Discrete Applied Mathematics 216 (2017) 461–481

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Discrete rigid registration: A local graph-search approach[☆]

Phuc Ngo[a,∗], Yukiko Kenmochi[b], Akihiro Sugimoto[c], Hugues Talbot[b], Nicolas Passat[d]

[a] *Université de Lorraine, LORIA, France*
[b] *Université Paris-Est, LIGM, CNRS, France*
[c] *National Institute of Informatics, Japan*
[d] *Université de Reims Champagne-Ardenne, CReSTIC, France*

A R T I C L E I N F O

Article history:
Received 2 April 2015
Received in revised form 22 April 2016
Accepted 5 May 2016
Available online 11 June 2016

Keywords:
Image registration
Discrete rigid transformation
Discrete optimization
Discrete rigid transformation graph
Graph search

A B S T R A C T

Image registration has become a crucial step in a wide range of imaging domains, from computer vision to computer graphics. The core of image registration consists of determining the transformation that induces the best mapping between two images. This problem is ill-posed; it is also difficult to handle, due to the high size of the images and the high dimension of the transformation parameter spaces. Computing an actually optimal solution is practically impossible when transformations are assumed continuous (i.e., defined on $\mathbb{R}^n$). In this article, we initiate the exploration of a new way of considering image registration. Since digital images are basically defined in a discrete framework (i.e., in $\mathbb{Z}^n$), the transformation spaces – despite a potentially high complexity – actually remain finite, allowing for the development of explicit exploration of the parameter space via discrete optimization schemes. We propose an analysis of the very basis of registration, by considering rigid registration between 2D images. We show, in particular, how this problem can be handled in a fully discrete fashion, by computing locally the combinatorial structure of the parameter space of discrete rigid transformations, and by navigating on-the-flight within this space via gradient descent paradigms. This registration framework is applied in real imaging cases, emphasizing the relevance of our approach, and the potential usefulness of its further extension to higher dimension images and richer transformations.

1. Introduction

Image processing and analysis applications are involving an increasing amount of images, which result from several total/partial acquisitions of a same scene or different scenes of same semantics, from different viewpoints and/or times. This multiplicity of data related to a same structure of interest has motivated the development, during the last twenty years, of a novel branch of image processing, devoted to image registration [24]. The most frequent application cases of image registration can be found (non-exhaustively) in medical imaging for inter- or intraindividual mapping from multimodal/multitime acquisition [18], in remote sensing for image orthorectification [20], in computer vision for stereovision or augmented reality.

☆ The research leading to these results has received funding from the French *Agence Nationale de la Recherche* (Grant Agreement ANR-2010-BLAN-0205) and the *Programme d'Investissements d'Avenir* (LabEx Bézout, ANR-10-LABX-58).

∗ Corresponding author. Tel.: +33 354958430; fax: +33 383278319.
E-mail address: hoai-diem-phuc.ngo@loria.fr (P. Ngo).

http://dx.doi.org/10.1016/j.dam.2016.05.005
0166-218X/© 2016 Elsevier B.V. All rights reserved.

Basically, registration consists of mapping "at best" a source image I_s onto a target image I_t. This implies that I_s and I_t have equal – or at least similar – semantic contents. In general, registration techniques involve searching over the parameter space $\mathbb{T}$ of certain geometric transformations to find the optimal transformation that maps I_s onto I_t. In this context, registration is indeed interpreted as an optimization problem. In addition, it is generally modelled as a continuous problem; in other words, $\mathbb{T}$ is considered as a subset of $\mathbb{R}^n$. In particular, this continuous paradigm induces an infinity of possible transformations, and the chosen optimization schemes have to handle this issue.

Our working hypothesis is that, since the considered images are generally discrete, and more precisely digital (i.e., defined in $\mathbb{Z}^k$ mainly with $k = 2$ or 3), a relevant optimization scheme may consist of navigating within the associated (discrete) parameter space of the transformations from $\mathbb{Z}^k$ to $\mathbb{Z}^k$. In particular, even if such parameter space is huge, it may be computed and explored on-the-flight, leading to combinatorial strategies for determining exact locally optimal solutions, or estimating globally optimal ones.

In this article, we initiate the study of this paradigm, by considering the basic case of rigid transformations between 2D images, i.e., images defined on $\mathbb{Z}^2$. In the case of rigid transformations, that are obtained by composition of translations and rotations, the parameter space has three dimensions: one for each principal direction of $\mathbb{Z}^2$, and a third for the rotation angle. The overall parameter space, that is a (finite) subdivision of a subset of $\mathbb{R}^3$, has a polynomial complexity [14], and thus cannot be computed as a whole, in practice. Nevertheless, it can be locally computed as a graph. Such a computation, involving a local exploration of the graph, can be integrated into a gradient descent scheme, which leads to a local optimum, for solving the image registration issue. In particular, a multi-scale scheme is proposed in order to speed-up the optimization process. This study then constitutes a proof of concept for a general combinatorial way of considering image registration.

The remainder of this article – that is an extended and improved version of the conference papers [16,7] – is organized as follows. Section 2 introduces and formalizes the optimization problem of image registration, and our specific purpose, namely discrete image registration. Section 3 describes the parameter space of rigid transformations and the way it can be expressed as a combinatorial structure (named a DRT graph [14]) when handling discrete images. The main contributions of this article can be found in Sections 4 and 5, in which we describe a discrete exploration procedure using neighbouring relations defined on DRT graphs to solve rigid registration issues. Experiments and a conclusion are proposed in Sections 6 and 7, respectively.

2. Image registration

In general, registration consists of estimating the optimal spatial transformation so that the source and target images are aligned. In other words, it aims to solve an optimization problem expressed as minimizing a metric error between two images, i.e.

$$\mathcal{T}^* = \arg\min_{\mathcal{T}\in\mathbb{T}} d((I_s \circ \mathcal{T}), I_t) \tag{1}$$

where the objective function d to minimize is the distance (dissimilarity) between images. The transformation $\mathcal{T}^*$ is then searched among the space $\mathbb{T}$ of authorized transformations $\mathcal{T}$, defined via the degrees of freedom on their parameters.

Over the years, many approaches have been proposed to register images, each of which is usually designed for specific applications and types of data. All of them practically rely on the following three key-points.

1. "What": the *distance between images* quantifies how much these images are similar, from a spatial/semantic point of view. In the literature, several distances have been considered for image registration. Depending on the density [6]/sparseness of images, on the one hand, and intensity-based or feature-based [10] paradigms, on the other hand, many distances have been introduced, e.g., cross-correlation, mutual information, ratio image uniformity, least square difference, signed distance, etc. Detailed discussions of these distances can be found, e.g., in [24,3].
2. "Where": the *geometric transformations* that are assumed valid to register the source image onto the target image. The most common transformations are rigid, affine, projective, perspective and global. The determination of these transformations is generally guided by the application and the content of the images. The range is wide, from rigid to non-rigid transformations; the more complex the transformation, the higher the associated parameter space $\mathbb{T}$ (leading to systems that involve from 2 or 3 to thousands of freedom degrees). In addition to geometrical issues, the most complex transformations also induce topological difficulties [17].
3. "How": the *optimization scheme* provides the modus operandi for estimating the optimal distance by exploring the parameter space of geometric transformations. The choice of an optimization technique relies, in particular, on the nature of the distance function and the parameter space of transformations. Various continuous optimization schemes have been employed, such as gradient descent [8], conjugate gradient descent [23], Newton-type methods [2], etc. Due to the ill-condition of the problem, most of these approaches use iterative models of optimization, which consist of performing local search in the parameter space of rigid transformations to find the minima. However, such approaches limit their use in two ways: (1) they strongly rely on the computation of the gradient of the optimizing function, thus this function must be differentiable; (2) they are sensitive to the initial solution and tend to reach local minima. Furthermore, since the problem of image registration is considered in a continuous space, the points of images are assumed to be real values – despite the fact that images are defined in a discrete space. In addition, since the transformation space $\mathbb{T}$ is continuous, all possible rigid transformations cannot be reached. More precisely, a sampling process is necessary to explore the search

space, potentially biasing to the final result. In particular, the more densely the solution space is sampled, the better accuracy we may achieve, at expense of a higher computational cost. Nonetheless, there is no guarantee that the obtained result is *exact* with respect to Eq. (1). Discrete schemes (e.g., graph-cut methods [22], particle swarm optimizations [21]) rely on alternative paradigms, where the search space is huge, but finite. Still, the question raised when using these discrete optimizations is how to explore such search space.

We mainly consider this specific issue of discrete schemes for registration, directly related to the third item of the above list. In this context, we focus on the case of 2D images and rigid transformations, i.e., we fix some hypotheses related to the second item. The case of rigid transformations constitutes the simplest – yet non-trivial – case of image registration. This will allow us to emphasize the basics of our combinatorial approach, and provide a sound basis that may be further extended to higher dimension images and transformations. Finally, concerning the distance d for the dissimilarity between images, the choice is – in theory – independent from our optimization framework. Nevertheless, the time cost of distance computation and update has a direct impact on the time cost of the whole process. As a consequence, we will rely preferentially on distances whose local update can be carried out in constant time. The signed distance transform [3], involved in the experiments of Section 6, is an example of such distance.

In our recent works, a fully discrete framework [14] was proposed to model the space $\mathbb{T}$ of rigid transformations over the discrete 2D images. It was proved that the 3D parameter space of the induced discrete rigid transformations is isomorphic to a combinatorial structure, namely a *graph*. Navigating within this – so-called Discrete Rigid Transformation (DRT) – graph then allows us to explicitly explore all the potential transformations, by iteratively considering "neighbouring" transformations, that differ pairwise by exactly one point of $\mathbb{Z}^2$.

In the next sections, we show how such navigation can be carried out in a standard gradient descent paradigm with a low computational cost, by computing on-the-flight the useful part of the parameter space, without building its whole – polynomial size – structure, thus leading to efficient discrete registration strategies.

3. Rigid transformations on $\mathbb{Z}^2$ and DRT graph

In order to make this article self-contained, we recall in this section some basic notions of [14], dealing with rigid transformations defined on $\mathbb{R}^2$ and $\mathbb{Z}^2$ (Section 3.1), the induced parameter space (Section 3.2) and the way this space can be modelled by a combinatorial model (Section 3.3).

3.1. Digital rigid transformations

A 2D rigid transformation is defined as a rotation composed with a translation. In the continuous framework, such a transformation can be formally expressed as a bijective function $\mathcal{T} : \mathbb{R}^2 \to \mathbb{R}^2$ such that for any $\mathbf{x} = (x, y) \in \mathbb{R}^2$, the transformed point $\mathcal{T}(\mathbf{x})$ has the form

$$\mathcal{T}(\mathbf{x}) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \tag{2}$$

where the parameters $a_1, a_2 \in \mathbb{R}$ represent the translation, and $\theta \in [0, 2\pi[$ is the rotation angle. In particular, such a transformation is unambiguously modelled by the triplet of parameters (a_1, a_2, θ), and will be often denoted by $\mathcal{T}_{a_1a_2\theta}$. When applied to an image $I : \mathbb{R}^2 \to \mathbb{V}$, it provides a new transformed image $I \circ \mathcal{T} : \mathbb{R}^2 \to \mathbb{V}$, where $\mathbb{V}$ is a finite set of value space. (As examples, if $|\mathbb{V}| = 2$, we say that I is a binary image; if $|\mathbb{V}|$ is equipped with a total order, we say that I is a grey-level image.)

