Efficient Neighbourhood Computing for Discrete Rigid Transformation Graph Search[⋆]

Yukiko Kenmochi[1], Phuc Ngo[2], Hugues Talbot[1], and Nicolas Passat[3]

[1] Université Paris-Est, LIGM, CNRS, France
[2] CEA LIST – DIGITEO Labs, France
[3] Université de Reims Champagne-Ardenne, CReSTIC, France

Abstract. Rigid transformations are involved in a wide variety of image processing applications, including image registration. In this context, we recently proposed to deal with the associated optimization problem from a purely discrete point of view, using the notion of discrete rigid transformation (DRT) graph. In particular, a local search scheme within the DRT graph to compute a locally optimal solution without any numerical approximation was formerly proposed. In this article, we extend this study, with the purpose to reduce the algorithmic complexity of the proposed optimization scheme. To this end, we propose a novel algorithmic framework for just-in-time computation of sub-graphs of interest within the DRT graph. Experimental results illustrate the potential usefulness of our approach for image registration.

Keywords: image registration, discrete rigid transformation, discrete optimization, DRT graph.

1 Introduction

1.1 Discrete Rotations and Discrete Rigid Transformations

In continuous spaces (*i.e.*, $\mathbb{R}^n$), rotations are some of the simplest geometric transformations. However, in the discrete spaces (*i.e.*, $\mathbb{Z}^n$), their analogues, namely discrete rotations, are more complex. The induced challenges are not simply due to high-dimensionality: indeed, even in $\mathbb{Z}^2$, discrete rotations raise many difficulties, deriving mainly from their non-necessary bijectivity [1]. In this context, discrete rotations – and the closely related discrete rigid tansformations – have been widely investigated.

From a combinatorial point of view, discrete rotations have been carefully studied [2–4], in particular to shed light on remarkable configurations induced by the periodicity of rotations with respect to the discrete grid. At the frontier between combinatorics and algorithmics, the problem of 2D pattern matching under discrete rotations has also been explored [5, 6].

From an algorithmic point of view, efforts have been devoted to effectively compute discrete rotations. In particular, the quasi-shear rotations [7, 8] were introduced to preserve bijectivity, by decomposing rotations into successive quasi-shears.

[⋆] The research leading to these results has received funding from the French *Agence Nationale de la Recherche* (Grant Agreement ANR-2010-BLAN-0205).

Finally, from an applicative point of view, discrete rotations have been used for image/signal processing purposes [9, 10]. Other strategies have also been proposed to pre-process 2D images in order to guarantee the preservation of topological properties under discrete rigid transformations [11].

Recently, we proposed a new paradigm to deal with discrete rotations, and more generally rigid transformations. This paradigm relies on a combinatorial structure, called discrete rigid transformation graph (DRT graph, for short) [12]. This structure describes the quantification of the parameter space of rigid transformations, in the framework of hinge angles, pioneered in [13–15].

The DRT graph has already allowed us to contribute to the state of the art on rigid transformations from a combinatorial point of view, by establishing the complexity of "free" [12] and "constrained" [16] discrete rigid transformations. From an algorithmic point of view, it has been used to characterise topological defects in transformed images [17]. Finally, we recently started to explore the applicative possibilities offered by the DRT graph. In particular, we have considered its potential usefulness in the context of image registration [18].

1.2 Registration Issues

In the context of image processing, geometric transformations are often considered for registration purposes [19]. Registration is indeed a complex, often ill-posed problem, that consists of defining the transformation that is required to correctly map a source image onto a target image.

Registration is mandatory in various application fields, from remote sensing [20] to medical imaging [21]. According to the specificities of these fields, registration can implicate different types of images (2D, 3D) and transformations, both rigid and non-rigid. However, the problem remains almost the same in all applications. Given two images A and B, we aim at finding a transformation T^* within a given transformation space $\mathbb{T}$. This transformation minimizes a given distance d between the image A and the transformed image $T(B)$ of the image B by T, *i.e.*

$$T^* = \arg\min_{T \in \mathbb{T}} d(A, T(B)) \tag{1}$$

In recent works [18], we investigated how to use the DRT graph in order to solve this problem in the case of rigid registration of 2D images. The novelty of this approach, with respect to the state of the art, was to provide exact transformation fields, so as to avoid any interpolation process and numerical approximations.

