convolutional proposal show a reduction in the GW registration time above 90% (91.5% and 93.2% on CPU and GPU implementation, respectively) in comparison with the classical tensor product formulation. In addition, execution time reduction in the estimation of both the transformations and the gradients has also been analyzed. Table 3 shows that gradients calculation represents the bottleneck of the non-rigid registration algorithm. Figure 5 reveals that regardless of the control point spacing, the convolutional formulation provides a reduction in CPU runtime above 93% for both the transformation and the gradient computation.

Previous works have already dealt with the high computational times of the FFD-based registration. However, the comparison of our proposal with these solutions is not straightforward. The referenced Elastix framework is executed from a compiled language while our routines are interpreted. Therefore, these comparisons have not been included. Other previously proposed solutions are based on sophisticated paralellization strategies both in multi-CPU [12, 13, 14] and GPU implementations [15, 17, 19]. Nevertheless, we have not explored any parallel processing or data distribution methodology. For our GPU executions, the default parallelization provided by the *Parallel Computing Toolbox* for GPU computing in Matlab has been directly used.

With our method, additional optimizations are possible; Elastix, for instance, makes use of undersampling strategies and adaptive stochastic gradient descent optimization to reduce the computational cost without losing accuracy. This can be easily accomplished with our method; the transformation in Eq (11) can be directly applied in any level of the resolution pyramid. In addition, if a stochastic gradient descent strategy is used, the gradient will be sparser than in the case that a batch gradient is used. This would give rise to additional savings by using sparse convolution algorithms, which is a current hot topic spurred by the deep learning paradigm shift [44, 45].

Our convolutional implementation allows for further optimization on GPU hardware that seems not so immediate with the classical formulation. Specifically, source data for Eq. (12) is sufficiently small so as to be held within GPU compute unit shared memories —aka as local memory—, if appropriately distributed among them, so that each compute unit uniquely sees its relevant neighborhood. This memory is typically one or two orders of magnitude faster than the GPU's main memory —aka as global memory—[46]. In contrast, Eq. (2) apparently needs access to the whole volume at once, which would not fit in the shared memory of any GPU we are aware of (a typical, current GPU has 5-70 compute units with 32-64 KB of shared memory each). Although source data for Eqs. (13)-(14) will likely not fit within current GPUs shared memories, it may be arranged so that accesses to main memory are contiguous for neighbor workers, which allows for GPU bus utilization to be maximized [46]. As evaluation of Eq. (2) needs to traverse source data along the three dimensions for any given voxel, combining memory accesses poses a more difficult problem at the least.

5. Conclusions

This paper proposes a highly efficient implementation for the FFD. B-spline based FFD models are reformulated by means of 1D convolutional operations. This simple modification allows us to substantially alleviate computational time in registration algorithms, since convolution operation has been extremely optimized on different development environments. Our proposal is especially useful to deal with high resolution 3D images, i.e. registration problems with large datasets. In this work, the new convolutional implementation of FFD has been tested in a 4D GW registration approach applied to cardiac cine MRI and thoracic CT data. The experiments show a reduction in the execution time above 90%, both in CPU and GPU executions. Note that the proposed implementation only affects the evaluation of deformations and gradients during the optimization process of image registration, therefore, it is independent of the registration metric used. In addition, it can also be adapted to multi-resolution registration strategies.

Acknowledgments

This work is partially supported by the Spanish '*Ministerio de Economía, Industria y Competitividad*' under grant TEC2017-82408-R. Images used for the experiments were provided by Dr. Filgueiras at the '*Centro Nacional de Investigaciones Cardiovasculares*' (CNIC), Madrid, Spain, to whom the authors are indebted.