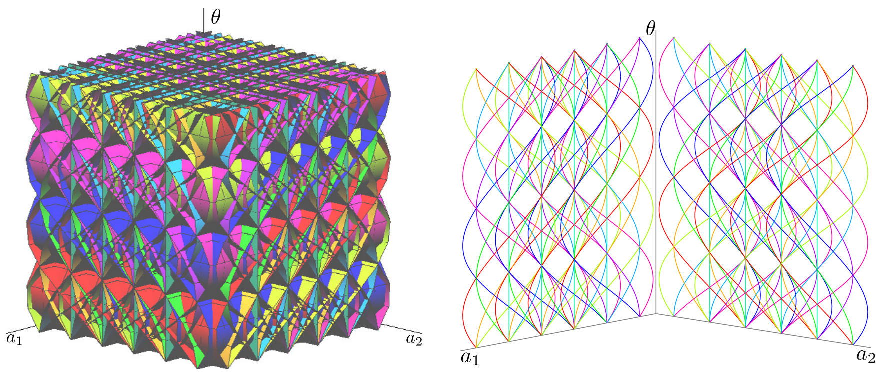
It is not possible to apply directly $\mathcal{T}$ to a digital image $I : \mathbb{Z}^2 \to \mathbb{V}$, since there is no guarantee that $\mathcal{T}(\mathbf{x}) \in \mathbb{Z}^2$ for $\mathbf{x} \in \mathbb{Z}^2$. In the discrete framework, the handling of *digital* rigid transformations requires to define a function $T_{a_1a_2\theta} : \mathbb{Z}^2 \to \mathbb{Z}^2$, which is a discrete analogue of $\mathcal{T}_{a_1a_2\theta}$. Following a digitization mapping D that associates to each real point $x \in \mathbb{R}$ its round value $[x] \in \mathbb{Z}$, a *digital rigid transformation* T associated to $\mathcal{T}$ can be conveniently performed by setting $T = D \circ \mathcal{T}$, as illustrated in the following diagram.

$$\begin{tikzcd} \mathbb{Z}^2 \arrow[r, "T=D\circ\mathcal{T}"] \arrow[d, "Id"] & \mathbb{Z}^2 \\ \mathbb{R}^2 \arrow[r, "\mathcal{T}"] & \mathbb{R}^2 \arrow[u, "D"'] \end{tikzcd} \tag{3}$$

The function $T : \mathbb{Z}^2 \to \mathbb{Z}^2$ is then explicitly defined for $\mathbf{p} = (p, q) \in \mathbb{Z}^2$ by

$$T(\mathbf{p}) = D \circ \mathcal{T}(\mathbf{p}) = \begin{pmatrix} [p\cos\theta - q\sin\theta + a_1] \\ [p\sin\theta + q\cos\theta + a_2] \end{pmatrix}. \tag{4}$$

In general, this function is not bijective. However, by setting $T^{-1} : \mathbb{Z}^2 \to \mathbb{Z}^2$ as $T^{-1} = D \circ \mathcal{T}^{-1}$, i.e., by considering the standard backward mapping, it is possible to define the digital transformed image $I \circ T^{-1} : \mathbb{Z}^2 \to \mathbb{V}$ with respect to T. In



(a) Tipping surfaces. (b) Tipping curves.

Fig. 1. (a) Tipping surfaces in the 3D parameter space (a_1, a_2, θ) and (b) their cross-sections, namely tipping curves, in the 2D planes (a_1, θ) and (a_2, θ).

the sequel of this article, we focus on such digital rigid transformations. From this point on – for the sake of readability and without loss of correctness – we will note T instead of T^{-1}, due to the bijectivity of $\mathcal{T}$ and $\mathcal{T}^{-1}$.

From a theoretical point of view, the above notions (images, rigid transformations) are defined on both $\mathbb{Z}^2$ and $\mathbb{R}^2$. Practically, our purpose is however to study rigid transformations on images of *finite* size. Under this hypothesis, only some digital rigid transformations are relevant, namely those that actually have an effect on such finite images. We focus on this finite case, and we assume that digital images are defined on subsets of $\mathbb{Z}^2$ of size $N \times N$. Without loss of generality, a digital image I is then written as $I : \mathbb{S} \to \mathbb{V}$ for $\mathbb{S} = [\![0, N-1]\!]^2 \subset \mathbb{Z}^2$.

3.2. *Digital rigid transformation space*

By contrast with rigid transformations defined in $\mathbb{R}^2$ (Eq. (2)), their digital analogues are not continuously defined with respect to the parameters a_1, a_2 and θ. This is pointed out by the fact that a same discrete point $T(\mathbf{p})$ can be obtained with slight variations of (a_1, a_2, θ), due to the rounding function D involved in T (Eq. (4)). In particular, it is shown in [15] that digital rigid transformations are generally not bijective and neither preserve the geometric and topological properties of digital objects.

The above-mentioned discontinuities result in a subdivision of the parameter space $\mathbb{R}^3$ of (a_1, a_2, θ) for digital rigid transformations. More precisely, this space is divided into 3D open cells, in which the function $(a_1, a_2, \theta) \mapsto T_{a_1 a_2 \theta} = D \circ \mathcal{T}_{a_1 a_2 \theta |\mathbb{S}}$ is piecewise constant. These (maximal) open cells are bounded by 2D closed sets where this function is non-continuous. From the very definition of D, these discontinuities correspond to the set of transformations that map discrete points onto semi-integer coordinate points, modelling "boundaries" of pixel cells (called hereafter, the half-grid $\mathcal{H}$).

The transformations $\mathcal{T}_{ab\theta}$ that lead to the discontinuities of T are expressed, for any $\mathbf{p} = (p, q) \in \mathbb{S}$ that is mapped onto a half-grid point which can be either a vertical $\mathbf{p}_\Phi = (k + \frac{1}{2}, \lambda) \in \mathcal{H}$, or a horizontal $\mathbf{p}_\Psi = (\lambda, l + \frac{1}{2}) \in \mathcal{H}$ (with $k, l \in \mathbb{Z}$ and $\lambda \in \mathbb{R}$) by

$$\left| \begin{array}{llll} \Phi_{pqk} & : \mathbb{R}^2 & \longrightarrow & \mathbb{R} \\ & (a_2, \theta) & \longmapsto & a_1 = \phi_{pqk}(\theta) = k + \dfrac{1}{2} + q \sin\theta - p \cos\theta \end{array} \right. \tag{5}$$

$$\left| \begin{array}{llll} \Psi_{pql} & : \mathbb{R}^2 & \longrightarrow & \mathbb{R} \\ & (a_1, \theta) & \longmapsto & a_2 = \psi_{pql}(\theta) = l + \dfrac{1}{2} - p \sin\theta - q \cos\theta. \end{array} \right. \tag{6}$$

The surfaces Φ_{pqk} (resp. Ψ_{pql}) are called *tipping surfaces*. Their cross-section ϕ_{pqk} (resp. ψ_{pql}) on the 2D plane (a_2, θ) (resp. (a_1, θ)) are called *tipping curves*. For an image of size $N \times N$, the tipping surfaces Φ_{pqk} and Ψ_{pql} satisfy $p, q \in [\![0, N-1]\!]$ and $k, l \in [\![0, N]\!]$. Fig. 1 illustrates these notions.

A set of tipping surfaces induces a subdivision of the (a_1, a_2, θ) space into equivalence classes, each of which consists of rigid transformations leading to a same digital result in $\mathbb{Z}^2$. Such classes are called *discrete rigid transformations* (DRTs).

3.3. *Discrete rigid transformation graph*

By mapping each 3D open cell onto a 0D vertex, and each separating 2D surface onto an 1D edge – following a Voronoi/Delaunay duality – we can model the subdivision of the 3D parameter space of (a_1, a_2, θ), as a graph, called *DRT graph*.

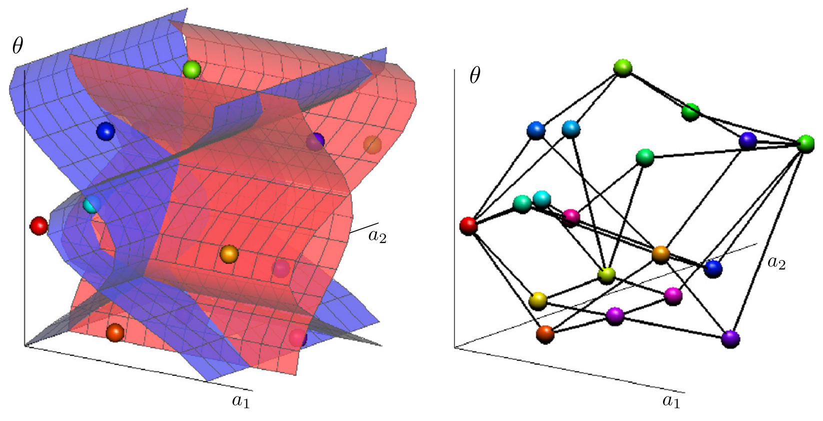


Fig. 2. (a) Subdivision of the (a_1, a_2, θ) parameter space into 3D cells by tipping surfaces and (b) the associated DRT graph.

Definition 1 (*DRT Graph [14]*)**.** Let $G = (V, E)$ be the graph defined such that:

(i) any vertex $v \in V$ models a 3D open cell associated to a DRT;
(ii) any edge $e = (v, w) \in E$ connects two vertices $v, w \in V$ sharing a 2D surface.

The graph $G = (V, E)$ is called a *DRT graph*.

In other words, the parameter space $\mathbb{R}^3$ of rigid transformations is subdivided, and the set V of $G = (V, E)$ is the direct transposition of this subdivision. In particular, G associated to a digital image I of finite size $N \times N$ is a finite data-structure that describes *all* the possible digital rigid transformations of I. An example of DRT graph is illustrated in Fig. 2.

It should be mentioned that the DRT graph G does not contain any geometric parameters (a_1, a_2, θ) of rigid transformations but only the *topological* information, that models the relationship between any neighbouring transformed images. More precisely, each vertex $v \in V$ of G is associated with a unique transformed image generated by any digital rigid transformation in v. It is showed in [14] that two connected transformed images in G differ by at most *one* over the N^2 pixels of I. Thus, each edge $e = (v, w) \in E$ of G is – implicitly – associated to a function indicating the *only* modification that differs between the transformed images corresponding to the DRTs v and w. This allows us to use the DRT graph to produce all the transformed images via successive *elementary* (i.e., single-pixel) modifications. This property opens a way of involving the DRT graph in digital image processing and analysis tasks. In the next section, we show how to use this combinatorial structure for handling image registration.

4. Discrete rigid transformation graph for image registration

4.1. DRT formulation of rigid image registration

On the one hand, the registration problem stated in Section 2 and formalized in Eq. (1) consists of finding the transformation that allows us to map at best a source image I_s onto a target image I_t with respect to a given distance. On the other hand, the DRT graph $G = (V, E)$ associated to an image of finite size, models all the existing DRTs for this image, i.e., all its transformed images; in particular this set of solutions is finite.

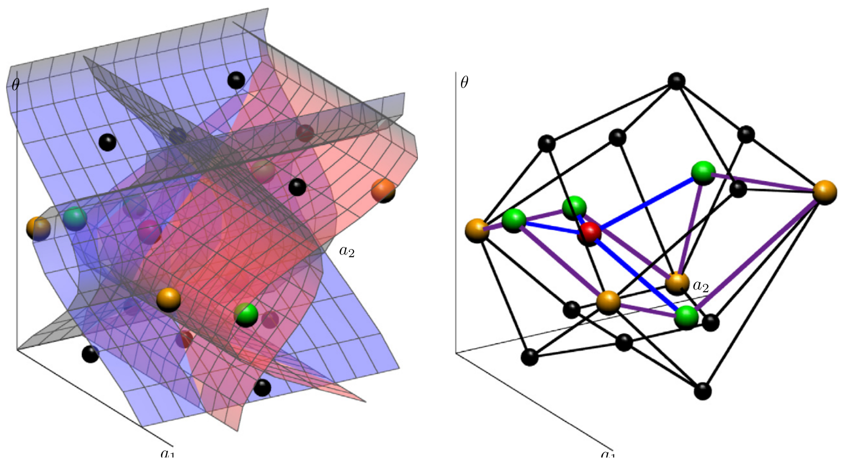
Therefore, one can use G to find the best rigid registration between I_s and I_t. More precisely, let $G = (V, E)$ be the DRT graph associated to the source image I_s. There exists $v \in V$ such that $d(I_s \circ T_v, I_t)$ is minimal, with T_v the transformation associated to the vertex v and $I_s \circ T_v$ the transformed image of I_s associated to T_v.[1] By that way, Eq. (1) can be reformulated by considering the rigid transformation space $\mathbb{T} = \{T_v \mid v \in V\}$ of all the DRTs as follows:

$$T_v^* = \arg\min_{v \in V} d(I_s \circ T_v, I_t). \tag{7}$$

Roughly speaking, a brute-force search within the whole graph G would lead to a global optimal solution for Eq. (7). In particular, from Section 3.3, the DRT graph transforms progressively (i.e., pixel by pixel) the image. Therefore, the exhaustive scanning of the DRT graph can be done by updating incrementally the distance between images.

However, it is proved in [14] that the DRT graph G, associated to an image of size $N \times N$, has a space complexity of $\mathcal{O}(N^9)$, which induces the same time complexity both for construction and exhaustive exploration of G. This polynomial complexity practically forbids the use of G as a whole in real applications for large images.

[1] Note that when applying a transformation T_v on I_s, the transformed digital image $I_s \circ T_v$ no longer fits in the $N \times N$ support of I_s. We deal with this issue by surrounding the image with a padded border. The border values are that of the background. We include this in the distance computation $d(I_s \circ T_v, I_t)$.



(a) Parameter space. (b) Neighbours in the DRT graph.

