Abstract

This paper proposes a novel approach to optimally solve volumetric registration problems. The proposed framework exploits parametric dictionaries for sparse volumetric representations, ℓ¹ dissimilarities and DC (Difference of Convex functions) decomposition. The SAD (sum of absolute differences) criterion is applied to the sparse representation of the reference volume and a DC decomposition of this criterion with respect to the transformation parameters is derived. This permits to employ a cutting plane algorithm for determining the optimal relative transformation parameters of the query volume. It further provides a guarantee for the global optimality of the obtained solution, which – to the best of our knowledge – is not offered by any other existing approach. A numerical validation demonstrates the effectiveness and the large potential of the proposed method.

1. Introduction

Registration [1] is a fundamental problem in computer vision and in particular in medical image analysis. It is an elementary step towards bringing various volumetric data into the same reference space, which in turn permits to gather statistics and exploit similarities across subjects.

Geometric and iconic methods are often used to address this problem. Geometric methods [2] extract characteristic landmarks between two images, and then seek the optimal transformation that establishes geometric correspondences between the images. Unfortunately, such an approach may be very sensitive to the landmark extraction process. Furthermore, solving the correspondence problem between landmarks, which is a pre-step of the registration, is highly nontrivial. Often, robust EM-like methods are used for this purpose. These methods iteratively determine the optimal transformation for a set of correspondences and then improve the correspondences based on this transformation. Naturally such a method may converge to a local minimum, mostly due to erroneous correspondences.

Iconic methods [1] employ a (dis)similarity criterion on the observation space that is a function of rigid transformation parameters, which are optimized to minimize / maximize this criterion. The selection of the criterion and the optimization method are the two critical components of iconic registration. SAD, SSD, NCC, CR [3], as well as complex statistical metrics [4] in the case of multi-modal data have been considered. The optimization of the criterion is often performed using descent-like methods that are sensitive to initial conditions and do not provide guarantees on the optimality of the obtained solution. Recently the use of global optimization frameworks such as discrete MRFs was suggested [5]. However, the dimensionality of the resulting continuous search space makes its quantization quite problematic and even inefficient and therefore the results are far from being optimal.

Despite an enormous effort in the field [6], none of the existing methods can guarantee optimality of the obtained solution even in the case of volumes coming from the same modality. In this paper we propose a novel approach that estimates optimal transformation parameters. Global optimality is achieved through the expression of the objective function as a DC (difference of convex functions) decomposition and with the use of a cutting plane algorithm to estimate the optimal registration parameters (see [7] for a similar approach for two-dimensional images).

Input volumes are sparsely represented over a redundant dictionary of geometric atoms. Using such a representation, the set of all transformations of a certain volume (which constitutes the so-called transformation manifold) admits a closed form expression with respect to the transformation
parameters. This relation is used to derive a \( \ell^1 \) criterion between the two volumes in terms of the registration parameters. Using basic theorems on DC functions [8, 9, 10], we prove that the resulting objective function admits a DC decomposition with respect to the rigid transformation parameters.

Once a DC decomposition is established, a number of algorithms are available to solve the optimization problem in an efficient and robust manner [8]. In this paper, we propose a modified version of the cutting plane algorithm [8, Thm 5.3] and use it to recover the optimal registration parameters. The modifications are introduced in order to accelerate the convergence of the original cutting plane algorithm. In addition, we implement the DC function evaluation on a graphics processing unit (GPU), in order to further speed up our method. We provide numerical experiments demonstrating the effectiveness and the global optimality property of our method.

The rest of this paper is organized as follows. In Section 2, we briefly present the sparse geometric representations of volumes as well as the corresponding transformation manifolds. Section 3 is devoted to the definition of the registration problem. Next, Section 4 provides some background material on DC functions and Section 5 introduces our modified cutting plane method. We discuss our GPU implementation of the DC function evaluation in Section 6. In Section 7, we present numerical experiments of our approach, followed by some conclusions in Section 8.