Appendix A. Gradient Evaluation

The evaluation of the cost function gradient is detailed here. The parameters that define the set of transformations are $\mathbf{\Theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n, \ldots, \boldsymbol{\theta}_N\}$, with each $\boldsymbol{\theta}_n = \{\boldsymbol{\theta}_{n1}, \ldots, \boldsymbol{\theta}_{nu}, \ldots, \boldsymbol{\theta}_{nK}\}$, and $\boldsymbol{\theta}_{nu} = (\theta_{nu_1}, \theta_{nu_2}, \theta_{nu_3}) = \theta_{nul}$. Then, from expression in Eq. (16), the gradient of the cost function is defined as:

$$\frac{\partial \mathcal{C}(\mathbf{\Theta})}{\partial \theta_{nu_l}} = \int_{\mathcal{X}} \left(\frac{\partial \mathcal{V}_{\mathbf{\Theta}}(\mathbf{x})}{\partial \theta_{nu_l}} + \frac{\partial \mathcal{R}_{\mathbf{\Theta}}(\mathbf{x})}{\partial \theta_{nu_l}} \right) d\mathbf{x} \tag{A.1}$$

Here, we focus on the gradient of the function $\mathcal{V}_{\mathbf{\Theta}}$. For more details about gradient of the regularization term $\mathcal{R}_{\mathbf{\Theta}}$, refer to [40]. Thus, from the expression in Eq. (17) by applying the chain rule, we can write:

$$\begin{aligned}
\frac{\partial \mathcal{V}(\mathbf{x})_{\mathbf{\Theta}}}{\partial \theta_{nu_l}} &= \sum_{n'=1}^{N} \frac{\partial \mathcal{V}_{\mathbf{\Theta}}}{\partial \mathbf{m}_{n'}} \sum_{n''=1}^{N} \sum_{l'=1}^{3} \frac{\partial \mathbf{m}_{n'}}{\partial x'_{n''l'}} \frac{\partial T_{n''l'}(\mathbf{x})}{\partial \theta_{nu_l}} = \frac{\partial \mathcal{V}_{\mathbf{\Theta}}}{\partial \mathbf{m}_n} \cdot \frac{\partial \mathbf{m}_n}{\partial x'_{nl}} \frac{\partial T_{nl}(\mathbf{x})}{\partial \theta_{nu_l}} = \\
&= \underbrace{\frac{\partial \mathcal{V}_{\mathbf{\Theta}}}{\partial \mathbf{m}_n} \cdot \frac{\partial \mathbf{m}_n}{\partial x'_{nl}}}_{\mathbf{V}^{nl}(i,j,k)} \cdot \left[B_3\left(\frac{i - p_{u_1}}{\Delta_1}\right) B_3\left(\frac{j - p_{u_2}}{\Delta_2}\right) B_3\left(\frac{k - p_{u_3}}{\Delta_3}\right) \right]
\end{aligned} \tag{A.2}$$

where $x'_{nl} = T_{nl}(\mathbf{x})$, i.e., the l-th component of the transformation of point $\mathbf{x}$ in the common reference to the n-th image, and

$$\mathbf{V}^{nl}(\mathbf{x}) = \mathbf{V}^{nl}(i,j,k) = \frac{2}{N} \left(\mathbf{m}_n(\mathbf{x}'_n) - \overline{\mathbf{m}_{\mathbf{\Theta}}(\mathbf{x})} \right) \frac{\partial \mathbf{m}_n(\mathbf{x}'_n)}{\partial x'_{nl}} \tag{A.3}$$

Therefore,

$$\int_{\mathcal{X}} \frac{\partial \mathcal{V}_{\boldsymbol{\Theta}}(\mathbf{x})}{\partial \theta_{nu_l}} d\mathbf{x} \approx \sum_{i,j,k} \mathbf{V}^{nl}(\mathbf{x}) \left[B^{\Delta_1}(i - p_{u_1}) \cdot B^{\Delta_2}(j - p_{u_2}) \cdot B^{\Delta_3}(k - p_{u_3}) \right] \tag{A.4}$$