In this context, a preliminary algorithm was proposed for computing a local minimum for Eq. (1), thus providing a solution in a neighbourhood of depth $k \geq 1$, to the above registration problem. This algorithm strongly relies on the DRT graph, and consists of exploring a sub-graph defined around a given vertex, modeling an initial transformation. Its time complexity was $O(m^k N^2)$, which is linear with respect to the image size, but exponential with respect to the neighbourhood depth (with m the size of the 1-depth neighbourhood).

1.3 Contribution

We propose an improved algorithm (Sec. 3), which dramatically reduces the exponential complexity of that developed in [18]. Indeed, we show that the k-depth neighbourhood of a DRT graph can be computed with a time complexity $O(kN^2)$ (Sec. 4). Experiments emphasise the methodological interest of the proposed approach (Sec. 5).

2 Introduction to Discrete Rigid Transformation Graphs

2.1 Rigid Transformation Space

In the continuous space $\mathbb{R}^2$, a rigid transformation is a bijection $\mathcal{T} : \mathbb{R}^2 \to \mathbb{R}^2$, defined, for any $\boldsymbol{x} = (x, y) \in \mathbb{R}^2$, by

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

where $a_1, a_2 \in \mathbb{R}$ and $\theta \in [0, 2\pi[$ ($\mathcal{T}$ is sometimes noted $\mathcal{T}_{a_1 a_2 \theta}$). In order to apply such rigid transformations on $\mathbb{Z}^2$, a post-processing digitization is required. More precisely, a digitized rigid transformation $T : \mathbb{Z}^2 \to \mathbb{Z}^2$ is defined as $T = D \circ \mathcal{T}$ where $D : \mathbb{R}^2 \to \mathbb{Z}^2$ is a rounding function. In other words, for any $\boldsymbol{p} = (p, q) \in \mathbb{Z}^2$, we have

$$T(\boldsymbol{p}) = \begin{pmatrix} p' \\ q' \end{pmatrix} = D \circ \mathcal{T}(\boldsymbol{p}) = \begin{pmatrix} [p\cos\theta - q\sin\theta + a_1] \\ [p\sin\theta + q\cos\theta + a_2] \end{pmatrix} \tag{3}$$

The use of the rounding function D implies that digitized rigid transformations are not continuous within the 3D parameter space induced by a_1, a_2 and θ. The transformations leading to such discontinuities are called critical transformations. In the space (a_1, a_2, θ), the subspace of critical transformations is composed of 2D surfaces $\Phi_{pqp'}$ and $\Psi_{pqq'}$, analytically defined, for any $\boldsymbol{p} = (p, q) \in \mathbb{Z}^2$ and any vertical (resp. horizontal) pixel boundary $x = p' + \frac{1}{2}$ (resp. $y = q' + \frac{1}{2}$) with $p' \in \mathbb{Z}$ (resp. $q' \in \mathbb{Z}$), by

$$\Phi_{pqp'} : p\cos\theta - q\sin\theta + a_1 = p' + \frac{1}{2} \tag{4}$$

$$\Psi_{pqq'} : p\sin\theta + q\cos\theta + a_2 = q' + \frac{1}{2} \tag{5}$$

For a given triplet (p, q, p') (resp. (p, q, q')), $\Phi_{pqp'}$ (resp. $\Psi_{pqq'}$) is called a vertical (resp. horizontal) tipping surface in the parameter space (a_1, a_2, θ), and a vertical (resp. horizontal) tipping curve in the 2D plane (a_1, θ) (resp. (a_2, θ)).

For an image of size $N \times N$, $\Phi_{pqp'}$ and $\Psi_{pqq'}$ verify $p, q \in [\![0, N-1]\!]$ and $p', q' \in [\![0, N]\!]$. Examples of tipping surfaces and curves are illustrated in Fig. 1.

2.2 Discrete Rigid Transformation Graph

A set of tipping surfaces induces a subdivision of the (a_1, a_2, θ) space into classes, each consisting of transformations $\mathcal{T}_{a_1 a_2 \theta}$ such that $(a_1, a_2, \theta) \mapsto T = D \circ \mathcal{T}_{a_1 a_2 \theta}$ is constant. These classes – called discrete rigid transformations (DRTs) – indeed form 3D

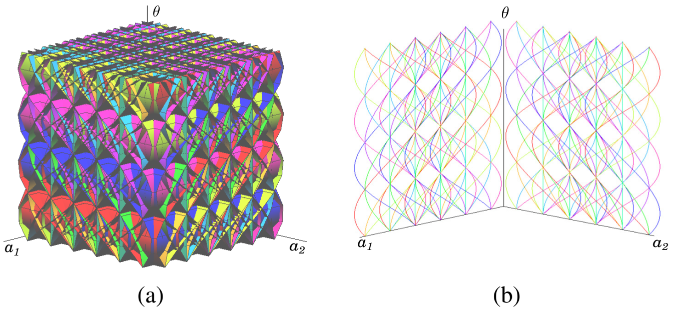

Fig. 1. (a) Tipping surfaces in the space (a_1, a_2, θ), and (b) their tipping curves [16]