2. Volume transformation manifolds

In the following, we define and characterize the transformation manifold of a certain volume. For this purpose, we represent the volume by a parametric sparse model extracted from a dictionary of geometric functions. Such a geometric representation leads to a closed form expression for the transformation manifold, which is used in the computation of \( \ell^1 \) dissimilarity measures.

2.1. Sparse atomic volumetric representations

We represent the volume of interest as a linear combination of geometric functions (usually called atoms), taken from a parametric and (typically overcomplete) dictionary \( \mathcal{D} = \{ \phi_\gamma, \gamma \in \Gamma \} \) spanning the input volume space. This representation generally captures the most prominent geometric features of the volume. The atoms in \( \mathcal{D} \) are constructed by applying geometric transformations to a generating function denoted by \( \phi \). Representing the geometric transformation \( \gamma \in \Gamma \) by an operator \( U(\gamma) \), the parametric dictionary takes the form

\[
\mathcal{D} = \{ \phi_\gamma = U(\gamma)\phi, \gamma \in \Gamma \}. \tag{1}
\]

In this work, a transformation \( \gamma = (a, R, b) \in \Gamma \), will denote a synthesis of translations \( \bar{b} \in \mathbb{R}^{3 \times 1} \), anisotropic scalings \( \bar{a} \in \mathbb{R}^{3 \times 1} \) and rotations \( R \in SO(3) \). The dictionary is built from three-dimensional atoms that can efficiently capture the salient geometrical features in volumetric images.

A sparse approximation of a given volume \( v \in \mathbb{R}^{n_1 \times n_2 \times n_3} \) with atoms from the dictionary \( \mathcal{D} \) can be obtained in various ways. Even if finding the sparsest approximation of \( v \) is generally a hard problem, effective sub-optimal solutions are usually sufficient to capture the salient geometric structures of a signal with only a few atoms. Such solutions are obtained, for example, by Orthogonal Matching Pursuit (OMP) [11, Sec. 9.5.3] and Tree-based Pursuit [12], to name just a few. In this work we use Tree-based Pursuit, which organizes the dictionary in a tree structure and admits significantly faster searches over the dictionary compared to OMP. Hence, this provides an effective algorithm for computing sparse volume approximations in practice. After \( K \) steps of the algorithm, the volume \( v \) is approximated by a sparse linear combination of a few atoms i.e.,

\[
v = \sum_{k=1}^K \xi_k \phi_k \gamma_k + r_K, \tag{2}
\]

where \( r_K \) is the residual of the approximation. In what follows we will assume that \( r_K \) is negligible and can be dropped.

2.2. Characterization of transformation manifolds

The set of all geometric transformations applied to a certain volume \( v \) generates a manifold \( \mathcal{M} \) in the high-dimensional ambient observation volume space. Each point on this manifold corresponds to a transformed version of \( v \). In the following, we only consider transformations \( \eta = (s, G, t) \) consisting of a synthesis of translations \( t = [t_x, t_y, t_z] \), isotropic scaling \( s \in \mathbb{R}_+ \) and rotations \( G \in SO(3) \). Then the transformation manifold \( \mathcal{M} \) can be expressed as follows:

\[
\mathcal{M} = \{ v(\eta) \equiv U(\eta)v, \text{ where } \eta = (s, G, t) \}. \tag{3}
\]

Note that although the manifold is embedded in a high-dimensional space, its intrinsic dimension is rather small and equals the number of transformation parameters.