Fig. 3. Example of *k*-neighbours of a DRT. The given DRT is depicted in red, its 1-neighbours in green and its 2-neighbours in green and yellow. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

To overcome this issue, a local approach is proposed to reach a local optimum in G. Such an approach is based on a local search allowing to efficiently explore the graph without generating it entirely, and thus leads to a much lower algorithmic complexity. Thanks to the property of local modifications along the edges of the DRT graph, this approach can be integrated into an optimization process that guarantees to reach such optimum in the sense of minimizing the objective function with respect to a given distance between images.

4.2. *Local search in a DRT graph*

In this section, we propose a combinatorial optimization – gradient descent – applied on the investigated structure of DRT graph, in order to find locally optimal solutions for the rigid registration problem stated above. To this end, we first need to introduce a notion of *neighbour* of DRTs, and properties related to the locally optimal solutions induced by such neighbourhoods.

4.2.1. *Neighbourhood in a DRT graph*

In order to perform a gradient descent, for searching a local optimum, two questions are raised: (i) the choice of the initial vertex/transformation; and (ii) the choice of a "good" search area around a vertex, namely *neighbours* of this vertex. The latter is the most critical issue while the first is often guided by the application.

From the above discussions on the definition, structure and semantics of a DRT graph, we define the notion of *neighbours* by using the standard definition in graph theory. Two DRTs are adjacent – neighbours – in a DRT graph if their respective vertices are adjacent (i.e., share an edge) in this graph. Equivalently, the two DRTs share a surface in the dual parameter space of (a_1, a_2, θ). Based on the adjacency relation in the DRT graph, we can define the neighbours of depth $k \geq 1$ of a vertex, namely *k-neighbours*, as follows.

Definition 2 (*DRT Neighbourhood*)**.** Given a DRT graph $G = (V, E)$, the *k-neighbour* of a vertex $v \in V$ is

$$\mathcal{N}^k(v) = \mathcal{N}^{k-1}(v) \cup \bigcup_{u \in \mathcal{N}^{k-1}(v)} \{w \in V \mid \exists e = (u, w) \in E\}$$

where $\mathcal{N}^0(v) = \{v\}$.

This notion is illustrated in Fig. 3.

4.2.2. *Gradient descent within a DRT graph*

We now describe a generic local search algorithm, namely a gradient descent, within the DRT graph by exploring k-neighbourhoods of DRTs.

Given two images I_s and I_t, the DRT graph $G = (V, E)$ associated to I_s, and an initial DRT $v \in V$ (namely a seed), we determine a local optimum (or a set of local optima), i.e., a DRT minimizing the distance d between I_s and I_t. Since each DRT corresponds to a unique transformed image, and as there is at most one different pixel between two neighbouring DRTs (Section 3.3), it is possible to locally and incrementally compute the gradient Δd of d between the transformed image obtained from I_s and the target image I_t. By examining Δd for all neighbouring vertices of $\mathcal{N}(v)$ for a given vertex v, we can

Algorithm 1: Local search in the DRT graph.

```
Input: Source and target images $I_s$ and $I_t$; a vertex $v_0$ of the DRT graph $G$.
Output: A set $\widehat{V} \subseteq V$ of local optima with respect to Equation (7).
1 $\widehat{V}, V^* \leftarrow \{v_0\}$
2 while ($\widehat{V} = V^*$) do
3     $\widehat{V} \leftarrow V^*$
4     $V^* \leftarrow \arg\min\{d(I_s \circ T_v, I_t) \mid v \in \bigcup_{v' \in V^*} \mathcal{N}^k(v')\}$
```

then find the vertices presenting the minimal value $d(I_s \circ T_v, I_t)$. The obtained DRT(s) then become(s) new seed(s) for the next step.

By iterating this process, it is then possible to carry out a gradient descent from a seed vertex $v_0 \in V$, that finally leads to a local optimum with respect to Definition 2. The number of vertices of the DRT graph G being finite, this process, summarized in Algorithm 1, has a guaranteed termination.

Due to digitization effects, the distance d may be non-smooth, and can locally increase on a globally decreasing trajectory. In order to avoid the side effects on such disturbance on the gradient descent, that requires a globally decreasing distance along a reasonably long trajectory, it is preferable to compute Δd on a k-neighbourhood, with $k > 1$, to improve the robustness of the process.

Computing the 1-neighbourhood $\mathcal{N}(v)$ of a vertex v in low time cost is indeed tractable in a DRT graph, but preserving a good time complexity for larger neighbourhoods $\mathcal{N}^k(v)$ constitutes a real challenge. In particular, it is the main issue to tackle in our proposed framework.

In [13], we initially proposed an algorithm which could be adapted to find 1-neighbourhoods $\mathcal{N}(v)$. This algorithm presents a complexity of $\mathcal{O}(N^4 \log N)$, where $N \times N$ is the size of the given image. Based on the relations that link tipping surfaces and tipping curves (Section 3.2), a better algorithm was proposed in [16] to compute $\mathcal{N}(v)$ with a complexity of $\mathcal{O}(mN^2)$, where m is the maximum degree of the DRT graph (i.e., the number of 1-neighbours of a DRT). A recursive version of this algorithm could be derived for computing k-neighbourhoods, but it has an exponential complexity $\mathcal{O}(m^k N^2)$ with respect to k. In the next section, we present a more efficient algorithm, initially introduced in [7], that allows us to compute $\mathcal{N}^k(v)$, with a low complexity $\mathcal{O}(kN^2)$.

It should be mentioned that the proposed framework of registration based on DRT graph search is valid for any digital 2D image presenting isotropic pixels, since DRT graphs, by definition, do not depend on the values that are assigned to the pixels of images. In other words, the structure is invariant for any image defined on the same support; i.e., any image of same size $N \times N$ (see Section 3.3). Furthermore, the proposed framework is also independent from the choice of a distance metric, which is calculated at each iteration of the gradient process. For the sake of readability – but without loss of generality – we will consider binary and grey-level images with the distance based on the signed distance function [3,11] in the experiences of Sections 6.3 and 6.5.

5. Neighbourhood construction

The complexity of the above local search algorithm (Algorithm 1) directly relies on the complexity of the construction of the k-neighbours $\mathcal{N}^k(v)$, i.e., the computation of a local part of the DRT graph modelling the neighbours of a given vertex v. In this section, we describe an efficient algorithm for constructing such neighbours, based on the sweeping plane technique introduced in [14]. To this end, we first describe a modified sweeping algorithm for DRT sub-graph construction in an interval of θ within $[0, 2\pi[$.

5.1. Sweeping algorithm for discrete rigid transformation sub-graph construction

Similarly to the algorithm proposed in [14] for building incrementally the *whole* DRT graph, we describe hereafter how to build *partially* the graph within a given range of θ values, smaller than $[0, 2\pi[$. This problem is formalized as follows: given a set $\mathbf{S}$ of s_1 vertical and s_2 horizontal tipping surfaces, we aim to construct the DRT sub-graph G corresponding to the interval $[\theta_{start}, \theta_{end}[$, with $\theta_{start} < \theta_{end}$.

The sweeping algorithm consists of using a plane orthogonal to the θ-axis, denoted by γ, swept along this axis, from θ_{start} to θ_{end}. From the definition of tipping surfaces (Eqs. (5)–(6)), the tipping surfaces of $\mathbf{S}$ intersect and subdivide γ into $(s_1+1) \times (s_2+1)$ rectangular cells, as illustrated in Fig. 4. Each rectangle corresponds to a vertex, while each frontier between two rectangles corresponds to an edge of the constructed graph. When γ reaches an intersection of tipping surfaces of $\mathbf{S}$, a rectangle disappears while another appears: new vertices and edges are then generated and added into the DRT sub-graph G. Therefore, it is only required to maintain a set of sorted intersection points of tipping surfaces with respect to θ in $[\theta_{start}, \theta_{end}[$, and to make γ progress in this increasing order, to construct incrementally G. (The reader is referred to [14] for more details.)

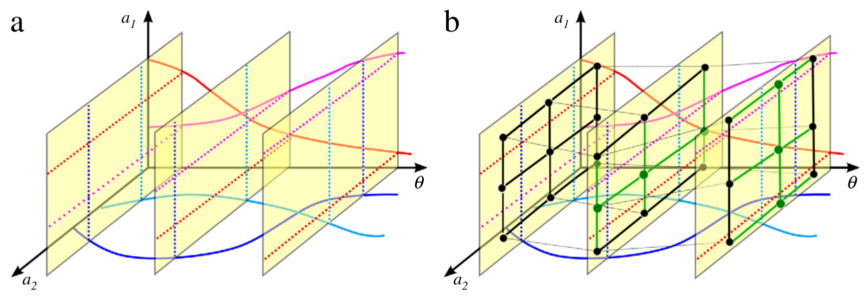


Fig. 4. DRT graph construction by the sweeping plane algorithm, with 2 vertical (blue, cyan) and 2 horizontal (red, magenta) tipping surfaces. (a) 3×3 rectangular cells generated by the tipping surfaces in each sweeping plane. (b) The associated part of the DRT graph in each plane (in green: new vertices and edges in the second and third planes). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Since we consider the interval $[\theta_{start}, \theta_{end}[$, we only need to calculate the intersections of tipping surfaces that occur in this interval. Consequently, the algorithm has a complexity that depends on the number of these intersections, namely $\mathcal{O}(|\mathbf{S}|^2)$ [14].

Nonetheless, from Fig. 4, one remarks that the tipping surfaces intersect the sweeping plane γ with a specific order along each $a_\star$-axis. At each intersection, this order evolves, but only between consecutive tipping surfaces involved in the intersection. Therefore, instead of calculating intersections between *all* tipping surfaces in $\mathbf{S}$, we can calculate only those of consecutive tipping surfaces in their ordered structure in γ along each $a_\star$-axis. In consequence, we consider at most $|\mathbf{S}| - 2$ intersections at each update of γ. The next intersection of tipping surfaces for updating the planar subdivision is the closest among these $|\mathbf{S}| - 2$. After such intersection, the order of the tipping surfaces generating the intersection in γ is swapped.

The overall procedure starts at $|\theta_{start}|$ with the ordered structure in γ with $\mathbf{S}$ along each $a_\star$-axis and, at each intersection, calculates $s_\star$ new vertices and their associated $(3s_\star + 2)$ edges, that are integrated into G, while the ordered structure in γ is updated. This is repeated until γ reaches θ_{end}. After each update, the modifications of such intersections can be performed in constant time. When $|\theta_{end} - \theta_{start}| \ll 2\pi$, this number of intersections can be considered as a small constant. Hereafter, we denote this specific procedure by $Sweep(\mathbf{S}, \theta_{start}, \theta_{end})$, and we write $G = Sweep(\mathbf{S}, \theta_{start}, \theta_{end})$ for the output DRT sub-graph.

In the sequel, we describe how to use this partial sweeping technique for computing the direct neighbours $\mathcal{N}^k(v)$ of v, of depth $k = 1$ and then $k \geq 1$. To guarantee the efficiency of this method, we will need as inputs: (1) the analytic representation of a DRT, and (2) a "representative" continuous rigid transformation within a given DRT.

5.2. Analytic representation of discrete rigid transformations

Let I_v be the transformed image associated to the vertex $v \in V$, obtained by applying the rigid transformation(s) of v on the initial image I. Each pixel (p_i, q_i) of I_v has a corresponding pixel (p'_i, q'_i) in I. Such correspondence is modelled by the following inequalities deriving from Eq. (4)

$$p'_i - \frac{1}{2} < p_i \cos\theta - q_i \sin\theta + a_1 < p'_i + \frac{1}{2} \tag{8}$$

$$q'_i - \frac{1}{2} < p_i \sin\theta + q_i \cos\theta + a_2 < q'_i + \frac{1}{2}. \tag{9}$$

Since I has a size $N \times N$, each of its N^2 pixels generates 4 such inequalities. Then, this set of $4N^2$ inequalities provides an analytic expression of the DRT v. In the parameter space (a_1, a_2, θ), the obtained $4N^2$ inequalities then define a 3D cell, called *feasible region* and denoted as R_v. In other words, such region gathers all the parameter triplets associated to the DRT corresponding to the vertex v.

When interpreting these inequalities in terms of tipping surfaces/curves (Eqs. (5)–(6)), it appears that for each pixel of I_v, Eqs. (8)–(9) define a region of the parameter space that is bounded by two offset vertical (resp. horizontal) tipping surfaces/curves $\Phi_{p_i q_i p'_i}$ and $\Phi_{p_i q_i p'_i - 1}$ (resp. $\Psi_{p_i q_i q'_i}$ and $\Psi_{p_i q_i q'_i - 1}$). For any $i \in [\![1, N^2]\!]$, $\Phi_{p_i q_i p'_i}$ (resp. $\Psi_{p_i q_i q'_i}$) is called an *upper tipping surface/curve*, while $\Phi_{p_i q_i p'_i - 1}$ (resp. $\Psi_{p_i q_i q'_i - 1}$) is called a *lower tipping surface/curve*. The sets composed by these surfaces/curves, for all $i \in [\![1, N^2]\!]$, are denoted $\mathbf{S}_1^+(I_v)$ and $\mathbf{S}_1^-(I_v)$ (resp. $\mathbf{S}_2^+(I_v)$ and $\mathbf{S}_2^-(I_v)$), and exemplified in Fig. 5.