Moreover, due to the compact support of B-spline functions

$$\begin{aligned} \int_{\mathcal{X}} \frac{\partial \mathcal{V}_{\boldsymbol{\Theta}}(\mathbf{x})}{\partial \theta_{nu_l}} d\mathbf{x} &\approx \sum_{i=S_{11}}^{S_{12}} B^{\Delta_1}(i - p_{u_1}) \sum_{j=S_{21}}^{S_{22}} B^{\Delta_2}(j - p_{u_2}) \underbrace{\sum_{k=S_{31}}^{S_{32}} B^{\Delta_3}(k - p_{u_3}) \cdot \mathbf{V}^{nl}(i,j,k)}_{\boldsymbol{\Upsilon}} = \\ &= \sum_{i=S_{11}}^{S_{12}} B^{\Delta_1}(i - p_{u_1}) \underbrace{\sum_{j=S_{21}}^{S_{22}} B^{\Delta_2}(j - p_{u_2}) \cdot \boldsymbol{\Upsilon}(i,j,p_{u_3})}_{\boldsymbol{\zeta}} = \\ &= \underbrace{\sum_{i=S_{11}}^{S_{12}} B^{\Delta_1}(i - p_{u_1}) \cdot \boldsymbol{\zeta}(i, p_{u_2}, p_{u_3})}_{\boldsymbol{\Omega}} \qquad \text{(A.5)} \end{aligned}$$

(A.6)

where $S_{l1} = p_{u_l} - R_l$, and $S_{l2} = p_{u_l} + R_l$; with R_l the influence radius of the control points. Once again, functions $\boldsymbol{\Upsilon}$, $\boldsymbol{\zeta}$ and $\boldsymbol{\Omega}$ are the results of 1D convolutions at the control point locations, each of which along a different spatial dimension. This reformulation is of special interest because it allows us to evaluate the gradient with respect to the whole parameter set $\boldsymbol{\Theta}$ very easily. Specifically, at each iteration of the optimization process, we define the five-dimensional array $\mathbf{V}(\mathbf{x}) = \{\mathbf{V}^1, \ldots, \mathbf{V}^n, \ldots, \mathbf{V}^N\}$, with each $\mathbf{V}^n = \{\mathbf{V}^{n1}, \mathbf{V}^{n2}, \mathbf{V}^{n3}\}$ and $\mathbf{V}^{nl}$ defined as Eq. (A.3) indicates. Then, we proceed as follows:

1. 1D convolution along k-axis and selection of all points corresponding to control point positions at the third dimension:

$$\begin{aligned} \boldsymbol{\Upsilon}(i,j,k,n,l) &= B^{\Delta_3}(k) * \mathbf{V}(i,j,k,n,l) \\ \boldsymbol{\Upsilon}(i,j,u_3,n,l) &= \boldsymbol{\Upsilon}(i,j,p_{u_3},n,l) \end{aligned} \tag{A.7}$$

2. 1D convolution along j-axis and selection of all points corresponding to control point positions at the second dimension:

$$\begin{aligned} \boldsymbol{\zeta}(i,j,u_3,n,l) &= B^{\Delta_2}(j) * \boldsymbol{\Upsilon}(i,j,u_3,n,l) \\ \boldsymbol{\zeta}(i,u_2,u_3,n,l) &= \boldsymbol{\zeta}(i,p_{u_2},u_3,n,l) \end{aligned} \tag{A.8}$$

3. Finally, 1D convolution along i-axis and selection of all control point positions at the first dimension:

$$\begin{aligned} \boldsymbol{\Omega}(i, u_2, u_3, n, l) &= B^{\Delta_1}(i) * \boldsymbol{\zeta}(i, u_2, u_3, n, l) \\ \frac{\partial \mathcal{V}_{\boldsymbol{\Theta}}}{\partial \boldsymbol{\Theta}} &= \boldsymbol{\Omega}(u_1, u_2, u_3, n, l) = \boldsymbol{\Omega}(p_{u_1}, u_2, u_3, n, l) \end{aligned} \tag{A.9}$$

References

[1] M. Polfliet, S. Klein, W. Huizinga, M. M. Paulides, W. J. Niessen, J. Vandemeulebroucke, Intrasubject multimodal groupwise registration with the conditional template entropy, Medical Image Analysis 46 (2018) 15 – 25. doi:https://doi.org/10.1016/j.media.2018.02.003.