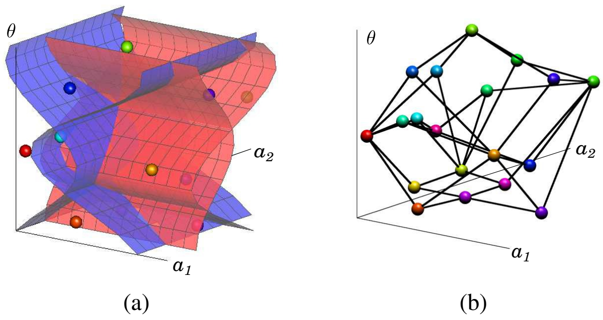

Fig. 2. (a) Subdivision of the (a_1, a_2, θ) parameter space into 3D cells by tipping surfaces, and (b) the associated DRT graph [17]

cells, bounded by tipping surfaces that correspond to discontinuities. By mapping each cell onto a vertex, and each tipping surface piece onto an edge, in a Voronoi/Delaunay paradigm, we can model this subdivided parameter space as a graph, called the DRT graph, as illustrated in Fig. 2.

Definition 1 ([12]). *A DRT graph $G = (V, E)$ is defined such that (i) each vertex $v \in V$ models a DRT; and (ii) each labelled edge $e = (v, w, f) \in E$, where f is either $\Phi_{pqp'}$ or $\Psi_{pqq'}$, connects two vertices $v, w \in V$ sharing the tipping surface f as boundary.*

For a given image I, each vertex is associated with a unique transformed image, induced by the DRT corresponding to the vertex. The existence of an edge between two vertices indicates a "neighbouring" relation between the two associated DRTs. More precisely the two transformed images differ by at most one over the N^2 pixels of I; the edge label f provides this information. This allows us to use the DRT graph to produce all the transformed images via successive elementary (*i.e.*, single-pixel) modifications.

2.3 Discrete Rigid Transformation Graph and Image Registration

The registration problem formalised in Eq. (1) consists of finding the transformation that best maps a source image onto a target image, with respect to a given distance.

In the discrete framework, the number of transformations is actually finite. In particular, in the case of rigid registration, the solution(s) to Eq. (1) can be found within the DRTs exhaustively modeled by the DRT graph. In other words, by considering a brute-force search, a solution, *i.e.*, a global optimum, can be determined. However, the DRT graph G of an image of size $N \times N$, has a high space complexity $O(N^9)$ [12] that induces the same time complexity both for its construction and exhaustive search.

This limits exploration of the whole structure to relatively small images. Nevertheless, as already discussed in [18], it is possible to perform a local search on G in order to determine a local optimum.

2.4 Local Search on a Discrete Rigid Transformation Graph

To find such an optimum, a local search begins at a given transformation, *i.e.*, a chosen vertex v of G. Then, it moves towards a better solution in its neighbourhood – following a gradient descent – as long as an improved solution can be found. Beyond the choice of the initial vertex – often guided by the application context – the most critical issue is the choice of a "good" search area around this vertex, *i.e.*, a depth of its neighbourhood. In particular, the trade-off is time efficiency versus exhaustiveness.

The neighbourhood of depth 1, noted $\mathcal{N}^1(v)$, actually corresponds to the set $\mathcal{N}(v)$ of vertices adjacent to v in G. More generally, neighbourhoods of depth $k \geq 1$, also called k-neighbourhoods, are then recursively obtained as

$$\mathcal{N}^k(v) = \mathcal{N}^{k-1}(v) \cup \bigcup_{u \in \mathcal{N}^{k-1}(v)} \mathcal{N}(u) \tag{6}$$

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

Our initial algorithm [18] was directly mapped on this recursive definition. As a consequence, this approach led to a high time complexity $O(m^k N^2)$, that is exponential with respect to the depth k of the neighbourhood with vertex degree m, which is supposed to be constant in average (Sec. 4.2). In the next section, we propose a more efficient algorithm, that removes this exponential cost.