5.3. Representative of a discrete rigid transformation

From the definition of DRTs (Section 3.2), the feasible region R_v of a DRT v contains the set of all the rigid transformations leading to a same digital transformation. We call a *representative* of R_v one of these rigid transformations, further noted $\mathcal{T}_v$.

From the sweeping algorithm described above, whenever the sweeping plane γ reaches an intersection in the parameter space, a 3D cell is ended while a new is generated. Equivalently in the dual space, new vertices and edges are generated in

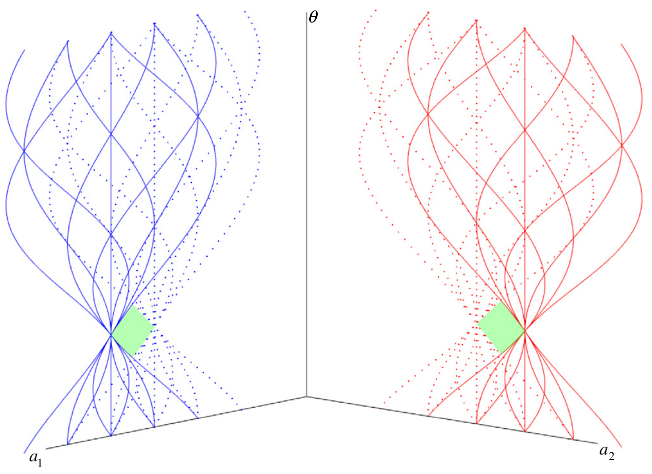


Fig. 5. Analytic representation of a DRT by its set of tipping surfaces. The figure illustrates the projection of these surfaces onto the planes $(a_\star, \theta)$, where the upper (resp. lower) tipping surfaces/curves are represented by continuous (resp. dashed) lines, while the feasible region of the DRT is depicted in green. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the DRT sub-graph. By tracking these evolutions during the sweep, one can determine the θ_{min} and θ_{max} of each DRT. More precisely, the θ value where the region is created (resp. ended) corresponds to θ_{min} (resp. θ_{max}). In particular, for a given DRT v analytically expressed by the sets of $|\mathbf{S}_1|$ vertical and $|\mathbf{S}_2|$ horizontal tipping surfaces, each of which is composed of the upper and lower tipping surfaces, i.e., $\mathbf{S}_1 = \mathbf{S}_1^+(I_v) \cup \mathbf{S}_1^-(I_v)$ and $\mathbf{S}_2 = \mathbf{S}_2^+(I_v) \cup \mathbf{S}_2^-(I_v)$ (Section 5.2), the θ_{min} (resp. θ_{max}) of the feasible region R_v of v is determined at the θ value for which the sweeping plane starts (resp. finishes) intersecting the tipping curves of each plane (a_*, θ) in the sequences of upper and lower curves (Fig. 6).

In [12], it was observed that the feasible regions of DRTs present directional convexity properties with respect to the $a_\star$-axes in the parameter space (a_1, a_2, θ). Thanks to these properties, we can calculate a representative $\mathcal{T}_v = (a_{1v}, a_{2v}, \theta_v)$ associated to R_v once knowing its θ_{min} and θ_{max}[2] as follows.

$$\begin{cases} \theta_v &= \Big(\theta_{min}(v) + \theta_{max}(v)\Big)/2 \\ a_{1v} &= \Big(\min\{\phi^+(\theta_v) \mid \phi^+ \in \mathbf{S}_1^+(I_v)\} + \max\{\phi^-(\theta_v) \mid \phi^- \in \mathbf{S}_1^-(I_v)\}\Big)/2 \\ a_{2v} &= \Big(\min\{\psi^+(\theta_v) \mid \psi^+ \in \mathbf{S}_2^+(I_v)\} + \max\{\psi^-(\theta_v) \mid \psi^- \in \mathbf{S}_2^-(I_v)\}\Big)/2. \end{cases} \tag{10}$$

As stated above, $\mathcal{T}_v$ can be any transformation within R_v. Thus, θ_v can be any value in the interval $]\theta_{min}(v), \theta_{max}(v)[$ of R_v. In particular, we can set θ_v at the middle of the interval as in Eq. (10).

In [14], it is shown that the value of θ can be expressed and compared as a periodic continued fraction, modelled by a sequence of integers. Using this representation, one can find a θ_v between $\theta_{min}(v)$ and $\theta_{max}(v)$ in Eq. (10) in an exact way.

It may be noticed that in the sweeping algorithm for DRT sub-graph construction (Section 5.1), we only need the interval of θ values as parameter, namely θ_{start} and θ_{end} (with $\theta_{start} = \theta_v$ and $\theta_{end} =$ either θ_{min} or θ_{max}). In such cases, the explicit calculation of a_{1v} and a_{2v} is not required.

5.4. 1-Neighbours of a discrete rigid transformation

From Definition 2 and Fig. 3, computing the 1-neighbours of a DRT v is equivalent to finding the vertices adjacent to v. Let us assume that we know a representative $\mathcal{T}_v$ associated to v (calculated as shown in Section 5.3, or given as input). We can now formulate the problem as follows: given a rigid transformation $\mathcal{T}_v$ associated to the feasible region R_v of a DRT v, represented by $4N^2$ inequalities of $\mathbf{S}_1 = \mathbf{S}_1^+(I_v) \cup \mathbf{S}_1^-(I_v)$ and $\mathbf{S}_2 = \mathbf{S}_2^+(I_v) \cup \mathbf{S}_2^-(I_v)$ (Section 5.2), how can we determine the neighbourhood $\mathcal{N}(v)$ of v.

The computation of $\mathcal{N}(v)$ is equivalent to constructing the DRT sub-graph of only neighbouring vertices and edges of v. From Section 5.1, we already have a sweeping algorithm for building the DRT sub-graph for $\theta \in [\theta_{start}, \theta_{end}[$. We now use it for finding $\mathcal{N}(v)$, by sweeping the plane γ twice. Indeed, the value[3] θ_v is considered as the initial position, and the plane is swept to the left-hand and right-hand sides of θ_v until the θ_{min} and θ_{max} values of R_v are reached, respectively (Fig. 7).

[2] The algorithm for finding the feasible region R_v, and in particular $\theta_{min}(v)$ and $\theta_{max}(v)$, given $4N^2$ tipping surfaces $\mathbf{S}_1$ and $\mathbf{S}_2$ has a complexity $\mathcal{O}(N^6)$ [12]. However, in practice, the initial solution for the gradient descent algorithm is given as a rigid transformation rather than a set of inequalities $\mathbf{S}_1$ and $\mathbf{S}_2$. Then, the θ_{min} and θ_{max} of R_v, as well as its neighbours, can be calculated during the sweeping algorithm. In consequence, knowing θ_{min} and θ_{max}, a representative T_v of R_v is calculated in constant time using Eq. (10).

[3] Note that θ_v can be easily computed using Eq. (10) with θ_{min} and θ_{max} calculated during the sweeping algorithm.

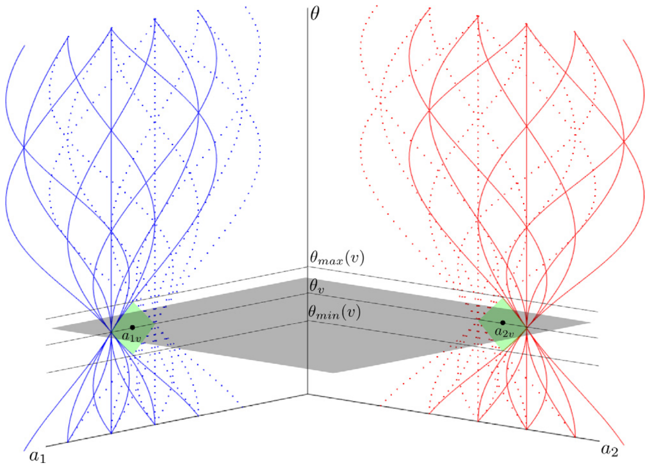


Fig. 6. A representative $\mathcal{T}_v = (a_{1v}, a_{2v}, \theta_v)$ of the feasible region(s) R_v (in green) associated to a DRT v, is any rigid transformation within R_v. Note that any sweeping plane (in grey) orthogonal to the θ-axis, and passing through R_v intersects the tipping curves of each plane (a_1, θ) and (a_2, θ) in a sequence in which the upper and lower curves are separated. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

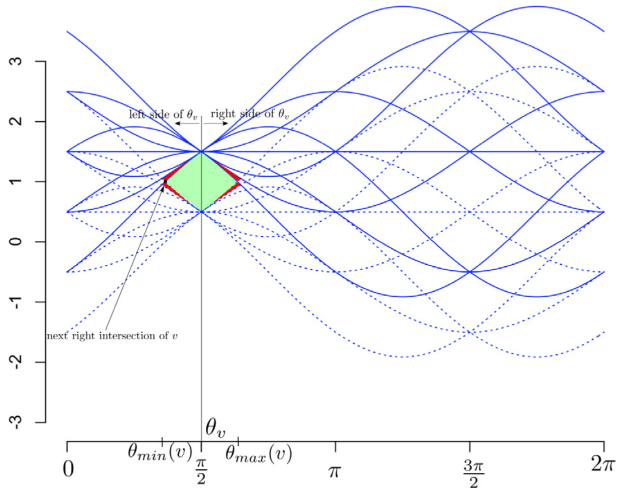


Fig. 7. Illustration of 1-neighbours construction of a given DRT v on the plane (a_1, θ). The region of v is in green while its 1-neighbours are in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

At each step of the algorithm – when the sweeping plane reaches an intersection of tipping surfaces – several vertices and edges are generated, but only those that are adjacent to v are considered for the DRT sub-graph of $\mathcal{N}(v)$.

The question raised by using the algorithm of Section 5.1 is how to limit the construction of the DRT sub-graph to these neighbours only. In fact, thanks to the directional convexity property along the $a_\star$-axis of a feasible region of a DRT, the sequence of tipping surfaces intersecting the sweeping plane is separated into two parts: the upper and the lower tipping surfaces, respectively (Section 5.2). Thus, by observing this sequence of tipping surfaces updated during the sweep, one can track them until the algorithm stops. The procedure for the left-hand side is given in Algorithm 2 (the right-hand side is similar).

5.5. *k*-neighbours construction algorithm

In this section, we extend the previous algorithm for finding the k-neighbours of a given DRT v, i.e., $\mathcal{N}^k(v)$ for $k \geq 1$ (Fig. 8). For this, we need a function that associates the depth of a vertex u with respect to v, denoted by $t_v(u)$ (in particular, $t_v(v) = 0$).

For each new vertex u generated during the sweep, the depth of u is defined from its two adjacent vertices w_1 and w_2 in the $a_\star$-direction of the tipping surface intersection, such that $t_v(u) = 1 + \min\{t_v(w_1), t_v(w_2)\}$. (Note that an iterative backtracking process is needed to update the depths of $w_\star$ and its successive neighbours, whenever $t_v(w_\star) > t_v(u) + 1$.)

