8

NON-PARAMETRIC IMAGE REGISTRATION

The approaches discussed in the previous part are based on certain parameters. Either the transformation φ is parametric, i.e., can be expanded in terms of some basis functions, or the registration is driven by external features, e.g., the so-called *landmarks*; cf., Chapter 4. The choice of the basis functions is in general artificial and needs to be discussed and justified. The detection of landmarks is still a delicate matter and requires additional time and expert knowledge; see, e.g. Rohr (2001).

In this part, non-parametric registration techniques are to be discussed. The idea behind this type of registration is to come up with an appropriate measure both for the similarity of images as well as for the likelihood of a non-parametric transformation.

The purpose of this chapter is to set up a general framework for consideration of different registration techniques. This framework is based on a variational formulation of the registration problem, and the numerical schemes to be considered are based on the Euler–Lagrange equations which characterize a minimizer. This general concept will be specified for various registration techniques in the next chapters.

8.1 A general framework

Given two images, a reference R and a template T, we are looking for a transformation φ, such that T_φ, where $T_\varphi(x) := T(\varphi(x))$, is similar to R in a certain sense. For the following discussion it is also convenient to split the transformation φ into the trivial identity part and the so-called *deformation* or *displacement* u, $u : \mathbb{R}^d \to \mathbb{R}^d$, i.e.,

$$\varphi(x) = x - u(x), \tag{8.1}$$

where we exploit an Eulerian viewpoint; cf., Section 3.3.2. To be precise, let $\hat{x}$, $\hat{u}$, and $\hat{\varphi}$ denote the point, displacement, and transformation with respect to the Lagrange coordinates and let x, u, and φ denote the point, displacement, and transformation with respect to the Euler coordinates. Thus,

$$x = \hat{\varphi}(\hat{x}) = \hat{x} + \hat{u}(\hat{x}) \quad \text{and} \quad \hat{x} = \varphi(x) = x - u(x),$$

where $\varphi = \hat{\varphi}^{-1}$, $\hat{u}(\hat{x}) = u(x)$, and φ is assumed to be diffeomorphic.

The most intuitive way of approaching the registration problem is to design a suitable distance measure $\mathcal{D}$ and to minimize the distance between R and T_u with respect to u,

$$\mathcal{D}[R,T;u] := \mathcal{D}[R,T_u] \xrightarrow{u} \min, \tag{8.2}$$

where, in order to keep the notation simple, we introduce

$$T_u(x) := T(x - u(x)). \tag{8.3}$$

A direct minimization of the distance measure has some drawbacks: the problem is ill-posed since small changes of the input data may lead to large changes of the output data, the solution is not unique since the problem is not convex, and the deformation may not even be continuous. Thus, it is not possible to construct an appropriate scheme for a numerical solution.

Moreover, in many applications additional implicit assumptions are made on the transformation. For example, within the HNSP it is implicitly assumed that the optimal transformation is diffeomorphic. This implies that no additional cracks and/or folding of the tissue are introduced by the registration process.

The remedy in both situations is to add an additional *regularizing* term or *smoother* $\mathcal{S}$. With an appropriate smoother it becomes possible to distinguish particular transformations which seem to be more likely than others. Moreover, the regularized problem becomes a starting point for a stable numerical implementation. The choice of the smoother depends on the particular application. In many applications the desired properties of the transformation are not known a priori. Therefore, different smoothing techniques have to be provided.

Problem 8.1 *Given two images R, T, and a positive regularizing parameter $\alpha \in \mathbb{R}_{>0}$, find a deformation u, such that*

$$\mathcal{J}[u] := \mathcal{D}[R,T;u] + \alpha\mathcal{S}[u] = \min.$$

Typical choices of smoothers $\mathcal{S}$ to be discussed in the following chapters are based on bi-linear forms a, i.e.,

$$\mathcal{S}[u] = \tfrac{1}{2}a[u,u],$$

where the bi-linear forms can be traced back to the inner product in $L_2(\mathbb{R}^d)$.

A necessary condition for a minimizer u of Problem 8.1 is that the Gâteaux derivative $d\mathcal{J}[u;v]$ of $\mathcal{J}$ vanishes for all suitable perturbations v. This derivative was also known as the first variation of $\mathcal{J}$ in the direction of v.