3 k-Neighbourhood Construction Algorithm

We now propose an algorithm that efficiently computes the part of a DRT graph that models the neighbourhood of depth k around a given vertex. To this end, we need to handle the analytical representation of the cells associated to the DRT graph vertices, inside the subdivided parameter space of (a_1, a_2, θ) (Sec. 3.1). Then, we develop a construction strategy that relies on a sweeping plane technique introduced in [12] (Sec. 3.2). The final algorithm is described and formalized in Sec. 3.3.

3.1 Tipping Surfaces Associated to a Discrete Rigid Transformation

A vertex v of a DRT graph G corresponds to one discrete rigid transformation, that induces a unique transformed image I_v obtained by applying this transformation on an initial image I. In other words, for each pixel (p_i, q_i) of I_v, we know which pixel (p'_i, q'_i) of I transfers its value to (p_i, q_i). This correspondence is modeled by the following inequalities deriving from Eq. (3)

$$p'_i - \frac{1}{2} < p_i \cos\theta - q_i \sin\theta + a_1 < p'_i + \frac{1}{2} \tag{7}$$

$$q'_i - \frac{1}{2} < p_i \sin\theta + q_i \cos\theta + a_2 < q'_i + \frac{1}{2} \tag{8}$$

For an image I of size $N \times N$, each of the N^2 pixels generates 4 such inequalities. In the parameter space of (a_1, a_2, θ), the obtained $4N^2$ inequalities then define a 3D cell, denoted by $\mathcal{R}_v$, which gathers all the parameter triplets associated to the discrete rigid transformation corresponding to the vertex v.

When interpreting these inequalities in terms of tipping surfaces/curves (see Eqs. (4–5)), it appears that for each pixel of I_v, Eqs. (7–8) 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 $\mathbf{S}_2^-(I_v)$).

We derive from Eqs. (7–8), that any cell $\mathcal{R}_v$ is directionally convex along the $a_\star$-axes [16]. This implies that for any θ value where it is defined, $\mathcal{R}_v$ is bounded by at least one upper (resp. lower) tipping surface, which constitutes the upper (resp. lower) part of its boundary in each $a_\star$-direction. This property can be used for constructing a DRT graph locally, or for obtaining topological information from a DRT graph such as a neighbourhood. One may notice that it is sufficient to consider only tipping surfaces of $\bigcup(\mathbf{S}_\star^+(I_v) \cup \mathbf{S}_\star^-(I_v))$ in order to obtain the k-neighbourhood of v, if $k < N$.

3.2 Sweeping Plane Algorithm for DRT Sub-graph Construction

In our new algorithm, the purpose is to build a k-neighbourhood "similarly" to the construction of a 1-neighbourhood in our previous version [18], that is by using a sweeping plane technique from one value θ_v within $\mathcal{R}_v$, to both the left-hand and right-hand sides along the θ-axis in the space (a_1, a_2, θ).

The differences between this new algorithm and the former are twofold. On the one hand, the range of the considered θ values is wider. Indeed, the sweep must be carried out inside $\mathcal{R}_v$ but also outside. On the other hand, a larger number of tipping surfaces are considered around $\mathcal{R}_v$, while only immediate neighbours were previously involved.

To ease the understanding of this algorithm, we first recall the general idea of the sweeping plane technique.

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 a given range $[\theta_{start}, \theta_{end}]$. By comparison to [12], the plane is then swept from θ_{start} to θ_{end}, instead of 0 to 2π. From the very definition of tipping surfaces, this plane is subdivided into $(s_1 + 1) \times (s_2 + 1)$ 2D rectangular cells, generated by its intersection with the tipping surfaces of $\mathbf{S}$. More precisely, we have $(s_\star + 1)$ divisions in each $a_\star$-direction, except at the intersection of tipping

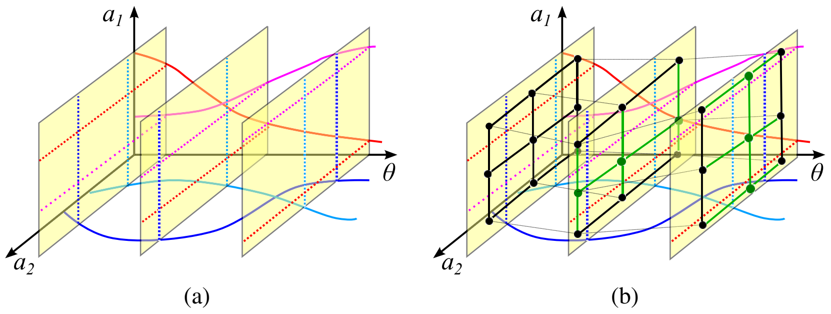

Fig. 3. DRT graph construction by the sweeping plane algorithm, with 2 vertical (blue, cyan) and 2 horizontal (red, magenta) tipping surfaces. (a) 3×3 rectangular cells generated by the tipping surfaces in each sweeping plane. (b) The associated DRT graph in each plane (in green: new vertices and edges in the second and third planes).