Algorithm 2: 1-Neighbours computation (left-hand side along the θ-axis)

```
Input: A DRT $v$ / its associated image $I_v$.
Output: The DRT sub-graph $G_v = (V_v, E_v)$ containing the 1-neighbours of $v$.
1  Initialize $V_v = \emptyset, E_v = \emptyset$
2  Initialize $\theta_{prev} = \theta_v$
3  for $\star = 1, 2$ do
4      Determine the tipping surfaces $\mathbf{S}^+_\star(I_v), \mathbf{S}^-_\star(I_v)$ associated to $v$ (Section 5.2).
5      Let $f^+_\star$ (resp. $f^-_\star$) be the lowest (resp. uppermost) tipping surface in $\mathbf{S}^+_\star(I_v)$ (resp. $\mathbf{S}^-_\star(I_v)$), that intersects the initial plane at $\theta_v$.
6      Let $\mathbf{S}_\star$ be an ordered set containing $f^+_\star$ (resp. $f^-_\star$) and the next tipping surface that is lower (resp. upper) than $f^+_\star$ (resp. $f^-_\star$).
7  repeat
8      for $\star = 1, 2$ do
9          Find the next left intersection $\theta^+_\star$ of $f^+_\star$ (resp. $\theta^-_\star$ of $f^-_\star$) with the other surfaces in $\mathbf{S}_\star$ for $\theta < \theta_{prev}$.
10     $\theta_{next} = \min\{\theta^+_1, \theta^-_1, \theta^+_2, \theta^-_2\}$
11     $\Delta G_v = Sweep(\mathbf{S}_1 \cup \mathbf{S}_2, \theta_{prev}, \theta_{next})$
12     if $\exists u \in \Delta V_v$ such that $e = (u, v) \in \Delta E_v$ then
13         $G_v = G_v \cup \Delta G_v$
14         $\theta_{prev} = \theta_{next}$
15         if the next intersecting surface with $f^+_\star$ (or $f^-_\star$) is in $\mathbf{S}_\star$ then
16             Exchange their order in $\mathbf{S}_\star$.
17         else
18             Replace $f^+_\star$ (or $f^-_\star$) in $\mathbf{S}_\star$ with the new intersecting surface.
19 until no more separation of tipping surfaces in $\mathbf{S}_\star$;
```

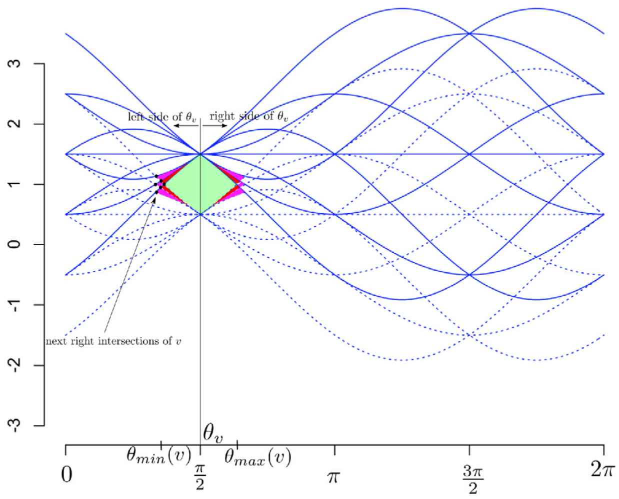


Fig. 8. Illustration of k-neighbours ($k = 2$) construction of a given DRT v on the plane (a_1, θ). The region of v is in green, while its 1- and 2-neighbours are in red and purple, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

This depth function is used to terminate the procedure of construction of the DRT sub-graph $\mathcal{N}^k(v)$. Indeed, if $t_v(u) > k$ for all vertices in the current sweeping plane, then the process ends. Furthermore, in order to obtain all the vertices u such that $t_v(u) \leq k$, we need to initialize the set of tipping surfaces $\mathbf{S}^+_\star(I_v)$ (resp. $\mathbf{S}^-_\star(I_v)$) with the $k+1$ lowest (resp. uppermost) tipping surfaces that intersect the initial sweeping plane at θ_v. Note that in the DRT sub-graph of $\mathcal{N}^k(v)$, we keep only vertices whose neighbourhood depth is lower than k. The modified algorithm for $\mathcal{N}^k(v)$ is given in Algorithm 3.

This algorithm is similar to Algorithm 2, except for three points (depicted in blue): (1) the initialization of the set $\mathbf{S}_\star$ with $2(k+1)$ tipping surfaces instead of 4 (Lines 3–6); (2) the determination of vertices whose neighbourhood depth is $\leq k$, to add them into the sub-graph of $\mathcal{N}(v)$ (Line 12); and (3) the second loop termination when all considered vertices have a depth $>k$ (Line 19). Roughly speaking, Algorithm 2 is a particular case of Algorithm 3 for $k = 1$.

Algorithm 3: k-Neighbours computation (in the left-hand side along the θ-axis)

```
Input: A DRT $v$ / its associated image $I_v$; a positive integer $k$.
Output: The DRT sub-graph $G_v = (V_v, E_v)$ containing the $k$-neighbours of $v$.
1  Initialize $V = \emptyset, E = \emptyset$
2  Initialize $\theta_{prev} = \theta_v$
3  for $\star = 1, 2$ do
4      Determine the tipping surfaces $\mathbf{S}^+_\star(I_v), \mathbf{S}^-_\star(I_v)$ associated to $v$ (Section 5.2).
5      In $\mathbf{S}^+_\star(I_v)$ (resp. $\mathbf{S}^-_\star(I_v)$), find the $(k+1)$-th lowest (resp. uppermost) tipping surface $f^+_\star$ (resp. $f^-_\star$), that intersects
       the initial plane at $\theta_v$.
6      Find and sort the $k+1$ tipping surfaces that are upper (resp. lower) or equal to $f^+_\star$ (resp. $f^-_\star$), and put them in an
       ordered set $\mathbf{S}_\star$.
7  repeat
8      for $\star = 1, 2$ do
9          Find the next left intersection $\theta^+_\star$ of $f^+_\star$ (resp. $\theta^-_\star$ of $f^-_\star$) with the other surfaces in $\mathbf{S}_\star$ for $\theta < \theta_{prev}$.
10     $\theta_{next} = \min\{\theta^+_1, \theta^-_1, \theta^+_2, \theta^-_2\}$
11     $\Delta G_v = Sweep(\mathbf{S}_1 \cup \mathbf{S}_2, \theta_{prev}, \theta_{next})$
12     if $\exists u \in \Delta V, t_v(u) \leq k$ then
13         $G_v = G_v \cup \Delta G_v$
14         $\theta_{prev} = \theta_{next}$
15         if the next intersecting surface with $f^+_\star$ (or $f^-_\star$) is in $\mathbf{S}_\star$ then
16             Exchange their order in $\mathbf{S}_\star$.
17         else
18             Replace $f^+_\star$ (or $f^-_\star$) in $\mathbf{S}_\star$ with the new intersecting surface.
19 until $\forall u \in \Delta V_v, t_v(u) > k$;
```

5.6. *Complexity analysis*

In this section, without lost of generality, we analyse the complexity of the k-neighbourhood algorithm (Algorithm 3 in Section 5.5), and Algorithm 2 in Section 5.4 is induced for $k = 1$.

In Algorithm 3, the first loop (Lines 3–6) initializes the set $\mathbf{S}_\star$ used to compute the k-neighbours $\mathcal{N}^k(v)$ of a given DRT v. More precisely, Line 4 finds the sets $\mathbf{S}^+_\star(I_v)$ and $\mathbf{S}^-_\star(I_v)$ containing the upper and lower tipping surfaces of v, and has a complexity of $\mathcal{O}(N^2)$. Lines 5–6 compute the $2k$ nearest tipping surfaces after the lowest and uppermost tipping surfaces $f^+_\star$ and $f^-_\star$ and store all in $\mathbf{S}_\star$. (Note that $\mathbf{S}_\star$ contains $2(k+1)$ tipping surfaces.) In particular, $\mathcal{O}(N^2)$ for Line 5 in average, if we use Hoare's find algorithm [5]; and $\mathcal{O}(N^2)$ and $\mathcal{O}(k \log k)$ for finding and sorting the tipping surfaces in Line 6, respectively.

The second loop (Lines 7–18) calculates $\mathcal{N}^k(v)$ from $\mathbf{S}_\star$. Firstly, Lines 8–10 compute the interval of interest of θ, by calculating the next intersection (namely, θ_{next}) from the current θ (namely, θ_{prev}). In this step, the most costly part is Line 9, which finds all intersections of $\mathbf{S}_\star$. Since $\mathbf{S}_\star$ contains $2(k+1)$ tipping surfaces, this costs $\mathcal{O}(|\mathbf{S}_\star|^2) = \mathcal{O}(k^2)$. Secondly, Line 11 builds a DRT sub-graph by using the Sweep algorithm for the interval $[\theta_{prev}, \theta_{next}[$; this requires $\mathcal{O}(N^2)$, which is the most costly part in the loop. The number of iterations of this second loop is estimated as $m(2k+1)$, since the k-neighbours of the vertex v contain at most $2k+1$ adjacent vertices in the θ-direction. Therefore, the complexity of Algorithm 3 is $\mathcal{O}(mkN^2)$. One remarks that this complexity depends on the size of the neighbourhood of v, namely the value of m. We now prove that m has a low constant value, that leads to a final complexity of $\mathcal{O}(kN^2)$.

From [14], the DRT graph space complexity for an image of size $N \times N$ is $\mathcal{O}(N^9)$, both for vertices and edges, since the number of vertices and that of edges grow at the same rate. However, we can infer that m is actually bounded, independently of N. Indeed, by analogy, let us imagine that we divide a 2D plane with straight lines defined randomly. Three lines will almost never intersect at a same point, and for a number of lines sufficiently large, the cells of the induced subdivision will be mostly triangles. Following this analogy, we may infer that the degree of the vertices of the 2D DRT graphs in the planes (a_1, θ) and (a_2, θ) is close to 3, in average. However, this analogy has some limits: the considered tipping curves are not straight lines, while their very regular structure implies that many curves often intersect at a same point. Nevertheless, we can assimilate a 2D DRT graph (which is the projection of a 3D DRT graph onto the $(a_\star, \theta)$ plane) to a planar graph whenever N is sufficiently large. Under such assumption, the Euler formula is valid, i.e., we have $v - e + f = 2$, where v, e and f are the number of (0D) vertices, (1D) edges and induced (2D) cells, respectively. From the construction of the DRT graph, each cell f is bounded by at least 4 edges e [14]. In other words, we have $4f \leq 2e$. It then comes that $2e/v \leq 4 - 8/v$. As $v \gg 8$ in DRT graphs, we have $2e/v < 4$, where $2e/v$ is indeed the average degree of the 2D DRT graph. It follows that the average degree m of the 3D DRT graph (obtained by Cartesian product of two 2D DRT graphs) is lower than $2 \times 4 = 8$. Fig. 9 illustrates the distribution of these degrees for a 2D DRT graph.

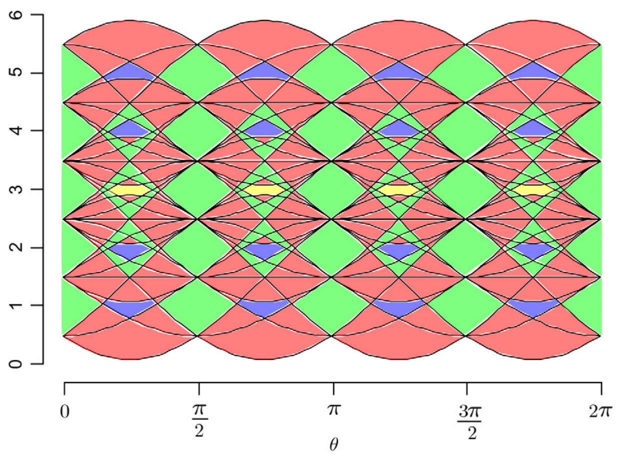


Fig. 9. Degree distribution in a 2D DRT graph, viewed in the dual subdivision of the parameter space. Each colour represents a given degree, that corresponds here to the number of curves bounding each cell (3: red, 4: green, 5: blue; 6: yellow). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

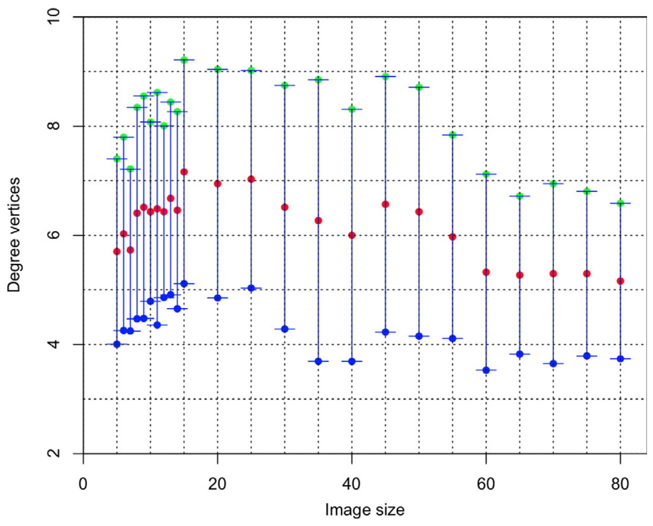


Fig. 10. Degree of vertices m within DRT graphs. The value m is calculated over 960 experiments for different image sizes. Red (resp. blue and green) points denote the average value of m (resp. the standard deviation $\pm\sigma$) over the experiments. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

6. Experimental results

In this section, we provide experiments devoted to validate the theoretical complexity analysis described in Section 5.6, as well as to demonstrate the behaviour of the local search approach using the algorithms developed in Section 5, in the context of image rigid registration.