For the Gâteaux derivative of $\mathcal{S}$, we will find

$$\begin{aligned} d\mathcal{S}[u;v] &= \lim_{h\to 0}\frac{1}{2h}\big(a[u+hv,u+hv] - a[u,u]\big) = a[u,v] \\ &= \int_{\mathbb{R}^d} \langle \mathcal{A}[u](x), v(x)\rangle_{\mathbb{R}^d}\,dx, \end{aligned} \tag{8.4}$$

where $\mathcal{A}$ is a partial differential operator. For the last equality, we apply Green's formula; cf., e.g., Buck (1978, §9.4). When appropriate, additional boundary conditions have to be imposed such that boundary integrals vanish.

Finally, for the Gâteaux derivative of $\mathcal{D}$, we will find

$$d\mathcal{D}[u;v] = \int_{\mathbb{R}^d} \langle f(x,u(x)), v(x)\rangle_{\mathbb{R}^d}\,dx,$$

where the so-called *force* f depends on the particular distance measure.

The minimization problem 8.1 has been formulated without explicit boundary conditions on a minimizer u, such that implicit boundary conditions arise naturally. However, due to our experiments, boundary conditions are of minor importance if the images are pre-registered and embedded into a uniform background. Thus, we may use explicit boundary conditions to come up with efficient numerical schemes.

Before starting the discussion of particular choices for distance measures $\mathcal{D}$ and smoothers $\mathcal{S}$ we make some general comments.

8.2 General solution schemes

A variety of different numerical schemes for the computation of a numerical solution of Problem 8.1 can be found in the literature. A first choice is a gradient-based steepest descent method. Henn (1997) used a scheme based on the gradient of the functional in Problem 8.1, while the scheme of Thirion (1995) might also be viewed as a particular gradient scheme. Here, the gradient of $\mathcal{D}$ is computed and projected in a suitable space. Higher order schemes, e.g., Newton-type schemes, cannot be recommended, since these schemes are in need of higher order derivatives of the image T, and the computation of higher order derivatives is in general time consuming and numerically unstable.

The class of approaches to be discussed here is based on the characterization of a minimizer by the Euler–Lagrange equations for Problem 8.1, which are of the form

$$\mathcal{A}[u](x) - f(x,u(x)) = 0, \quad \text{for all } x \in \Omega, \tag{8.5}$$

where $\Omega =]0,1[^d$ is the region under consideration and d denotes the spatial dimension of the images. The partial differential operator $\mathcal{A}$ is related to the smoother $\mathcal{S}$ (see also eqn (8.4)), and the *force* f is related to the distance measure $\mathcal{D}$.

A convenient way of solving this semi-linear partial differential equation and to by-pass the non-linearity is to exploit a fixed-point iteration scheme. Starting with an initial guess $u^{(0)}$ (e.g., $u^{(0)} \equiv 0$), we define $u^{(k+1)}$ implicitly by

$$\mathcal{A}[u^{(k+1)}](x) = f(x, u^{(k)}(x)), \quad x \in \Omega, \quad k \in \mathbb{N}_0. \tag{8.6}$$

The following slight modification of this iteration can be used to stabilize the scheme,

$$u^{(k+1)}(x) + \tau \mathcal{A}[u^{(k+1)}](x) = u^{(k)}(x) + \tau f(x, u^{(k)}(x)), \quad x \in \Omega, \quad k \in \mathbb{N}_0. \tag{8.7}$$

Equation 8.7 may also be rewritten as

$$\frac{u^{(k+1)}(x) - u^{(k)}(x)}{\tau} + \mathcal{A}[u^{(k+1)}](x) = f(x, u^{(k)}(x)), \quad x \in \Omega, \quad k \in \mathbb{N}_0. \tag{8.8}$$

Introducing an artificial time t, making the displacement u time dependent, $u = u(x,t)$, and setting $u^{(k)}(x) = u(x, k\tau)$, where τ denotes a fixed time-step, eqn (8.7) may also be viewed as a semi-implicit scheme for the time-dependent partial differential equation

$$\partial_t u(x,t) + \mathcal{A}[u](x,t) = f(x, u(x,t)), \quad x \in \Omega. \tag{8.9}$$

A steady-state solution of eqn (8.9) also fulfills the necessary condition for a minimizer of Problem 8.1.