The transformations \( \eta \) form a group, namely the similitude group \( \text{SIM}(3) \) in \( \mathbb{R}^3 \). If \( (a, R, b) \) and \( (a', R', b') \) are two elements from \( \text{SIM}(3) \) then the group law is

\[
(a, R, b) \circ (a', R', b') = (aa', RR', b + aRb'). \tag{4}
\]

Using (2) and dropping the residual term \( r_K \), it turns out that applying the transformation \( \eta \) to the volume \( v \) results in

\[
v(\eta) = U(\eta)v = \sum_{k=1}^K \xi_k U(\eta)\phi_k = \sum_{k=1}^K \xi_k \phi_k \circ \gamma_k, \tag{5}
\]

2450
where $\eta \circ \gamma_k$ is a product of transformations. In other words, the transformation is applied to each constituent atom individually, resulting in a sparse representation of the transformed volume over atoms with updated parameters. The group law (4) indeed applies [13] and can be further employed to work out the updated parameters of the transformed atoms. Equation (5) is of great importance in the proposed approach, since it expresses the manifold (3) in closed form with respect to the transformation parameters $\eta$. This is a key observation for the applicability of the DC programming methodology that is proposed in this work.

3. Rigid registration

After having introduced sparse geometric representations and transformation manifolds, we are now ready to provide the problem formulation. We are interested in estimating the optimal transformation parameters $\eta^*$ that best align $v$ with $p$. We formulate the transformation estimation problem as follows

$$\eta^* = \arg \min_{\eta \in \mathcal{M}(s,G,\ell)} f(\eta), \text{ where } f(\eta) = \|v(\eta) - p\|_1. \quad (6)$$

Here, $\|p\|_1 = \sum_{ijk} |p_{ijk}|$ denotes the $\ell^1$ norm of a volume $p \in \mathbb{R}^{n_1 \times n_2 \times n_3}$. The criterion (6) is also known as the sum of absolute differences (SAD) criterion.

Recall that $v(\eta) \in \mathcal{M}$ denotes the transformed volume $v$ subject to a transformation $\eta = (s, G, t)$. We assume that the reference volume $v$ has been well approximated by a sparse expansion over $\mathcal{D}$ according to (2), where $r_K$ is negligible. Note that in the above optimization problem, only the reference volume $v$ is expanded in the redundant basis and the query volume $p$ is treated as is.

The optimization problem (6) is generally a non-convex nonlinear optimization problem [14] and hard to solve using traditional methods. For example, steepest descent or Newton-type methods converge only locally and may get trapped in local minima. To avoid these issues, we will exploit that the above objective function is a DC function with respect to the transformation parameters, i.e., it can be expressed as the difference of two convex functions.

Theorem 1 The objective function

$$f(\eta) = \|v(\eta) - p\|_1 = \left\| \sum_{k=1}^{K} \xi_k \phi_{\eta_k} - p \right\|_1, \quad (7)$$

where $\eta_k = \eta \circ \gamma_k$, is DC.

The proof of this theorem can be found in the companion technical report SAM-2011-181 [15]. The proof is constructive and provides a procedure for evaluating the two convex parts of $f$. Using Theorem 1, the optimization problem (6) can be formulated as a DC program [8, 9, 10], which can be optimally solved by exploiting the special structure of the objective function. In this paper, we employ a cutting plane method to solve the DC formulation of (6). The proposed method is guaranteed to converge to the global minimizer. To the best of our knowledge, this is the first globally optimal algorithm that is proposed for the problem of rigid volume registration.

4. DC functions

Before we introduce the proposed algorithm, we start with some background material about DC functions [8, 9, 10]. Let $X$ be a convex subset of $\mathbb{R}^n$. A function $f : X \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$ is called DC [8, 9, 10] on $X$, if there exist two convex functions $g, h : X \rightarrow \mathbb{R}$ such that $f$ is expressed as

$$f(x) = g(x) - h(x). \quad (8)$$

A representation of the above form is called a DC decomposition of $f$. The DC decomposition of $f$ is not unique, since one can obtain a different decomposition by adding the same convex function $c(x)$ in both convex parts of it. We provide below an illustrative example of a DC function.