6.1. Average degree of DRT graphs

First, we observe the evolution of the average degree of the vertices within a DRT graph, i.e., the number m of 1-neighbours of a DRT. From Section 5.6, we proved that the average value of m is lower than 8. This is confirmed by the experimental analysis illustrated in Fig. 10. The value m is calculated using Algorithm 2. The experiments are carried out for images of size varying from 5×5 to 80×80, in each of which several DRTs are chosen randomly to compute the value of m.

Observing the experiments on 2D DRT graphs – the projections of a 3D DRT graph onto the $(a_\star, \theta)$ planes – the degree of the vertices remains close to the average value, i.e., 4 (this is 8 in case of 3D DRT graphs, which is twice of the 2D one). Indeed, Fig. 11 shows that the 2D DRT vertex degrees converge, with respect to the image sizes, to a stable distribution that contains mainly degrees of value 3 and 4 (with a maximal value experimentally identified at 8). This confirms the stable behaviour of m with respect to the image size, in practice.

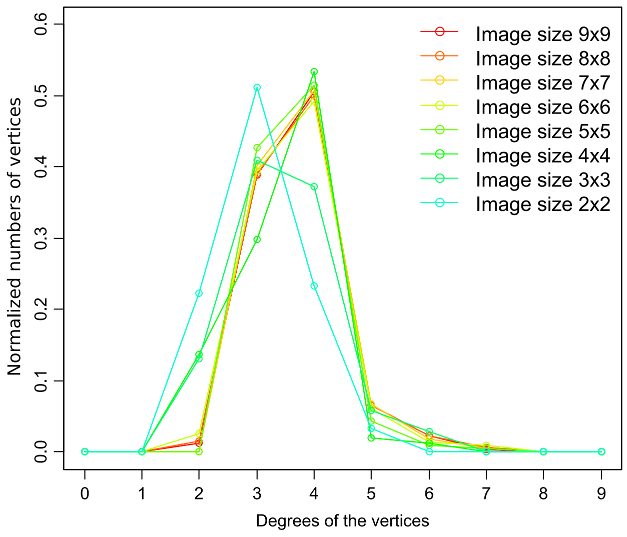


Fig. 11. Normalized vertex degree distribution in 2D DRT graphs.

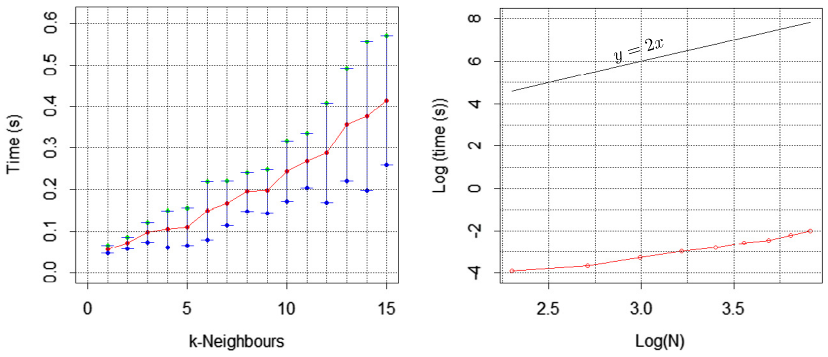


Fig. 12. Run-time complexity of k-neighbours computation is experimentally in linear time with respect to k (left) and image size N^2 (right). Red (resp. blue and green) points denote the average value (resp. the standard deviation $\pm\sigma$) of time over the experiments. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

6.2. Run-time complexity

We now observe the run-time computation of the k-neighbours algorithm. As described in Sections 5.5 and 5.6, Algorithm 3 has a theoretical complexity of $\mathcal{O}(kN^2)$, which is in linear time with respect to k for a given image size $N \times N$, and in linear time with respect to this size $N \times N$, for a given k. We experimentally assess the actual cost of this algorithm. To this end, we firstly consider an image of size 53×53 and vary the value of k from 1 to 15. Secondly, we fix k-neighbour size at 3 and vary N from 10 to 50. The results are shown in Fig. 12; for each varying value of k and N, the experiments are repeatedly performed, and the time values are then averaged, to confirm such linear result.

6.3. Experiments: image registration

We now describe some experiments and results of image registration, using an iterated local search developed in a *fully discrete* framework. More precisely, we combine a gradient descent method (Section 4.2) for the optimization part of the scheme, with the k-neighbours computation (Section 5.5) for the local search part. For run-time efficiency of Algorithm 1, instead of maintaining a set of all local optima at each gradient descent iteration, we will select randomly one in the set to iterate. It should be noticed that the proposed approach can be carried out on various types of images (binary, grey-level, label, colour, etc.) since DRT graphs are defined independently from the value space of the images (Section 3.3). It is however necessary to use an appropriate distance function d in Eq. (7). In order to illustrate and analyse the issues related to the digitization on the discrete space of the transformed images – and for the sake of readability – we focus our experiments on the case of binary and grey-level images with the distance based on the signed distance function [3,11]. The latter choice is justified as follows.

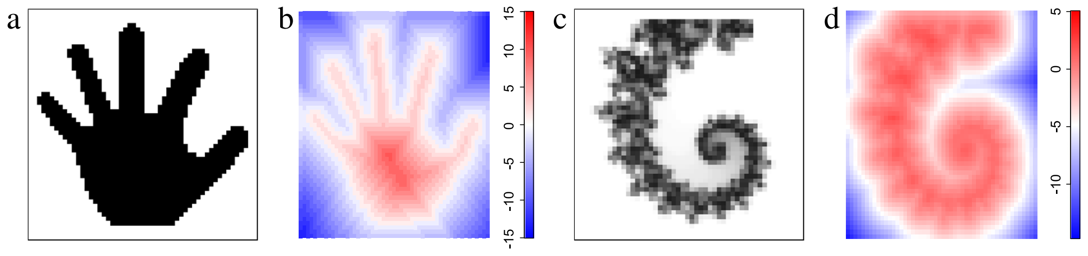


Fig. 13. Examples of signed distance map of images. (a) Binary image. (b) Signed distance transform of (a). (c) Grey-level image. (d) Average of the signed distance function of all binary level set images of (c).

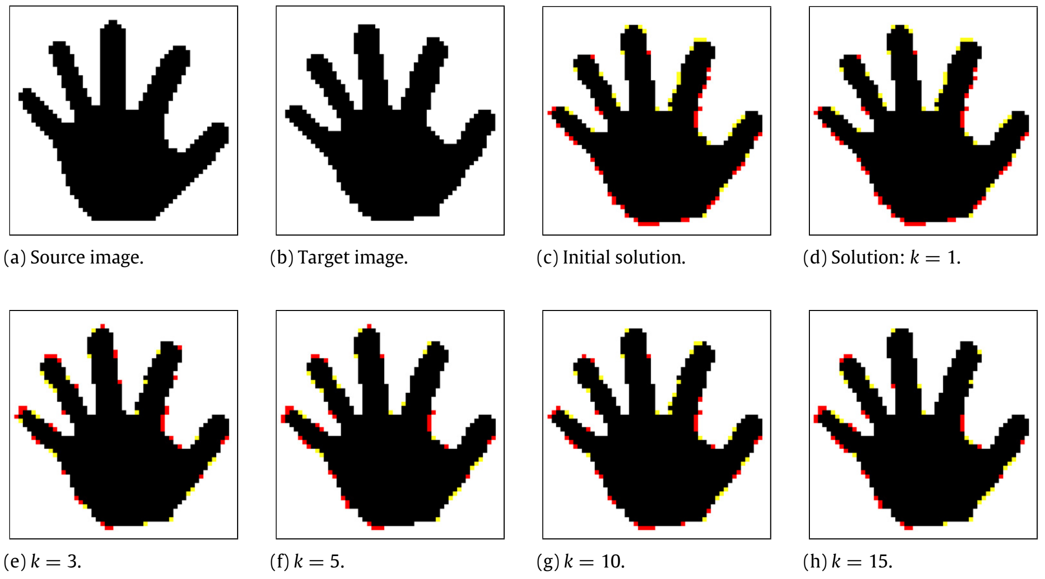


Fig. 14. Input images and results of the iterated local search for image registration. (a) Source image, (b) Target image generated from (a) with $(a_1, a_2, \theta) = (0.1, 0.3, 0.114)$ and (c) the initial transformed image of (a) with $(a_1, a_2, \theta) = (0.365, -0.045, 0.1423)$. (d–h) Local optima obtained from (c) by using k-neighbours for $k = 1, 3, 5, 10, 15$; the representative transformation parameter triples of the resulting DRT are $(a_1, a_2, \theta) = (0.3641, -0.0483, 0.1423)$, $(0.0118, 0.0176, 0.1458)$, $(-0.0109, 0.0125, 0.0147)$, $(0, -0.0529, 0.1446)$, $(-0.0385, -0.0619, 0.1442)$ respectively. Note that in (c–h), pixels are coloured if they are different from those in (b); yellow (resp. red) pixels are considered as white (resp. black) in (c–h) and black (resp. white) in (b). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

We recall that the signed distance describes the distance of a given point to the boundary of the object. More precisely, the distance is positive (resp., negative) if the point is inside (resp., outside) the boundary, and its absolute value decreases towards this boundary, where the signed distance function is zero (see Fig. 13). Formally, let $\Omega \subset \mathbb{Z}^2$ be a bounded domain, for any point $\mathbf{x} \in \mathbb{Z}^2$, the *signed distance* function $sdf : \mathbb{Z}^2 \to \mathbb{R}$ is defined as:

$$sdf_{\Omega}(\mathbf{x}) = d_{\Omega}(\mathbf{x}) - d_{\overline{\Omega}}(\mathbf{x})$$

where $d_{\Omega}(\mathbf{x})$ denotes the usual Euclidean distance transform to the set Ω and $\overline{\Omega}$ is the complement of Ω. Then, the *distance* based on signed distance function between two bounded sets Ω and Ω', which are in the target image I_t and the transformed source image $I_s \circ T$ respectively, is defined as:

$$d(\Omega, \Omega') = \int_{\Omega'} sdf_{\Omega}(\mathbf{x}) - \int_{\Omega} sdf_{\Omega}(\mathbf{x}).$$

In the context of registration, the second term is constant, while the first term is defined as a point-wise distance between the transformed image of the source and the distance map of the target—generated once for the whole process.[4] From Section 3.3, two adjacent/consecutive transformed images in the DRT graph differ by only one pixel. By that way, at each iteration of the gradient descent method, we need to update the distance only at this changed pixel with respect to the distance map of the target image. In other words, the distance can be computed in a constant time, except for the very first

[4] Note that the distance based on signed distance function for grey-level images can be calculated similarly. More precisely, a grey-level image can be modelled by a finite set of its binary level set images. Then, the distance is calculated by summing all the signed distance transforms of the binary level set images.

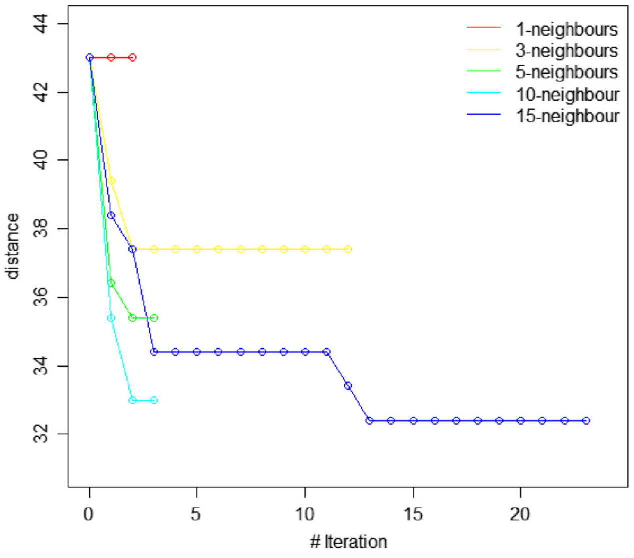


Fig. 15. Distance evolution during iterations of local search for the inputs in Fig. 14(a) and (b), from the initial transformation in Fig. 14(c), with respect to different depths k.