For a numerical treatment of either the fixed-point-type equation (8.6) or the time-dependent partial differential equation (8.9) two problems which are both related to discretization have to be solved: the computation of the force and the numerical solution of a partial differential equation.

8.3 Computing the forces

Since our main interest is the discussion of different smoothers, we focus on the distance measure $\mathcal{D} = \mathcal{D}^{\text{SSD}}$ already introduced in Section 6.1. The forces can be deduced from its Gâteaux derivative.

Theorem 8.1 *Let $d \in \mathbb{N}$ and $R, T \in \text{Img}(d)$, $T \in C^2(\mathbb{R}^d)$, $u, v : \mathbb{R}^d \to \mathbb{R}^d$, $\Omega :=]0,1[^d$. The Gâteaux derivative of $\mathcal{D}[R,T;u]$,*

$$\mathcal{D}[R,T;u] := \mathcal{D}[R,T;u] := \tfrac{1}{2} \|T_u - R\|_{L_2(\Omega)} \tag{8.10}$$

with respect to v is given by

$$d\mathcal{D}[R,T;u;v] = -\int_{\mathbb{R}^d} \langle f(x,u(x)), v(x) \rangle_{\mathbb{R}^d}\, dx,$$

where $f : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$,

$$f(x, u(x)) := (R(x) - T_u(x)) \nabla T_u(x). \tag{8.11}$$

Proof We take advantage of a Taylor expansion of $T(x - u(x) - hv(x))$ with respect to h at the expanding point $x - u(x)$,

$$T(x - u(x) - hv(x)) = T_u(x) - h \langle \nabla T_u(x), v(x) \rangle_{\mathbb{R}^d} + \mathcal{O}(h^2).$$

Thus,

$$\begin{aligned} d\mathcal{D}&[R,T;u;v] \\ &= \lim_{h\to 0} \frac{1}{h} (\mathcal{D}[R,T;u+hv] - \mathcal{D}[R,T;u]) \\ &= \lim_{h\to 0} \frac{1}{2h} \int_\Omega (T_u(x) - h \langle \nabla T_u(x), v(x) \rangle_{\mathbb{R}^d} + \mathcal{O}(h^2) - R(x))^2 \\ &\quad - (T_u(x) - R(x))^2\, dx \\ &= \int_\Omega \langle (R(x) - T_u(x)) \nabla T_u(x), v(x) \rangle_{\mathbb{R}^d}\, dx. \end{aligned}$$

□

Since the R and T are typically digital images, an interpolation scheme has to be used to compute $T_u(x)$ for non-integer values of $u(x)$. In our implementation, a bi- or tri-linear interpolation scheme (cf., Section 3.1.3) is used in order to keep the computation time reasonable.

8.4 General remarks on choosing the discretization

Due to the simplicity of the underlying domain $\Omega =]0,1[^d$, a finite difference scheme is used to solve the partial differential equation numerically. To this end, we make use of the grids introduced in Section 3.1.2.

For a function $g : \mathbb{R}^d \to \mathbb{R}$, we use the notation $\vec{g} := g(\vec{X}) := (g(x_j))_{j=1}^N \in \mathbb{R}^N$, and for a vector field $v : \mathbb{R}^d \to \mathbb{R}^d$, we set $\vec{v} := (\vec{v_1}^\top, \dots, \vec{v_d}^\top)^\top \in \mathbb{R}^{dN}$.

From the Taylor expansion of $g : \mathbb{R}^d \to \mathbb{R}$, we get the standard finite difference approximation for second order derivatives,

$$\partial_{x_j x_j} g(x) = \frac{g(x + h_j e_j) - 2g(x) + g(x - h_j e_j)}{h_j^2} + \mathcal{O}(h_j^2), \tag{8.12}$$

$$\begin{aligned} \partial_{x_j x_k} g(x) = \frac{1}{4 h_j h_k} &(g(x + h_j e_j + h_k e_k) - g(x - h_j e_j + h_k e_k) \\ &- g(x + h_j e_j - h_k e_k) + g(x - h_j e_j - h_k e_k)) \\ &+ \mathcal{O}(h_j^2 + h_k^2), \end{aligned} \tag{8.13}$$

where in the sequel we assume a fixed meshsize h for all spatial dimensions, i.e., $h_j = h$ for $j = 1, \dots, d$. If this assumption is not fulfilled in an application, one