Example 1 Consider the function $f(x) = \cos(x)$, $x \in [0, 2\pi)$ and suppose that we want to determine a DC decomposition of it. In other words, we seek two convex functions $g(x)$ and $h(x)$ such that $f(x) = g(x) - h(x)$. We know from Taylor’s theorem that

$$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots$$

Grouping together the terms of the same sign, one obtains

$$\cos x = \left[1 + \frac{x^4}{4!} + \ldots\right] - \left[\frac{x^2}{2!} + \frac{x^6}{6!} + \ldots\right], \quad (9)$$

which readily provides a DC decomposition of $f(x)$, since $g(x)$ and $h(x)$ are convex functions. The obtained decomposition is illustrated in Fig. 1.

The example above shows that even a highly non-linear and non-convex function such as the cosine, can be decomposed into two convex parts giving rise to a special structure that can be further exploited for global optimization. We present now a few properties of DC functions.

Proposition 1 [9, Sec 4.2] Let $f = g - h$ and $f_i = g_i - h_i$, $i = 1 \ldots, m$ be DC functions. Then the following functions are also DC:

for more details). In the next section we discuss the powerful enough to show that

Let

Proposition 2

larly important for our further developments.

(a) \[ \sum_{i=1}^{m} \lambda_i f_i = \left[ \sum_{\{i: \lambda_i \geq 0\}} \lambda_i g_i - \sum_{\{i: \lambda_i < 0\}} \lambda_i h_i \right] - \left[ \sum_{\{i: \lambda_i \geq 0\}} \lambda_i h_i - \sum_{\{i: \lambda_i < 0\}} \lambda_i g_i \right]. \]

(b) \[ |f| = 2 \max\{g, h\} - (g + h). \]

(c) If \( f_1 \) and \( f_2 \) are DC functions, then the product \( f_1 \cdot f_2 \) is DC. Moreover, if \( f_1 \) and \( f_2 \) have nonnegative convex parts, the following DC decomposition holds:

\[ f_1 \cdot f_2 = \frac{1}{2} [(g_1 + g_2)^2 + (h_1 + h_2)^2] - \frac{1}{2} [(g_1 + h_2)^2 + (g_2 + h_1)^2]. \] (10)

In addition, it can be shown that the synthesis of a convex function and a DC function is again DC, which is particularly important for our further developments.

**Proposition 2** Let \( f(x) : \mathbb{R}^n \to \mathbb{R} \) be DC and \( q : \mathbb{R} \to \mathbb{R} \) be convex. Then,

(a) the composition \( q(f(x)) \) is DC [9, Sec 4.2].

(b) \( q(f(x)) \) has the following DC decomposition:

\[ q(f(x)) = p(x) - K[g(x) + h(x)], \] (11)

where \( p(x) = q(f(x)) + K[g(x) + h(x)] \) is a convex function and \( K \) is a constant satisfying \( K \geq |q'(f(x))|/16, 7]. \)

When combined together, the above properties are powerful enough to show that \( f \) in (7) is DC (see the proof in [15] for more details). In the next section we discuss the proposed algorithm for solving the DC formulation of (6).

5. Proposed cutting plane method

An optimization problem is called a DC program if it takes the form

\[
\min_x \quad f(x) = g(x) - h(x), \\
\text{s.t.} \quad x \in X = \{x \in \mathbb{R}^n : \delta(x) \leq 0\},
\]

where \( g, h : X \to \mathbb{R} \) are convex functions and \( \delta : \mathbb{R}^n \to \mathbb{R} \) is a convex function. Denote by \( \omega^* \) the global minimum of (12). The next proposition provides an optimality condition for (12).