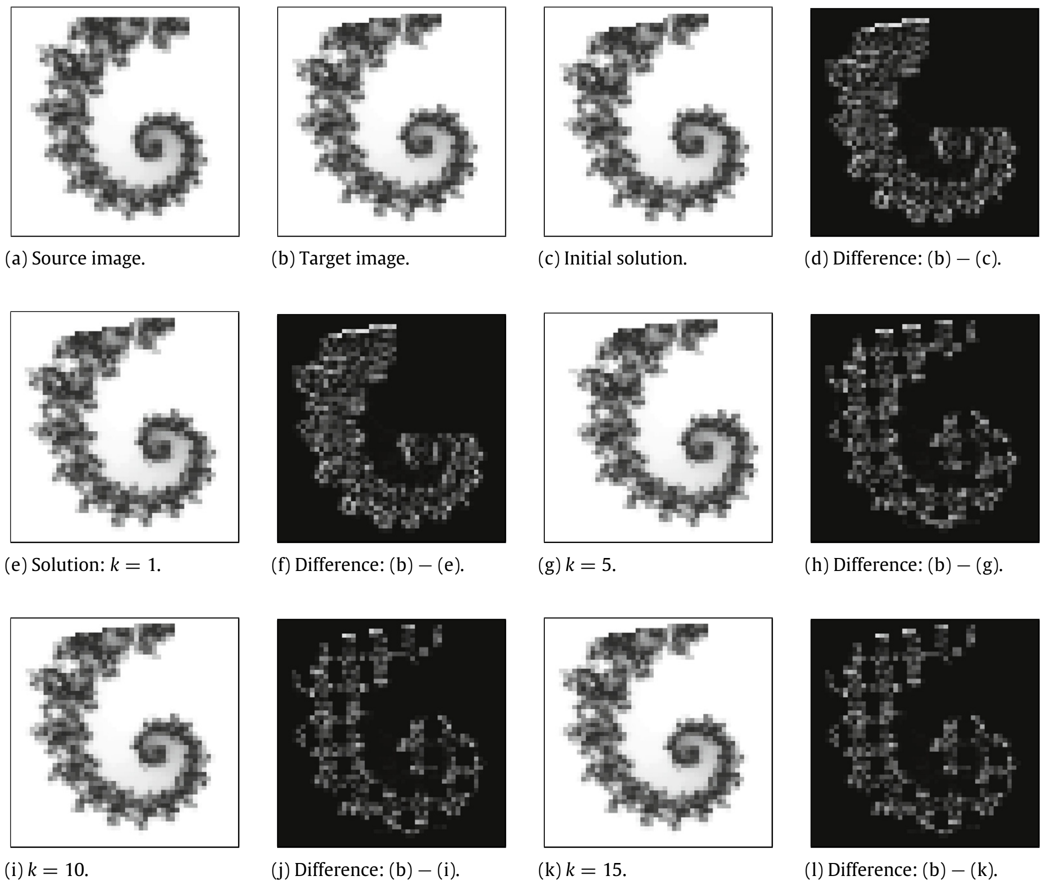

(a) Source image.
(b) Target image.
(c) Initial solution.
(d) Difference: (b) − (c).
(e) Solution: $k = 1$.
(f) Difference: (b) − (e).
(g) $k = 5$.
(h) Difference: (b) − (g).
(i) $k = 10$.
(j) Difference: (b) − (i).
(k) $k = 15$.
(l) Difference: (b) − (k).

Fig. 16. Input images and results of the iterated local search for image registration. (a) Source image, (b) Target image generated from (a) with $(a_1, a_2, \theta) = (0.1, 0.3, 0.114)$ and (c) the initial transformed image of (a) with $(a_1, a_2, \theta) = (0.49, 0.52, 0.153)$ and (d) difference between the target image (b) and the initial transformed image (c). (e, g, i, k) Local optima obtained from (c) by using k-neighbours for $k = 1, 5, 10, 15$; the representative transformation parameter triples of the resulting DRT are $(a_1, a_2, \theta) = (0.491, 0.5204, 0.153)$, $(0.4502, 0.5143, 0.1531)$, $(0.4529, 0.5118, 0.1535)$, $(0.4673, 0.4977, 0.153)$ respectively. (f, h, j, l) Difference between (e, g, i, k) and (b), respectively.

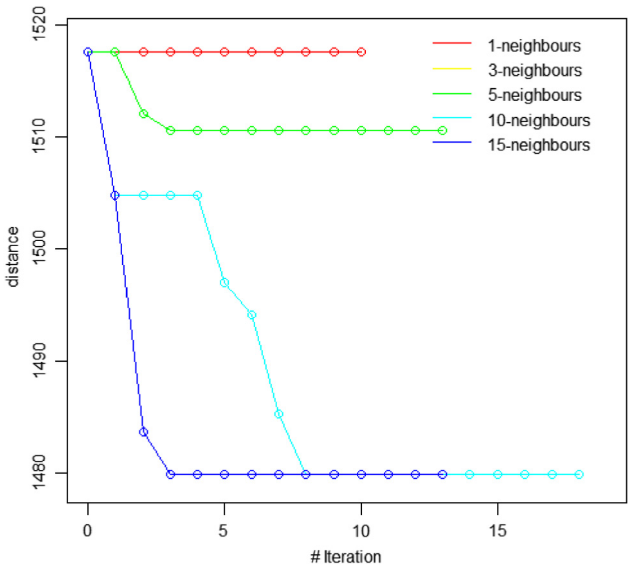


Fig. 17. Distance evolution during iterations of local search for the inputs in Fig. 16(a) and (b), from the initial transformation in Fig. 16(c), with respect to different depths $k = 1, 3, 5, 10$ and 15.

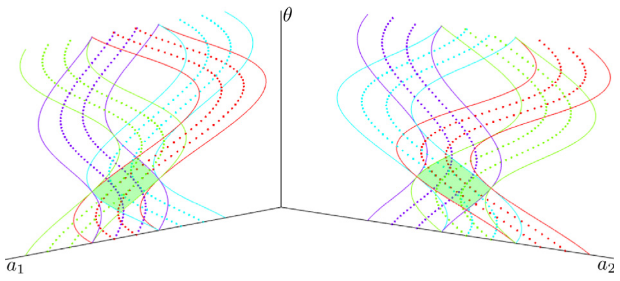


Fig. 18. Multi-scale approach for discrete rigid transformations. The image is considered in blocks of pixels of size $[-\frac{3}{2}, \frac{3}{2}]^2$, and a resolution of $\frac{1}{3}$. In the parameter space, at resolution $\frac{1}{3}$ we obtain the tipping surfaces/curves depicted by continuous lines. When increasing the resolution, "offset" surfaces/curves (dashed lines) appear.

iteration when the distance is calculated for the whole image. Furthermore, it is shown in [3,11] that this distance gives fewer flat zones for the gradient term of d.

Concerning the initial solution, namely a *seed*, for the local search algorithm, we use the first and second order central moments for binary images [4], and a SIFT feature based method [9,1] for grey-level images.

The experiments are carried out with different neighbourhood sizes $k = 1, 3, 5, 10, 15$ on binary and grey-level images of size 53×53 from given initial transformations (Figs. 14 and 16). The results obtained with the proposed algorithm are shown in Figs. 15 and 17, respectively. We can observe from these experiments that the locally optimal distance is improved when we use a larger neighbourhood, which is coherent in a gradient descent paradigm.

6.4. Multi-scale handling of the discrete rigid transformation graph

The gradient descent approach proposed for handling the rigid registration issue requires to compute, at each iteration of the descent, the k-neighbours of the current DRT. The cost of the descent is then $\mathcal{O}(skN^2)$, where s is the number of iterations until reaching a local minimum. On the one hand, when s is small, a local minimum is reached rapidly, with a high probability to converge on a non-satisfactory result, unless the initial DRT is chosen "close" to the optimal solution. On the other hand, when s is large, the overall computational cost is high.

In order to allow a robust, non-costly gradient descent registration, without requiring an accurate initialization of the process, a solution consists of considering a multiscale variant of the proposed scheme. This is tractable since the DRT graph presents a structure that is compliant with multiscale decomposition.

More precisely, for a scale parameter $\frac{1}{3}$, i.e., by considering "superpixels" on intervals $[-\frac{3}{2}, \frac{3}{2}]^2$ instead of $[-\frac{1}{2}, \frac{1}{2}]^2$, the tipping surfaces/curves of the initial DRTs are "subsampled" by a factor 3, as illustrated in Fig. 18.

Based on this fact, a first gradient descent can be considered on a coarse image of size $N/3^n \times N/3^n$, sampled from the real image, with a computational cost $\mathcal{O}(sk(N/3^n)^2)$. The local minimum obtained at this scale can then be used as initialization

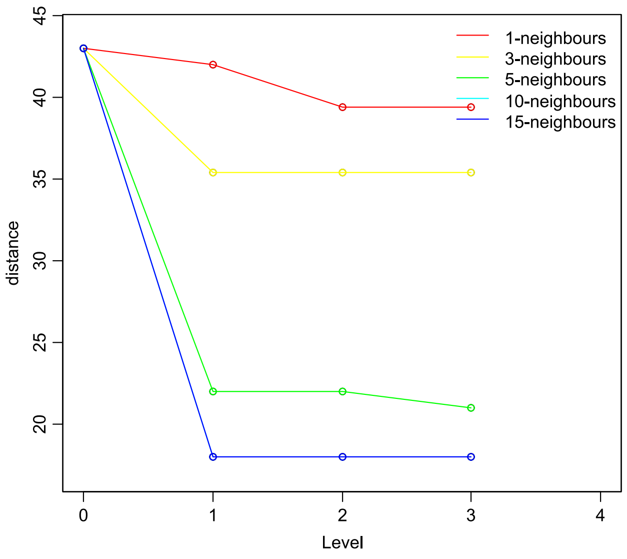


Fig. 19. Distance evolution at each scale level during the multi-scale local search for the inputs in Fig. 14 using the same seed as in Fig. 14.

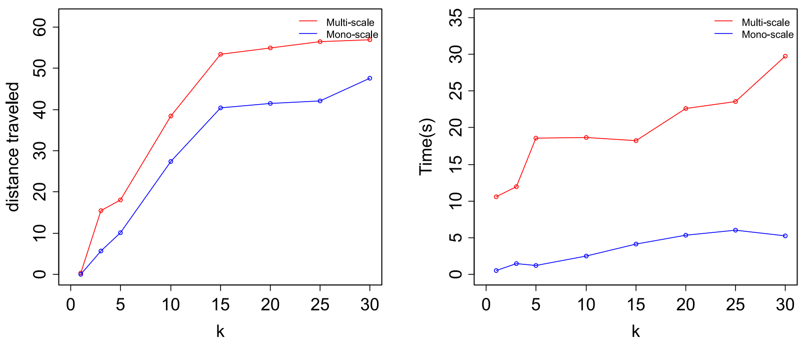


Fig. 20. Comparison between the k-neighbours and the multi-scale local search. Left: difference of distances from the initial DRT to that of local minimum for different k values. Right: total run-time of both approaches for different k values.

for a new gradient descent in a finer image of size $N/3^{n-1} \times N/3^{n-1}$ with a cost $\mathcal{O}(sk(N/3^{n-1})^2)$, and so on, until processing the real image of size $N \times N$, with a cost $\mathcal{O}(skN^2)$. By that way, for an average number of s iterations at each scale, the overall "distance" on the DRT graph is $s.\sum_{k=0}^{n} 3^k = s.\mathcal{O}(3^{n+1})$, for a computational cost of $n.\mathcal{O}(skN^2)$. The speed-up of this multiscale approach is then $3^{n+1}/n$, *i.e.*, exponential compared to the standard gradient descent. In addition, it allows to initialize the process potentially far from the optimal solution.

Fig. 19 shows the experimental results using the multi-scale approach for the source and target images in Fig. 14(a) and (b) with the seed $(0.365, -0.045, 0.1423)$ for $k = 1, 3, 5, 10, 15$. Fig. 20 compares between the k-neighbours local search and the multi-scale local search approaches in terms of optimization result of local minimum and the run-time.

6.5. *Application case: registration of granular material images*

In [19], a study on the relative displacement of grains, under directional shear, was investigated. To this end, several experiments were performed, in which an assembly of granular objects is used to simulate the grains. The object volume was slowly moved in simple shear at constant stress, as illustrated in Fig. 21. Though under a homogeneous deformation, the granular objects do not have a homogeneous displacement. The objective of the experiments was to observe the behaviour, in terms of translations and rotations, of each granular object under such shear stress. To this end, 2D pictures were taken at different moments of the experiment. These pictures were used to track and calculate the rigid motion from one image to another.

We use the proposed method to perform rigid registration on such granular objects over the images. It should be noticed that segmentation and tracking of the objects were necessary before applying our registration approach. Figs. 22 and 23 show the results obtained for the input of Fig. 21.