Algorithm 8.2 *General registration algorithm.*

Initialize $k = 0$, $\vec{X}^{(k)}$, and $\vec{U}^{(k)} = 0$.
For $k = 0, 1, 2, \ldots$
compute force $\vec{F}^{(k)} = f(\vec{X}, \vec{U}^{(k)})$;
solve partial differential equation, $A\vec{U}^{(k+1)} = \vec{F}^{(k)}$;
if converged, stop, end;
end.

has either to embed the domain in a cube or to weight the coordinates reciprocal to the resolution. A detailed description is technical and omitted here.

We assume that a finite difference approximation of $\mathcal{A}[u]$ at any grid point x_j can be obtained using a convolution filter $S^{\mathcal{A}}$,

$$\mathcal{A}[g](x_j) \approx \sum_{\kappa \in \mathcal{N}(j) \cup j} S^{\mathcal{A}}_{\kappa} g(x_\kappa) =: (S^{\mathcal{A}} * g)(x_j)$$

where $\mathcal{N}(j)$ denotes a neighborhood of the grid point j and $S^{\mathcal{A}}$ denotes a filter connected with the partial differential operator $\mathcal{A}$ via its finite difference approximation. Of course, the particular boundary conditions attached to Problem 8.1 have to be incorporated into the discrete formulation. Practically, this can be done easily by a zero padding (Dirichlet), mirroring (Neumann), or adequate copying (periodic).

With the lexicographical ordering (cf., Definition 3.2), we can also deduce a matrix representation of the convolution with $S^{\mathcal{A}}$,

$$A \cdot \vec{g} := (S^{\mathcal{A}} * g)(\vec{X}).$$

Using for example the fixed-point type-iteration (8.6), we end up with an overall algorithm as summarized in Algorithm 8.2.

9

ELASTIC REGISTRATION

Broit (1981) was the first to study what he called *elastic registration*, a method based on a quantitative measure for the deformation, a quantitative measure for similarity between deformed images, and a procedure that uses the measures for obtaining an optimal mapping.

Broit's approach is based on

$$\mathcal{D}[R, T; u] = \mathcal{D}^{\mathrm{SSD}}[R, T; u] = \tfrac{1}{2} \|T_u - R\|_{L_2(\Omega)} \quad \text{and} \quad \mathcal{S}[u] := \mathcal{P}[u]$$

where $\mathcal{P}$ denotes the *linearized elastic potential* of the displacement u,

$$\mathcal{P}[u] = \int_\Omega \frac{\mu}{4} \sum_{j,k=1}^{d} (\partial_{x_j} u_k + \partial_{x_k} u_j)^2 + \frac{\lambda}{2} (\operatorname{div} u)^2 dx. \tag{9.1}$$

Here, λ and μ denote the so-called Lamé constants. For this particular choice of regularizer, the Euler–Lagrange equations are nothing more than the Navier–Lamé equations.

Navier–Lamé equations:

$$f = \mu \Delta u + (\lambda + \mu) \nabla \operatorname{div} u. \tag{9.2}$$

In the next section we give a physical motivation for this particular regularizer. The images are viewed as two different observations of an elastic body, one before and one after a *deformation*. The *displacement* of the elastic body is derived using a linear elasticity model. This development follows Broit (1981). In order to introduce different boundary conditions and for later reference, the eigenfunctions and eigenvalues of the Navier–Lamé operator are computed in Section 9.2. In Section 9.3 we show how the Navier–Lamé equations can be derived from the elastic potential using the general framework introduced in Chapter 8. This also provides a weak form of the Navier–Lamé equations.

Based on the general remarks on discretization (see Section 8.4) we derive a finite difference approximation of eqn (9.2). For ease of presentation, we focus on dimensions $d = 2, 3$. In Section 9.5 it is shown that a general class of matrices, including the discrete Navier–Lamé operator, can be diagonalized using fast Fourier transformation (FFT) techniques. The Moore–Penrose pseudo-inverse