surfaces, where a rectangle disappears while a new appears. By observing these rectangle updates during the plane sweeping from θ_{start} to θ_{end}, we can construct the DRT sub-graph, where each rectangle corresponds to a vertex while each tipping surface between two rectangles corresponds to an edge. In other words, at each intersection of tipping surfaces, $s_\star$ new vertices and their associated $(3s_\star + 2)$ edges are generated, as illustrated in Fig. 3. (The reader is referred to [12] for more details.)

Our modified algorithm consists of using a topological sweep [22] in order to find the next closest intersection of tipping surfaces for updating the planar division. We consider at most $|\mathbf{S}| - 2$ intersections at each update, by considering only the intersections of consecutive tipping surfaces in their ordered structure in the sweeping plane along each $a_\star$-axis, and find the closest one among them. After each update, the modifications of such intersections can be performed in constant time. We can also ignore the intersections that are not in the range between θ_{start} and θ_{end}. In particular, since we have $|\theta_{end} - \theta_{start}| \ll 2\pi$, the number of intersections can be considered 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.

3.3 *k*-Neighbourhood Construction

Finding the neighbouring vertices and edges of a given vertex v with depth k, is actually equivalent to constructing the DRT sub-graph containing those vertices and edges around v. Here, we assume to know a value θ_v lying into $\mathcal{R}_v$, and we use it as initial value of the sweeping algorithm. The plane is thus swept twice, in the space (a_1, a_2, θ), along the two directions of the θ-axis.

The key-point is how to limit the construction of the DRT sub-graph. For this purpose we verify, for each generated vertex u, its neighbourhood depth $t_v(u)$ with respect to v. If $t_v(u) > k$ for all vertices in the current sweeping plane, the process ends.

Algorithm 1. k-neighbourhood computation (in the left-hand side along the θ-axis)

```
Input: A DRT $v$ (or its associated image $I_v$); a positive integer $k$.
Output: The DRT sub-graph $F = (V, E)$ containing the $k$-neighbours of $v$.
1  for $\star = 1, 2$ do
2      Determine the tipping surfaces associated to $v$: $\mathbf{S}^+_\star(I_v)$, $\mathbf{S}^-_\star(I_v)$ (Sec. 3.1).
3      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 $\theta_v$.
4      Find and sort the $k+1$ tipping surfaces that are lower (resp. upper) or equal to $f^+_\star$
       (resp. $f^-_\star$), and put them in an ordered set $\mathbf{S}_\star$.
5  Initialize $V = \emptyset$, $E = \emptyset$
6  Initialize $\theta_{prev} = \theta_v$
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 surface in
           $\mathbf{S}_\star$ for $\theta < \theta_{prev}$.
10     $\theta_{next} = \min\{\theta^+_1, \theta^-_1, \theta^+_2, \theta^-_2\} - \epsilon$ with $\epsilon \ll 1$
11     $\Delta F = Sweep(\mathbf{S}_1 \cup \mathbf{S}_2, \theta_{prev}, \theta_{next})$
12     if $\exists u \in \Delta V, t_v(u) \leq k$ then
13         $F = F \cup \Delta F$, $\theta_{prev} = \theta_{next}$
14         if the next intersecting surface with $f^+_\star$ (or $f^-_\star$) is in $\mathbf{S}_\star$ then
15             Exchange their order in $\mathbf{S}_\star$.
16         else
17             Replace $f^+_\star$ (or $f^-_\star$) in $\mathbf{S}_\star$ with the new intersecting surface.
18 until $\forall u \in \Delta V, t_v(u) > k$;
```

When a vertex u is created, its depth $t_v(u)$ depends on that of the two vertices w_1 and w_2 to which it is adjacent in the $a_\star$-direction of the tipping surface intersection. We then have $t_v(u) = 1 + \min\{t_v(w_1), t_v(w_2)\}$. (An iterative backtracking process is also needed to update the depth of $w_\star$ and its successive neighbours, whenever $t_v(w_\star) > t_v(u) + 1$.)

In each $a_\star$-direction, by considering the $(k+1)$ closest tipping surfaces around $\mathcal{R}_v$, we can obtain all the vertices u such that $t_v(u) \leq k$. In the θ-direction, we need to check if $t_v(u) > k$ for all vertices u in the current sweeping plane; if so, the sweeping ends.