[2] P. Xue, E. Dong, H. Ji, Lung 4d ct image registration based on high-order markov random field, IEEE transactions on medical imaging.

[3] C. T. Metz, S. Klein, M. Schaap, T. van Walsum, W. J. Niessen, Nonrigid registration of dynamic medical imaging data using nd+ t b-splines and a groupwise optimization approach, Medical image analysis 15 (2) (2011) 238–249.

[4] M. S. Asif, L. Hamilton, M. Brummer, J. Romberg, Motion-adaptive spatio-temporal regularization for accelerated dynamic mri, Magnetic Resonance in Medicine 70 (3) (2013) 800–812.

[5] J. Royuela-del Val, L. Cordero-Grande, F. Simmross-Wattenberg, M. Martín-Fernández, C. Alberola-López, Nonrigid groupwise registration for motion estimation and compensation in compressed sensing reconstruction of breath-hold cardiac cine mri, Magnetic resonance in medicine 75 (4) (2016) 1525–1536.

[6] J.-M. Guyader, W. Huizinga, D. H. Poot, M. van Kranenburg, A. Uitterdijk, W. J. Niessen, S. Klein, Groupwise image registration based on a total correlation dissimilarity measure for quantitative mri and dynamic imaging data, Scientific reports 8 (1) (2018) 1–14.

[7] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, D. J. Hawkes, Nonrigid registration using free-form deformations: application to breast MR images, IEEE Transactions on Medical Imaging 18 (8) (1999) 712–721. doi:10.1109/42.796284.

[8] D. Rueckert, P. Aljabar, Non-rigid registration using free-form deformations, Springer US, Boston, MA, 2015, pp. 277–294. doi:10.1007/978-0-387-09749-7_15.

[9] R. Hua, J. M. Pozo, Z. A. Taylor, A. F. Frangi, Multiresolution extended free-form deformations (XFFD) for non-rigid registration with discontinuous transforms, Medical Image Analysis 36 (2017) 113 – 122. doi:10.1016/j.media.2016.10.008.

[10] J. A. Schnabel, D. Rueckert, M. Quist, J. M. Blackall, A. D. Castellano-Smith, T. Hartkens, G. P. Penney, W. A. Hall, H. Liu, C. L. Truwit, F. A. Gerritsen, D. L. G. Hill, D. J. Hawkes, A generic framework for non-rigid registration based on non-uniform multi-level free-form deformations, in: W. J. Niessen, M. A. Viergever (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2001, Springer Berlin Heidelberg, Berlin, Heidelberg, 2001, pp. 573–581. doi:10.1007/3-540-45468-3_69.

[11] W. Sun, W. J. Niessen, S. Klein, Free-form deformation using lower-order b-spline for nonrigid image registration, in: P. Golland, N. Hata, C. Barillot, J. Hornegger, R. Howe (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014, Springer International Publishing, Cham, 2014, pp. 194–201. doi:10.1007/978-3-319-10404-1_25.

[12] T. Rohlfing, C. R. Maurer, Nonrigid image registration in shared-memory multiprocessor environments with application to brains, breasts, and bees, IEEE Transactions on Information Technology in Biomedicine 7 (1). doi:10.1109/TITB.2003.808506.

[13] F. Ino, K. Ooyama, K. Hagihara, A data distributed parallel algorithm for nonrigid image registration, Parallel Computing 31 (1) (2005) 19 – 43. doi:10.1016/j.parco.2004.12.001.

[14] J. Rohrer, L. Gong, G. Székely, Parallel mutual information based 3 d non-rigid registration on a multi-core platform, in: Proceedings of the MICCAI Workshop in High-Performance. HPMICCAI, 2008.

[15] X. Du, J. Dang, Y. Wang, S. Wang, T. Lei, A parallel nonrigid registration algorithm based on b-spline for medical images, Computational and Mathematical Methods in Medicine 2016 (Article ID 7419307) (2016) 14. doi:10.1155/2016/7419307.