**Proposition 3** ([8]) The point \( x^* \in X \) is an optimal solution to the DC problem (12) if and only if there exists \( t^* \in \mathbb{R} \) such that

\[
0 = \inf \{-h(x) + t : x \in X, t \in \mathbb{R}, \quad g(x) - t \leq g(x^*) - t^*\}. \] (13)

One may solve optimally the DC program (12) using the cutting plane method of [8, Thm 5.3], which is briefly discussed in the sequel. The cutting plane algorithm seeks a point \( x^* \) that satisfies the global optimality condition (13). Each iteration involves the minimization of the concave function \( -h(x) + t \) over a convex constraint set of the form \( C_k := \{g(x) - t \leq \omega_k\} \), where \( \omega_k \) is the best upper bound for \( \omega^* \) as of iteration \( k \). The cutting plane method uses a polytope that provides an outer approximation of the set \( C_k \); hence, the minimizer \( x^k \) of the concave function \( -h(x) + t \) can be readily found at one of the extreme vertices of the polytope. A cutting plane is defined based on \( x^k \) and further used to refine the polytope by excluding points that are guaranteed to violate the constraints \( C_k \). The same process is repeated in the next iteration. Once a better bound \( \omega_k \) of \( \omega^* \) has been found in the course of the algorithm, the constraint set \( C_k \) is updated. More details about the cutting plane method can be found in [8, Thm 5.3].

We introduce two modifications to the standard cutting plane method described above, in order to accelerate its
Algorithm 1 Modified Cutting Plane Algorithm

1: Initialization: Set \( \omega^0 = g(y^0) - h(y^0) \), the first upper bound of the optimal value \( \omega^* \) of the Problem (12).
2: Construct a polytope \( P^0 \) that contains \( \{(x, t) : x \in X, \ t \in \mathbb{R}, \ g(x) - t - \omega^* = 0 \} \).
3: Compute the vertex set \( V(P^0) \) of the polytope \( P^0 \).
4: \( f_{\min}^0 = \infty \).
5: Set \( k = 0 \).
6: Iteration:
   7: Compute the minimizer \( (x^k_1, t^k_1) \) and second best minimizer \( (x^k_2, t^k_2) \) of the problem:
      \[
      \min \{-h(x) + t : (x, t) \in V(P^k)\}.
      \]
   8: if \( -h(x^k_i) + t^k_i = 0 \) then
      9: \( y^k \) is the optimal solution with optimal value \( \omega^k \).
   else
   10: Compute \( s^k_i \in \partial g(x^k_i), \ i = 1, 2 \).
   11: \( f_{\min}^{k+1} = \min\{g(x) - h(x) : (x, t) \in V(P^k)\} \).
   12: if \( f_{\min}^{k+1} < f_{\min}^k \) then
      13: Identify \( \bar{x}^k \) such that \( f_{\min}^{k+1} = f(\bar{x}^k) \).
      14: Compute \( \bar{y}^k \) such that \( f_{\min}^{k+1} = \text{subgradientDescent}(\bar{x}^k) \).
      15: Compute the improved upper bound \( \omega^{k+1} = \min\{\omega^k, f_{\min}^{k+1}\} \).
      16: Update \( y^{k+1} \) such that \( g(y^{k+1}) - h(y^{k+1}) = \omega^{k+1} \).
      17: Construct the cutting planes:
         \[
         l^k_i(x, t) = (x - x^k_i)^T s^k_i + g(x^k_i) - \omega^{k+1} - t, \ i = 1, 2.
         \]
      18: Set \( P^{k+1} = P^k \cap \{(x, t) : l^k_1(x, t) \leq 0, l^k_2(x, t) \leq 0\} \) and compute \( V(P^{k+1}) \).
      19: Set \( k = k + 1 \) and go to step 6.

Convergence. The main steps of the modified cutting plane method are summarized in Algorithm 1.

- First, we use two cutting planes in each iteration in order to make the cut more effective. To define the two cutting planes, we use the minimizer and the second best minimizer\(^2\) of \( -h(x) + t \) over the polytope (see also Lines 7 and 19). Empirically we found no further improvement by using more than two cutting planes; thus, we employ two cutting planes in our experiments.

- Second, we employ subgradient descent as a means to further improve the bound for \( \omega^* \) (see also Lines 12-17). Experimental evidence suggests that this modification often greatly accelerates the convergence of the method. It is important to emphasize that subgradient descent has only an auxiliary role: it is only used as a means to provide improved bounds \( \omega^{k+1} \) to the global minimizer \( \omega^* \) in the early iterations of the algorithm. Of course, the global optimality property of the proposed modified cutting plane method still holds, and this is verified in practice as will be shown below in the numerical experiments.