The global process is described in Alg. 1. (Note that the algorithm describes only the k-neighbourhood construction in the left-hand side along the θ-axis, but the right-hand side can be constructed similarly.) The first loop (Lines 1–4) initializes the set of tipping surfaces that are needed to generate the k-neighbours of a given DRT v. We obtain $2(k+1)$ vertical (resp. horizontal) tipping surfaces close to $\mathcal{R}_v$ at $\theta = \theta_v$, and sort and store them in the lists $\mathbf{S}_\star$. In the second loop (Line 7), we first verify how long we can keep the same tipping surface sets $\mathbf{S}_\star$ (Lines 9–10), and then build a DRT sub-graph by using the $Sweep$ algorithm for this verified θ interval (Line 11). After verifying if there still exists a generated vertex whose neighbourhood depth is $\leq k$ (Line 12), we update the tipping surface sets $\mathbf{S}_\star$ for the next interval (Lines 14–17).

Obviously, F is not the smallest sub-graph G including the k-neighbours of v. To obtain G from F, we simply keep vertices whose neighbourhood depth is $\leq k$.

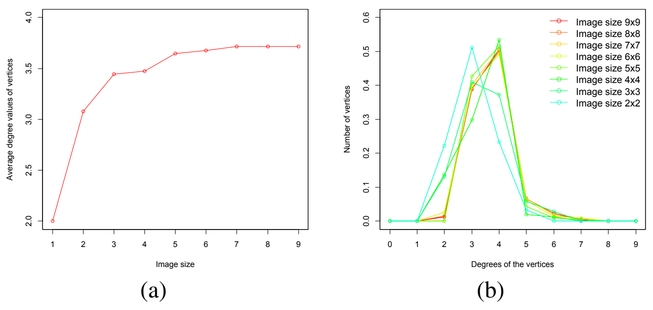

Fig. 4. (a) Average vertex degree in a 2D DRT graph. (b) Normalised vertex degree distribution in a 2D DRT graph.

4 Complexity Analysis

4.1 Time Complexity of k-Neighbourhood Construction Algorithm

In order to obtain the initial $2(k + 1)$ vertical (resp. horizontal) tipping surfaces of $\mathbf{S}_\star$, the time complexity is $O(N^2)$ for Line 2; $O(N^2)$ for Line 3 on the average case if we use Hoare's FIND algorithm [23]; and $O(N^2)$ and $O(k \log k)$ for finding and sorting the tipping surfaces in Line 4, respectively. Then, we carry out the plane sweep for each updated $\mathbf{S}_1 \cup \mathbf{S}_2$. For each iteration in the loop, the most costly parts are Lines 9 and 11, which require $O(N^2)$ and $O(k^2)$, respectively.

The next question concerns the number of updates for $\mathbf{S}_1 \cup \mathbf{S}_2$. If m is the degree of any vertex u of a DRT graph, this update number can be estimated as $m(2k + 1)$, since the union of $\mathcal{R}_u$ for all u in the k-neighbourhood of a given vertex v contains at most $2k + 1$ adjacent $\mathcal{R}_u$ in the θ-direction. Therefore, the time complexity is $O(mkN^2)$ for this iterative plane sweep loop.

The time complexity of Alg. 1 is thus $O(mkN^2)$, which is significantly lower than that of our previous algorithm [18], namely $O(m^kN^2)$. We observe, in the next section, that m can be estimated as a low constant value, leading to a final complexity of $O(kN^2)$.

4.2 Average Degree of DRT Graphs

The DRT graph space complexity for an image of size $N \times N$ is $O(N^9)$, both for vertices and edges [12]. In other words, the number of vertices and that of edges grow at the same rate. We can then infer that m is actually bounded, independently of N.

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. Indeed, the considered tipping curves are not straight lines, while their very regular structure implies that many curves often intersect at a same point.

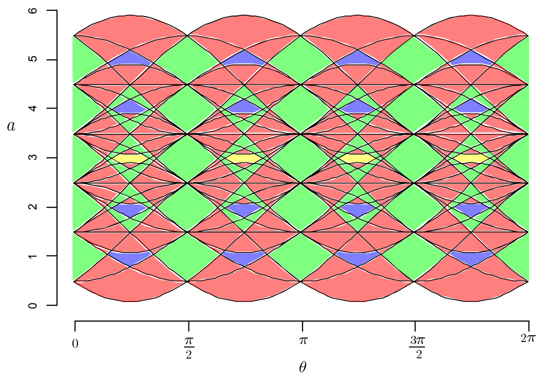

Fig. 5. Degree distribution in a 2D DRT graph, viewed in the dual subdivision of the parameter space. Each colour represents a given degree, that corresponds here to the number of curves bounding each cell (3: red, 4: green, 5: blue; 6: yellow).