[16] D. Ruijters, B. M. ter Haar-Romeny, P. Suetens, Efficient GPU-accelerated elastic image registration, in: Proceedings of the 6th IASTED International Conference on Biomedical Engineering, BioMED '08, ACTA Press, Anaheim, CA, USA, 2008, pp. 419–424.

[17] M. Modat, G. R. Ridgway, Z. A. Taylor, M. Lehmann, J. Barnes, D. J. Hawkes, N. C. Fox, S. Ourselin, Fast free-form deformation using graphics processing units, Computer Methods and Programs in Biomedicine 98 (3) (2010) 278 – 284. doi:10.1016/j.cmpb.2009.09.002.

[18] D. Ruijters, B. M. ter Haar Romeny, P. Suetens, GPU-accelerated elastic 3D image registration for intra-surgical applications, Computer Methods and Programs in Biomedicine 103 (2) (2011) 104 – 112. doi:10.1016/j.cmpb.2010.08.014.

[19] N. D. Ellingwood, Y. Yin, M. Smith, C.-L. Lin, Efficient methods for implementation of multi-level nonrigid mass-preserving image registration on GPUs and multi-threaded CPUs, Computer Methods and Programs in Biomedicine 127 (2016) 290 – 300. doi:https://doi.org/10.1016/j.cmpb.2015.12.018.

[20] K. Punithakumar, P. Boulanger, M. Noga, A GPU-accelerated deformable image registration algorithm with applications to right ventricular segmentation, IEEE Access 5 (2017) 20374–20382. doi:10.1109/ACCESS.2017.2755863.

[21] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, A. V. Dalca, Voxelmorph: A learning framework for deformable medical image registration, IEEE Transactions on Medical Imaging 38 (8) (2019) 1788–1800. doi:10.1109/TMI.2019.2897538.

[22] A. V. Dalca, G. Balakrishnan, J. Guttag, M. R. Sabuncu, Unsupervised learning of probabilistic diffeomorphic registration for images and surfaces, Medical Image Analysis 57 (2019) 226 – 236. doi:10.1016/j.media.2019.07.006.

[23] T. Che, Y. Zheng, J. Cong, Y. Jiang, Y. Niu, W. Jiao, B. Zhao, Y. Ding, Deep group-wise registration for multi-spectral images from fundus images, IEEE Access 7 (2019) 27650–27661. doi:10.1109/ACCESS.2019.2901580.

[24] R. Keys, Cubic convolution interpolation for digital image processing, IEEE transactions on acoustics, speech, and signal processing 29 (6) (1981) 1153–1160.

[25] H. S. Hou, H. Andrews, Cubic splines for image interpolation and digital filtering, IEEE Transactions on Acoustics, Speech, and Signal Processing 26 (6) (1978) 508–517. doi:10.1109/TASSP.1978.1163154.

[26] T. Briand, P. Monasse, Theory and practice of image b-spline interpolation, Image Processing On Line 8 (2018) 99–141.

[27] A. Zwanenburg, S. Leger, M. Vallières, S. Löck, Image biomarker standardisation initiative (2016). arXiv:1612.07003.

[28] S. Lee, G. Wolberg, S. Y. Shin, Scattered data interpolation with multilevel b-splines, IEEE Transactions on Visualization and Computer Graphics 3 (3) (1997) 228–244. doi:10.1109/2945.620490.

[29] T. W. Sederberg, S. R. Parry, Fee-form deformation of solid geometric models, SIGGRAPH Comput. Graph. 20 (4). doi:10.1145/15886.15903.

[30] C. de Boor, A Practical Guide to Splines, Vol. 27 of Applied Mathematical Sciences, Springer-Verlag, 1978. doi:10.1007/978-1-4612-6333-3.

[31] A. Oppenheim, R. Schafer, Discrete Time Signal Processing, Prentice Hall, 1989.

[32] L. Cordero-Grande, S. Merino-Caviedes, S. Aja-Fernández, C. Alberola-López, Groupwise elastic registration by a new sparsity-promoting metric: Application to the alignment of cardiac magnetic resonance perfusion images, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (11) (2013) 2638–2650. doi:10.1109/TPAMI.2013.74.