6. Computational aspects

Evaluating the DC decomposition of the objective function \( f \) scales proportionally with \( K \cdot n_1 \cdot n_2 \cdot n_3 \), due to the fact that the DC decomposition needs to be evaluated for each voxel of the \( K \) atoms of size \( n_1 \times n_2 \times n_3 \). In order to accelerate the computational speed, we perform the function evaluation on the graphics processing unit (GPU). In what follows, we provide an overview of the GPU implementation.

**GPU architecture and CUDA.** We make use of a CUDA enabled GPU; more specifically, we used the NVIDIA Tesla C1060. This processor consists of 30 multiprocessors, with each of them containing a collection of 8 CUDA cores. Following the SIMD model, a CUDA function, called kernel, is executed simultaneously by each thread on every CUDA core, i.e., there are 240 threads for the Tesla C1060. Threads are organized in blocks and the blocks in turn are organized in a grid.

The memory of a CUDA device is organized in a hierarchical way. All threads have access to the global memory, threads within a block share an on-chip, fast memory space called shared memory and each thread has its own private memory space and registers. We refer to the CUDA Programming Guide for further details.

**GPU implementation.** Since the evaluation of the DC decomposition is not only the computationally most expensive task but also well structured and highly parallelizable, it is well suited for GPU implementation. To compute this decomposition for a vertex in the polytope generated by the cutting plane method, the decomposition is evaluated at every voxel and summed up. In our GPU implementation, the loops over the vertices and the voxels are parallelized, leaving only one loop over the atoms in CPU code. A 2-D grid of blocks is created, with the blocks in the \( i \)th grid row evaluating the \( i \)th vertex. Each thread evaluates the decomposition of one voxel. The final summation for each vertex is performed using a parallel reduction algorithm as described in [17]. Vectors like \( \mu, \nu \) and \( \zeta \) (see Lemma 1 in the companion report [15]) that are required for the decomposition of every vertex are computed once and stored in the global memory, along with the sparse representation of the volume. Since not all vertices may fit into global

\(^2\)Actually, we mean the next possible minimizer which is not removed by the cutting plane of the best minimizer.
Table 1. Average timings of CPU and GPU implementations.

<table>
<thead>
<tr>
<th>volume size</th>
<th>CPU</th>
<th>GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>128x128x10</td>
<td>12.1 sec</td>
<td>390ms</td>
</tr>
<tr>
<td>64x64x20</td>
<td>5.2 sec</td>
<td>190ms</td>
</tr>
<tr>
<td>32x32x20</td>
<td>2.5 sec</td>
<td>60ms</td>
</tr>
</tbody>
</table>

Benchmarks. We compare the execution of the CPU and GPU implementations using an Intel Xeon X5570 (2.93 GHz) and the NVIDIA Tesla C1060, respectively. For the benchmarks we used volumes of sizes 32 × 32 × 20, 64 × 64 × 20 and 128 × 128 × 10, decomposed into K = 50 atoms. The comparison of the average time needed for evaluating the DC decomposition is shown in Table 1, demonstrating a remarkable speedup.

7. Numerical experiments

We apply the method proposed in this work to medical imaging. For this purpose, we use MRI calf muscle volumes, which have been obtained with 0.7812x0.7812x7 mm voxel spacing using a 1.5T Siemens scanner. Each volume slice is 7mm thick. Figure 2 shows a sample volume used in our experiments and its sparse representation with 1000 atoms. It is important to emphasize that for the purposes of rigid alignment, one does not actually need an accurate sparse approximation of the reference shape [7]. In practice, relatively few atoms providing a crude shape approximation are typically sufficient for successful alignment.