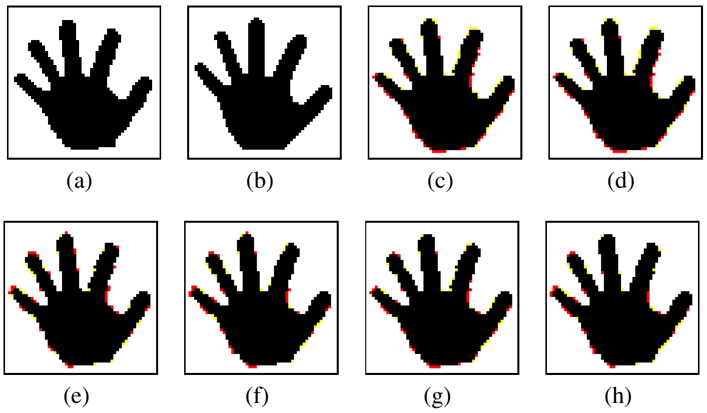

Fig. 6. Input images and results of the iterated local search for image registration. (a) Reference image, (b) target image, and (c) the initial transformed image of (b) with $(a_1, a_2, \theta) = (0.365, -0.045, 0.1423)$. (d–h) Local optima obtained from (c) by using k-neighbours for $k = 1, 3, 5, 10, 15$ respectively. Note that in (c–h), pixels are coloured if they are different from those in (a); yellow (resp. red) pixels are white (resp. black) in (c–h) and black (resp. white) in (a).

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 very definition of the DRT graph, 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 $2 \times 4 = 8$. This is confirmed by the experimental analysis, illustrated in Fig. 4(a).

In practice, the maximal degree of the vertices within a DRT graph also remains close to this average value. Indeed, the histograms depicted in Fig. 4(b) show that the

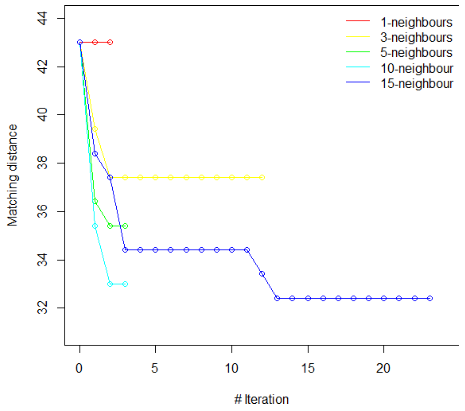

Fig. 7. Distance evolution during iterations of local search for the inputs in Fig. 6 (a) and (b), from the initial transformation in Fig. 6 (c), with respect to different depths k

2D DRT vertex degrees converge rapidly to a stable distribution that contains mainly degrees of value 3 and 4 (with a maximal value experimentally identified at 8). More qualitatively, Fig. 5 illustrates the distribution of these degrees of a 2D DRT graph.

5 Experiments

Iterated local search was applied to image registration. In this section we validate Alg. 1 in practice, and we observe its local behaviour when varying k. For simplicity, we use the same experimental settings as in [18], *i.e.,* two input binary images and a signed distance [24] for Eq. (1). In order to find an initial transformation, we use the first and second order central moments of a binary shape [25]. Experiments are carried out with different neighbourhood sizes, $k = 1, 3, 5, 10, 15$ on binary images of size 53×53 from the initial transformation, as illustrated in Fig. 6. We can observe in Fig. 7 that the locally optimal distance improves when we use a larger neighborhood, which is coherent in a gradient descent paradigm.

6 Conclusion

We have significantly improved the time complexity of the process of computing a neighbourhood of given depth within a DRT graph, without requiring the computation of the whole graph. This time complexity may be reduced in some cases, in particular if the image is binary by dealing only with the pixels that compose the binary object border. The proposed applications only validate our approach as a proof of concept. Nevertheless, an exact – *i.e.*, numerical error-free – strategy is novel in the field of image registration and may open the way to new image processing paradigms. In future work we will explore the notion of DRT graph in $\mathbb{Z}^3$.