[33] J. Royuela-del-Val, L. Cordero-Grande, F. Simmross-Wattenberg, M. Martín-Fernández, C. Alberola-López, Nonrigid groupwise registration for motion estimation and compensation in compressed sensing reconstruction of breath-hold cardiac cine MRI, Magnetic Resonance in Medicine 75 (4) (2016) 1525–1536. doi:10.1002/mrm.25733.

[34] J. Royuela-del-Val, L. Cordero-Grande, F. Simmross-Wattenberg, M. Martín-Fernández, C. Alberola-López, Jacobian weighted temporal

total variation for motion compensated compressed sensing reconstruction of dynamic MRI, Magnetic Resonance in Medicine 77 (3) (2017) 1208–1215. doi:10.1002/mrm.26198.

[35] R.-M. Menchón-Lara, J. Royuela-del-Val, A. Godino-Moya, L. Cordero-Grande, F. Simmross-Wattenberg, M. Martín-Fernández, C. Alberola-López, An efficient multi-resolution reconstruction scheme with motion compensation for 5D free-breathing whole-heart MRI, in: M. J. C. et al. (Ed.), Molecular Imaging, Reconstruction and Analysis of Moving Body Organs, and Stroke Imaging and Treatment, Vol. 10555 of LNCS, Springer International Publishing AG, 2017, Ch. 14. doi:10.1007/978-3-319-67564-0_14.

[36] R.-M. Menchón-Lara, F. Simmross-Wattenberg, P. Casaseca-de-la-Higuera, M. Martín-Fernández, C. Alberola-López, Reconstruction techniques for cardiac cine MRI, Insights into Imaging 10 (1) (2019) 100. doi:10.1186/s13244-019-0754-2.

[37] S. Sanz-Estébanez, T. Pieciak, C. Alberola-López, S. Aja-Fernández, Robust estimation of the apparent diffusion coefficient invariant to acquisition noise and physiological motion, Magnetic Resonance Imaging 53 (2018) 123 – 133. doi:https://doi.org/10.1016/j.mri.2018.07.005.

[38] S. Sanz-Estébanez, I. Rabanillo-Viloria, J. R. del Val, S. Aja-Fernández, C. Alberola-López, Joint groupwise registration and ADC estimation in the liver using a b-value weighted metric, Magnetic Resonance Imaging 46 (2018) 1 – 9. doi:https://doi.org/10.1016/j.mri.2017.10.002.

[39] S. Ahmad, J. Fan, P. Dong, X. Cao, P.-T. Yap, D. Shen, Deep learning deformation initialization for rapid groupwise registration of inhomogeneous image populations, Frontiers in Neuroinformatics 13 (2019) 34.

[40] S. Y. Chun, J. A. Fessler, A simple regularizer for b-spline non-rigid image registration that encourages local invertibility, IEEE Journal of Selected Topics in Signal Processing 3 (1) (2009) 159–169. doi:10.1109/JSTSP.2008.2011116.

[41] J. Nocedal, S. Wright, Numerical optimization, Springer Series in Operations Research and Financial Engineering, Springer Science & Business Media, 2006.

[42] R. Fletcher, C. M. Reeves, Function minimization by conjugate gradients, The Computer Journal 7 (2). doi:10.1093/comjnl/7.2.149.

[43] J. Vandemeulebroucke, S. Rit, J. Kybic, P. Clarysse, D. Sarrut, Spatiotemporal motion estimation for respiratory-correlated imaging of the lungs, Medical Physics 38 (1) (2011) 166–178. doi:10.1118/1.3523619.

[44] J. Park, S. R. Li, W. Wen, H. Li, Y. Chen, P. Dubey, Holistic sparsecnn: Forging the trident of accuracy, speed, and size, arXiv preprint arXiv:1608.01409 1 (2).

[45] X. Chen, Escoin: Efficient sparse convolutional neural network inference on gpus, Matrix 4 (5) (2019) 7–8.

[46] nVidia Corporation, Opencl best practices guide (2011).