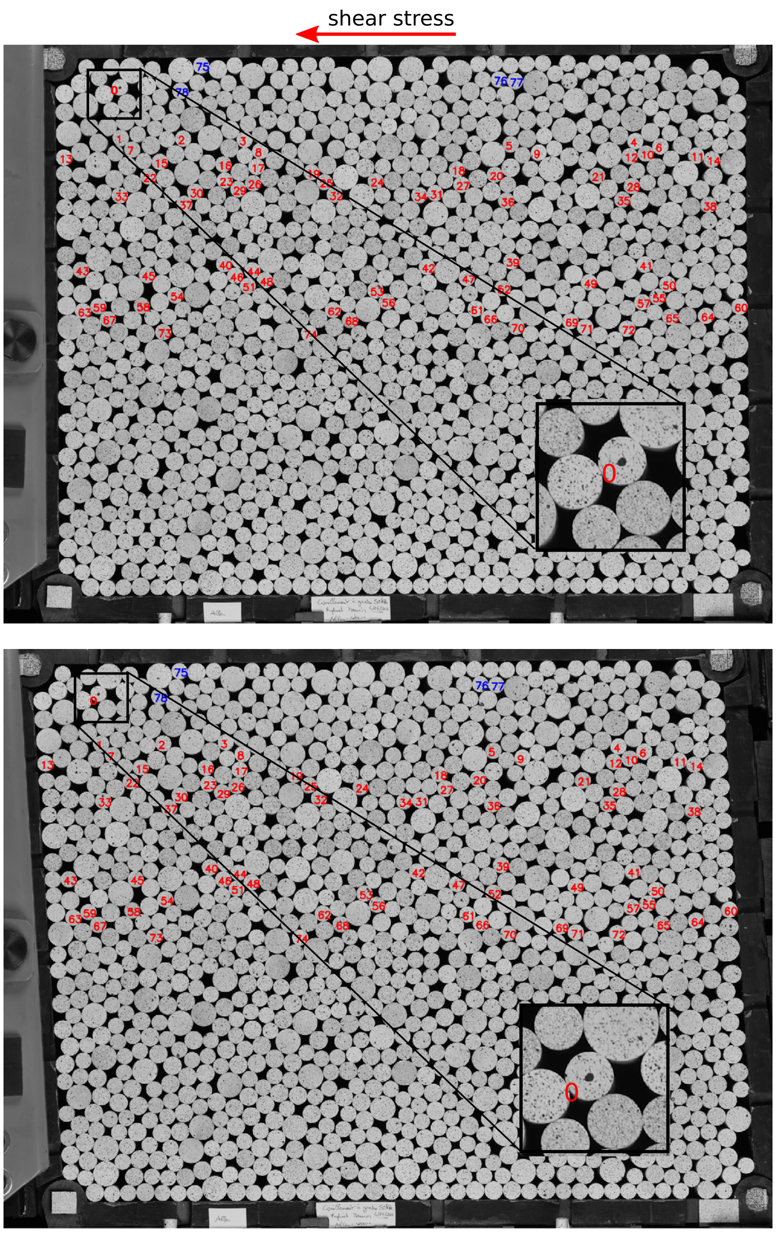


Fig. 21. Picture of granular objects – grains – slowly slip by a constant shear stress. In this experiment, the deformation is characterized by a rigid motion. Top: initial image. Bottom: image after shear stress application. The numerated grains are used for the experiments in Figs. 22 and 23, and Table 1.

Table 1
Distance results obtained with our method, using the initial transformation from the correlation method [19].

Grain n°	Correlation method [19]		Our method	
	Distance	Transformation	Distance	Transformation
75	41 049.1	$(-162, -1.10, 0.051)$	41 007.9	$(-161.506, -0.57, 0.073)$
76	32 230.8	$(-144, -5.73, 0.064)$	32 067.2	$(-143.998, -5.225, 0.085)$
77	25 618.8	$(-144, -7.02, 0.031)$	25 483.1	$(-143.443, -6.44, 0.0539)$
78	35 349.1	$(-160, 4.28, 0.061)$	35 247.7	$(-159.241, 5.067, 0.088)$

Still in [19], a digital image correlation approach was used to determine the motions of each granular object over the experiments. Using this result as initial transformations of our approach, Table 1 shows that the final results can be slightly improved by the proposed approach.

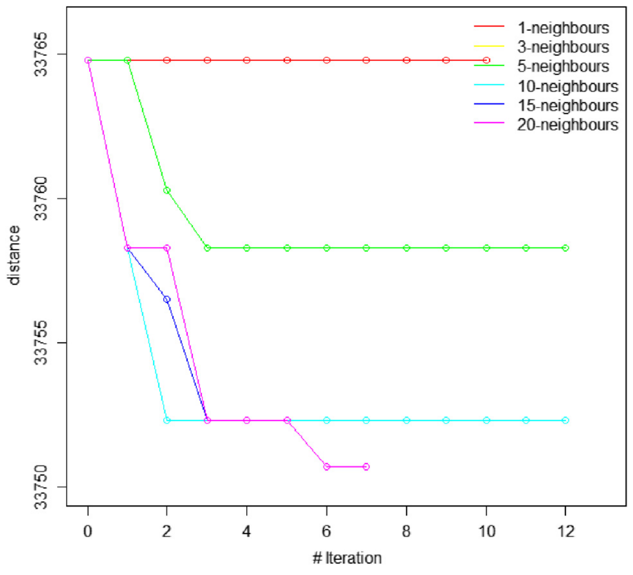


Fig. 22. Results of the local search obtained for the grain numbered 0 in Fig. 21, with respect to different k values.

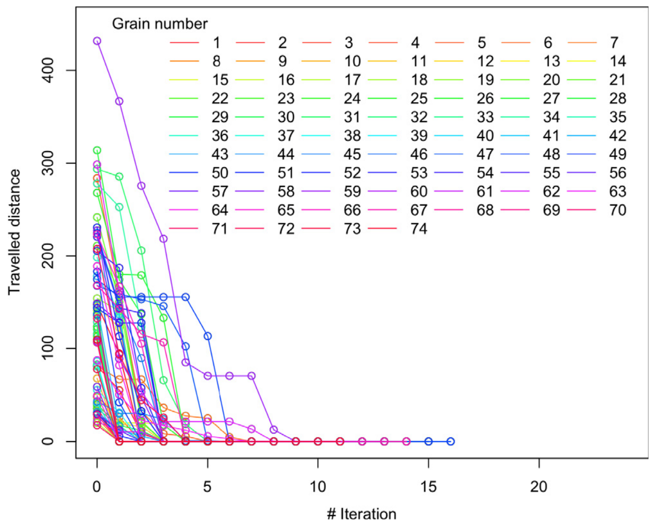


Fig. 23. Results of the local search obtained for the grains numbered from 1 to 74 in Fig. 21, with $k = 100$.

7. Conclusion

We have proposed a first attempt for a fully discrete framework of image registration under rigid transformations based on DRT graphs. Using a DRT graph, one can potentially generate *all* the transformed images of a given image under registration. Such a generation of transformed images presents a complexity in linear time with respect to the size of DRT graph which is $\mathcal{O}(N^9)$ for $N \times N$ is the size of the image. In this article, we have developed an efficient algorithm for a local computation of DRT graph with a complexity in linear time with respect to the image size. More precisely, this algorithm consists of computing a neighbourhood of a given DRT within a DRT graph, without requiring the computation of the whole graph. Thanks to the proposed algorithm, a local search can then be carried out via a gradient descent, by computing on-the-flight the useful local part of the DRT graph, to determine optimal transformations for registration issue. In particular, each iteration of this descent can be performed in linear time with respect to the size of the image, while a sublinear complexity can be reached with multiscale strategies.

This first algorithmic and experimental discussion opens the way to further developments of combinatorial approaches of image registration. On the one hand, the ways to extend this framework of rigid registration from 2D to 3D images will be investigated. This will require, in a first time, to accurately understand the subtleties of the analogue of the DRT graph for 3D transformations, and to solve some analytical open issues related to the exact computation of trigonometric surface intersection in 3D. On the other hand, the explicit decomposition of global 2D non-rigid registration into local 2D rigid registrations, may open the way to the discrete handling of image processing problems requiring the actual computation of complex deformation fields, allowing, for instance, to directly embed explicit topological constraints in the considered image models.

In this article, we have considered the case of rigid transformations. Further work will extend this framework to other classes of geometric transformations such as scaling, affine, projection, etc. Another perspective is to consider the non-rigid deformations as composition of local rigid transformations. Rigid transformations are restricted in usage as a global transformation. In future work we plan to approximate other registration frameworks by decomposition over collections of rigid transformations.

Acknowledgements

The authors thank Edward Andò and Gioacchino Viggiani of Laboratoire 3SR, Grenoble, France for providing the data for the experiments, as well as helpful discussions concerning this work. They would like also to thank Frank Schmidt of University of Freiburg, Germany for helpful discussions and providing Ref. [3].

References

[1] M. Amintoosi, M. Fathy, N. Mozayani, A fast image registration approach based on SIFT key-points applied to super-resolution, Imaging Sci. J. 60 (2011) 185–201.
[2] J. Ashburner, K.J. Friston, Diffeomorphic registration using geodesic shooting and Gauss–Newton optimisation, NeuroImage 55 (2011) 954–967.
[3] Y. Boykov, V. Kolmogorov, D. Cremers, A. Delong, An integral solution to surface evolution PDEs via geo-cuts, in: ECCV, Proc, in: LNCS, vol. 3953, Springer, 2006, pp. 409–422.
[4] J. Flusser, B. Zitová, T. Suk, Moments and Moment Invariants in Pattern Recognition, Wiley, 2009.
[5] C.A.R. Hoare, Algorithm 65: Find, Communications of the ACM, vol. 4, 1961, pp. 321–322.
[6] T. Hou, H. Qin, Robust dense registration of partial nonrigid shapes, IEEE Trans. Vis. Comput. Graphics 18 (2012) 1268–1280.
[7] Y. Kenmochi, P. Ngo, H. Talbot, N. Passat, Efficient neighbourhood computing for discrete rigid transformation graph search, in: DGCI, Proc., in: LNCS, vol. 8668, Springer, 2014, pp. 99–110.
[8] S. Klein, M. Staring, J.P.W. Pluim, Evaluation of optimization methods for nonrigid medical image registration using mutual information and B-splines, IEEE Trans. Image Process. 16 (2007) 2879–2890.
[9] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110.
[10] D. Lowe, Object recognition from local scale-invariant features, in: ICCV, Proc., 2014, pp. 1150–1157.
[11] I.S. Molchanov, P. Terán, Distance transforms for real-valued functions, J. Math. Anal. Appl. 278 (2) (2003) 472–484.
[12] P. Ngo, Y. Kenmochi, N. Passat, H. Talbot, On 2D constrained discrete rigid transformations, Ann. Math. Artif. Intell. 75 (2015) 63–193.
[13] P. Ngo, Y. Kenmochi, N. Passat, H. Talbot, Combinatorial properties of 2D discrete rigid transformations under pixel-invariance constraints, in: IWCIA, Proc., in: LNCS, vol. 7655, Springer, 2012, pp. 234–248.
[14] P. Ngo, Y. Kenmochi, N. Passat, H. Talbot, Combinatorial structure of rigid transformations in 2D digital images, Comput. Vis. Image Underst. 117 (2013) 393–408.
[15] P. Ngo, Y. Kenmochi, N. Passat, H. Talbot, Topology-preserving conditions for 2D digital images under rigid transformations, J. Math. Imaging Vision 49 (2014) 418–433.
[16] P. Ngo, A. Sugimoto, Y. Kenmochi, N. Passat, H. Talbot, Discrete rigid transformation graph search for 2D image registration, in: PSIVT Workshops, Proc., in: LNCS, vol. 8334, Springer, 2013, pp. 228–239.
[17] V. Noblet, C. Heinrich, F. Heitz, J.-P. Armspach, 3-D deformable image registration: A topology preservation scheme based on hierarchical deformation models and interval analysis optimization, IEEE Trans. Image Process. 14 (2005) 553–566.
[18] F.P.M. Oliveira, J.M.R.S. Tavares, Medical image registration: A review, Comput. Methods Biomech. Biomed. Eng. 17 (2014) 73–93.
[19] V. Richefeu, G. Combe, G. Viggiani, An experimental assessment of displacement fluctuations in a 2D granular material subjected to shear, Géotech. Lett. 2 (2012) 113–118.
[20] R.A. Schowengerdt, Remote Sensing: Models and Methods for Image Processing, third ed., Elsevier Academic Press, 2007.
[21] H. Talbi, M.C. Batouche, Particle swarm optimization for image registration, in: ICICT, Proc., 2004, pp. 397–398.
[22] T.W.H. Tang, A.C.S. Chung, Non-rigid image registration using graph-cuts, in: MICCAI, Proc., in: LNCS, vol. 4791, Springer, 2007, pp. 916–924.
[23] N.J. Tustison, B.B. Avants, J.C. Gee, Directly manipulated free-form deformation image registration, IEEE Trans. Image Process. 18 (2009) 624–635.
[24] B. Zitová, J. Flusser, Image registration methods: A survey, Image Vis. Comput. 21 (2003) 977–1000.