References

1. Nouvel, B., Rémila, É.: Characterization of bijective discretized rotations. In: Klette, R., Žunić, J. (eds.) IWCIA 2004. LNCS, vol. 3322, pp. 248–259. Springer, Heidelberg (2004)

2. Nouvel, B., Rémila, E.: Configurations induced by discrete rotations: Periodicity and quasi-periodicity properties. Discrete Appl. Math. 147, 325–343 (2005)
3. Berthé, V., Nouvel, B.: Discrete rotations and symbolic dynamics. Theor. Comput. Sci. 380, 276–285 (2007)
4. Nouvel, B.: Self-similar discrete rotation configurations and interlaced Sturmian words. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) DGCI 2008. LNCS, vol. 4992, pp. 250–261. Springer, Heidelberg (2008)
5. Jacob, M.A., Andres, E.: On discrete rotations. In: Proc. DGCI, pp. 161–174 (1995)
6. Amir, A., Kapah, O., Tsur, D.: Faster two-dimensional pattern matching with rotations. Theor. Comput. Sci. 368, 196–204 (2006)
7. Reveillès, J.P.: Géométrie discrète, calcul en nombres entiers et algorithmique. Thèse d'État, Université Strasbourg 1 (1991)
8. Andres, E.: The quasi-shear rotation. In: Miguet, S., Ubéda, S., Montanvert, A. (eds.) DGCI 1996. LNCS, vol. 1176, pp. 307–314. Springer, Heidelberg (1996)
9. Richman, M.S.: Understanding discrete rotations. In: Proc. ICASSP, vol. 3, pp. 2057–2060. IEEE (1997)
10. Andres, E., Fernandez-Maloigne, C.: Discrete rotation for directional orthogonal wavelet packets. In: Proc. ICIP, vol. 2, pp. 257–260. IEEE (2001)
11. Ngo, P., Passat, N., Kenmochi, Y., Talbot, H.: Topology-preserving rigid transformation of 2D digital images. IEEE T. Image Process. 23, 885–897 (2014)
12. Ngo, P., Kenmochi, Y., Passat, N., Talbot, H.: Combinatorial structure of rigid transformations in 2D digital images. Comput. Vis. Image Und. 117, 393–408 (2013)
13. Nouvel, B.: Rotations discrètes et automates cellulaires. PhD thesis, École Normale Supérieure de Lyon (2006)
14. Nouvel, B., Rémila, É.: Incremental and transitive discrete rotations. In: Reulke, R., Eckardt, U., Flach, B., Knauer, U., Polthier, K. (eds.) IWCIA 2006. LNCS, vol. 4040, pp. 199–213. Springer, Heidelberg (2006)
15. Thibault, Y., Kenmochi, Y., Sugimoto, A.: Computing upper and lower bounds of rotation angles from digital images. Pattern Recogn. 42, 1708–1717 (2009)
16. Ngo, P., Kenmochi, Y., Passat, N., Talbot, H.: On 2D constrained discrete rigid transformations. Ann. Math. Artif. Intell. (in press), doi:10.1007/s10472-014-9406-x
17. Ngo, P., Kenmochi, Y., Passat, N., Talbot, H.: Topology-preserving conditions for 2D digital images under rigid transformations. J. Math. Imaging Vis. 49, 418–433 (2014)
18. Ngo, P., Sugimoto, A., Kenmochi, Y., Passat, N., Talbot, H.: Discrete rigid transformation graph search for 2D image registration. In: Huang, F., Sugimoto, A. (eds.) PSIVT 2013. LNCS, vol. 8334, pp. 228–239. Springer, Heidelberg (2014)
19. Zitová, B., Flusser, J.: Image registration methods: A survey. Image Vision Comput. 21, 977–1000 (2003)
20. Schowengerdt, R.A.: Remote Sensing: Models and Methods for Image Processing, 3rd edn. Elsevier Academic Press (2007)
21. Noblet, V., Heinrich, C., Heitz, F., Armspach, J.P.: Recalage d'images médicales. Tech Ing (MED910) (2014)
22. Edelsbrunner, H., Guibas, L.J.: Topologically sweeping an arrangement. Journal Comput. Syst. Sci. 38, 165–194 (1989); Corrig. 42, 249–251 (1991)
23. Hoare, C.A.R.: Algorithm 65: find. Commun. ACM 4, 321–322 (1961)
24. Boykov, Y., Kolmogorov, V., Cremers, D., Delong, A.: An integral solution to surface evolution PDEs via geo-cuts. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3953, pp. 409–422. Springer, Heidelberg (2006)
25. Flusser, J., Zitová, B., Suk, T.: Moments and Moment Invariants in Pattern Recognition. Wiley (2009)