7.1. 5D parameter space

In the first experiment, we consider transformations η = (q₀, q₁, q₂, q₃) consisting of five parameters, where q₀, q₁, q₂, q₃ are the quaternion parameters of the 3D rotation and s is the (isotropic) scaling. The reference volume consists of a 128 × 128 × 10 sparse representation of the calf volume with K = 50 atoms, and it is to be aligned with a query volume that is a rotated and scaled version of it. The ground-truth transformation parameters are η* = (q₀, q₁, q₂, q₃, s) = (−0.73, −0.36, 0.54, 0.18, 0.8). We give the query volume as input to the proposed cutting plane method which converged to the exact transformation parameters η* in 25 iterations. We did the same alignment experiment using subgradient descent that converged to the estimate (q₀, q₁, q₂, q₃, s) = (0.11, 0.55, −0.66, 0.5, 0.8), which is clearly suboptimal. Hence, we verify in practice the global optimality properties of our algorithm and its superiority over subgradient descent that is typically trapped in a local minimum.

Next, for the sake of completeness, we do the same experiment with 64 × 64 × 20 volume sizes and K = 100 atoms in the sparse representation of the reference volume. We used the same ground-truth transformation parameters η* as above. Our cutting plane method converged to η* in 15 iterations, whereas subgradient descent converged to (q₀, q₁, q₂, q₃, s) = (0.67, −0.23, 0.64, −0.3, 0.79), which is suboptimal as well.

7.2. 8D parameter space

In the sequel, we consider a full transformation η = (q₀, q₁, q₂, q₃, t₁, t₂, t₃, s) consisting of eight parameters: q₀, q₁, q₂, q₃ are the quaternion parameters of the 3D rotation, t₁, t₂, t₃ are the translation parameters and s is the (isotropic) scaling. In our experiment, we used η* = (−0.963, 0.053, −0.03, 0.26, 3, 2, 1, 0.8) as a ground-truth transformation. The sparse representation of the reference volume consists of K = 20 atoms of size 32 × 32 × 20. After 1003 iterations, the cutting plane provided the following transformation estimate: η̂ = (−0.969, 0.051, −0.033, 0.24, 2.91, 2.26, 0.25, 0.798).

Observe that the scaling and rotation parameters are almost exact and the translation parameters have been estimated with sub-pixel accuracy. Hence, the estimate from our method is indeed very close to the exact transformation η* . On the contrary, the estimate obtained from subgradient descent was found to be (−0.028, −0.027, 0.102, −0.994, −3.35, −3.00, 3.18, 0.8), which is again far from the optimal solution.
7.3. Warped query volumes

So far, we have only considered query volumes that have been constructed by applying a transformation on the sparse representation of the reference volume. In this experiment, we consider warped query volumes, where the intensity value at each transformed voxel position is obtained with nearest neighbor interpolation. This setup makes the alignment experiment even more challenging due to the involved approximation error, since the query volume does not exactly represent a transformed version of the sparse approximation of the reference volume.

We try to align a $32 \times 32 \times 20$ sparse representation of the reference volume with $K = 100$ atoms with a warped query volume corresponding to transformation $\eta^* = (q_0, q_1, q_2, q_3, s) = (0.924, 0.125, -0.327, 0.151, 1.2)$. After 107 iterations, our cutting plane method provided the estimate $\tilde{\eta} = (0.927, 0.127, -0.321, 0.147, 1.15)$ that is very close to the optimal transformation $\eta^*$. Hence, our method was still able to compute the optimal solution, despite the fact that the query volume does not exactly lie on the transformation manifold of the sparse reference volume.

Next, we explore the robustness of the method against different sparse representations of the reference volume. This also helps us study the impact of the information loss due to the sparse representation. To this end, we repeated the above experiment with different sparse representations obtained with increasing number of atoms. The obtained results are shown in Table 2, where the End Point Difference (EPD) corresponds to the distance between the optimally transformed voxel position and the one obtained with the estimated transformation. The Angular Error (AE) corresponds to the angle between them when treated as three-dimensional vectors. Note that the results for $K = 100$ atoms have been already reported above.

![Figure 3. Atlas obtained from the two volumes: before and after the alignment.](image)

<table>
<thead>
<tr>
<th>Atoms</th>
<th>Estimated transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>(0.934, 0.131, -0.317, 0.092, 1.142)</td>
</tr>
<tr>
<td>30</td>
<td>(0.920, 0.126, -0.339, 0.143, 1.157)</td>
</tr>
<tr>
<td>50</td>
<td>(0.922, 0.118, -0.332, 0.156, 1.154)</td>
</tr>
<tr>
<td>100*</td>
<td>(0.927, 0.127, -0.321, 0.147, 1.152)</td>
</tr>
<tr>
<td>200</td>
<td>(0.927, 0.125, -0.321, 0.146, 1.156)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Atoms</th>
<th>Mean EPD (std)</th>
<th>Mean AE (std)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1.3508 (0.4953)</td>
<td>0.1032 (0.0222)</td>
</tr>
<tr>
<td>30</td>
<td>0.4917 (0.1565)</td>
<td>0.0233 (0.0070)</td>
</tr>
<tr>
<td>50</td>
<td>0.4822 (0.1485)</td>
<td>0.0150 (0.0053)</td>
</tr>
<tr>
<td>100*</td>
<td>0.4846 (0.1491)</td>
<td>0.0117 (0.0035)</td>
</tr>
<tr>
<td>200</td>
<td>0.4624 (0.1429)</td>
<td>0.0130 (0.0037)</td>
</tr>
</tbody>
</table>

Table 2. Performance of the method with different sparse representations.

Observe that once the sparse representation of the reference volume becomes reasonable (e.g., more than 30 atoms), the solution is very close to the optimal one. Although increasing the number of atoms reduces the approximation error, it does not really influence the estimated transformation which remains very close to the optimal transformation (notice that in all cases the error is below voxel accuracy).

A few remarks are in order. First, we confirm that for the purpose of linear registration, where one seeks an optimal global estimation of the parameters, a very accurate approximation of the reference volume is not necessary. Second, although the sparse representation of the reference volume induces an information loss, this loss does not harm the optimality of the method, provided that the representation sufficiently captures the pattern shape. Moreover, there is no issue with the multiple approximations due to the redundancy of the dictionary. As a last note we emphasize that the approximation does not even have to be sparse. The sparsity of the representation can only impact the computational cost of the function evaluation, and it cannot compromise the global optimality of the obtained solution.
7.4. Alignment between two different volumes

In this last experiment, we consider the even more challenging case where another volume (i.e., from a different person) is used as a query volume, and we try to align it with the reference volume. In this case, the optimal transformation is unknown. The sparse representation of the reference volume consists of $K = 50$ atoms of size $128 \times 128 \times 20$. After 1416 iterations of the cutting plane method, the obtained transformation estimate is $\hat{\eta} = (-0.998, 0.01, 0.007, 0.051, 1.087)$, implying that the respective rotation is very small and that the query volume is slightly larger than the reference volume (see also Fig. 3). The Dice’s coefficient has increased from 0.71 (before alignment) to 0.77 (after alignment), which is very satisfactory for a rigid alignment method.

8. Conclusions

We have proposed a globally optimal method for rigid registration between volumetric images by transformation parameter estimation. The proposed methodology is based on sparse volumetric representations. We have shown that under such a representation, the $\ell^1$ similarity is a DC function of the transformation. The rigid registration problem becomes then equivalent to a DC optimization problem that can be optimally solved. We have proposed a modified cutting plane method for computing the globally optimal solution to the above problem. Finally, we have presented experimental results that (i) verify in practice the global optimality property of the method (ii) demonstrate its superiority over other descent-type of methods and finally show its large potential.

9. Acknowledgements

The first author would like to thank Prof. Pascal Frossard for his valuable feedback regarding the paper.

References


