Affine Puzzle: Realigning Deformed Object Fragments without Correspondences*

Csaba Domokos and Zoltan Kato

Department of Image Processing and Computer Graphics,
University of Szeged
H-6701 Szeged, PO. Box 652., Hungary
Fax: +36 62 546-397
`{dcs,kato}@inf.u-szeged.hu`

Abstract. This paper is addressing the problem of realigning broken objects without correspondences. We consider linear transformations between the object fragments and present the method through 2D and 3D affine transformations. The basic idea is to construct and solve a polynomial system of equations which provides the unknown parameters of the alignment. We have quantitatively evaluated the proposed algorithm on a large synthetic dataset containing 2D and 3D images. The results show that the method performs well and robust against segmentation errors. We also present experiments on 2D real images as well as on volumetric medical images applied to surgical planning.

1 Introduction

In this paper we address the problem of reassembling an object from its parts. This is also known as the *puzzle* problem, which is not only interesting from a theoretical point of view [1,2], but also arises in many application domains like archaeology [3] or medical imaging [4] *e.g.* bone fracture reduction [5,6,7]. The affine puzzle problem can be formulated as follows: Given a binary image of an object (the *template*) and another binary image (the *observation*) containing the fragments of the *template*, we want to establish the geometric correspondence between these images which reconstructs the complete *template* object from its parts. The overall distortion is a global nonlinear transformation with the following constraint: 1) the object parts are distinct (*i.e.* either disconnected or separated by segmentation), 2) all fragments of the *template* are available, but 3) each of them is subject to a *different* affine deformation.

A related problem is partial matching of shapes [8,9,10]. Partial matching addresses a particularly challenging setting of classical shape matching, where

---

* This research was partially supported by the Hungarian Scientific Research Fund (OTKA) – K75637 and by the grant CNK80370 of the National Office for Research and Technology (NKTH) & the Hungarian Scientific Research Fund (OTKA). The CT images were obtained from the University of Szeged, Department of Trauma Surgery and were used with permission of Prof. Endre Varga, MD.

K. Daniilidis, P. Maragos, N. Paragios (Eds.): ECCV 2010, Part II, LNCS 6312, pp. 777–790, 2010.

two shapes are dissimilar in general, but have significant similar parts. In this context, our problem would require to find a partial matching between the *template* and each fragments of the *observation*. Current approaches are usually based on the Laplace-Beltrami framework [11,10], but classical approaches like the Iterative Closest Point (ICP) [12] algorithm can also be used assuming an appropriate shape representation [8]. Considering the rather high computational complexity of these algorithms, this solution is far from optimal for our problem.

Another related problem is the piece-wise approximation of nonlinear deformations by locally linear transformations. In [13], the distortion is modeled as locally affine but globally smooth transformation, which accounts for local and global variations in image intensities. The classical solution [14] comprises identifying point correspondences based on salient points between the images and then either a time consuming optimization procedure or the solution of a system of equations provide the parameters of the unknown deformation. Finding reliable point correspondences between the images is a difficult problem on its own.

Most of the existing solutions to the puzzle problem [1,2,3] consist in matching fragment-pairs to find neighbors, which are then reassembled by a rigid body transformation. In [1], Kong and Kimia propose a 2D curve matching technique based on the geometric features of puzzle pieces. The solution is obtained by a recursive grouping of triples using a best-first search strategy. The method can be extended to 3D fragments scanned by a laser range finder, where a pair of ridges are matched using a generalization of the 2D curve matching approach. In [3], the rather high computational complexity of curve matching is reduced by adopting a multiscale technique. Papaioannou *et al.* address the problem of 3D object reconstruction using only the surface geometry of fragments, assuming no information about the final model to be reconstructed [2]. The basic idea of the method is that the best fit of two 3D fragments is likely to occur at their relative pose, which minimizes the point-by-point distance between the mutually visible faces of the fragments. Matched pieces are then glued via a rigid-body transformation.

Although classical approaches may account for a *template* object by incorporating a set of constraints to improve the overall performance, they are primarily targeted to problems where a *template* is not available, *e.g.* archaeology [3]. On the other hand, there are many applications where a *template* object is available: In industrial applications usually 3D models of manufactured parts can be easily produced. In medical imaging an *atlas* can be used or, by taking advantage of the symmetry of the human body, the intact bone can provide a *template* for bone fracture reduction, as shown in Section 5.3. Therefore we address this important setting of the puzzle problem and propose a generic solution which is then applied to 2D and 3D transformations. The methodology adopted here is similar in spirit to the affine matching methods of [15] and [16]. However, none of these works addresses the puzzle problem. [16] assumes that both images contain the same number of shapes and radiometric information is available. Based on these informations, Hagege and Francos construct a linear system of equations which provides the parameters of the aligning transformations. Since the partitioning of the *template* is not available, this method cannot be used here. In [15], Domokos and Kato presented an elegant solution to recover affine deformations between 2D shapes. This method is also unable to solve the puzzle problem because the deformation is nonlinear and there is no direct correspondence between the *template* and its observed fragments.

In Section 2, a general solution is proposed followed by Section 3 about the numerical implementation issues and Section 4 presenting the application of our method for various linear transformations. Finally, Section 5 presents quantitative results on 2D and 3D synthetic datasets as well as on various real images and Section 6 concludes the paper.

2 Realigning Object Parts

Given an n dimensional *template* object and an *observation* containing its affine deformed fragments, we want to recover the transformations realigning these shapes into their original position on the *template.* Let us denote the homogeneous point coordinates of the *template* and *observation* by $\mathbf{x} = [x_1, \ldots, x_n, 1]$ and $\mathbf{y} = [y_1, \ldots, y_n, 1] \in \mathbb{P}^n$. Furthermore, let $\ell \in \mathbb{N}$ denote the number of fragments on the *observation.* The transformation aligning the *observation* with the *template* is a non-linear one, composed of ℓ linear transformations

$$\mathbf{A}_i = \begin{bmatrix} a_{i11} & a_{i12} & \cdots & a_{i1(n+1)} \\ \vdots & & \ddots & \vdots \\ a_{in1} & a_{in2} & \cdots & a_{in(n+1)} \\ 0 & 0 & \ldots & 1 \end{bmatrix} \qquad i = 1, \ldots \ell \, . \tag{1}$$

Since the *observation* has disjoint parts, we can assume that points of each deformed shape are labeled by the function $\lambda' : \mathbb{P}^n \to \{0, 1, \ldots, \ell\}$, which assigns 0 to the background. Obviously, there is a corresponding *hidden* labeling $\lambda : \mathbb{P}^n \to \{0, 1, \ldots, \ell\}$ which assigns the label i to the *template* points corresponding to the i^{th} shape. Our goal is to recover the affine matrices $\{\mathbf{A}_i\}_{i=1}^{\ell}$. The main challenges are that neither the partitioning (*i.e.* the hidden labeling λ) of the *template* nor correspondences between the shapes are known.

2.1 Solution for One Pair of Shapes

Let us first establish a solution for the i^{th} shape. The *template* and *observation* domains are denoted by $\mathcal{D}_i = \{\mathbf{x} \in \mathbb{P}^n | \lambda(\mathbf{x}) = i\}$ and $\mathcal{D}'_i = \{\mathbf{y} \in \mathbb{P}^n | \lambda'(\mathbf{y}) = i\}$, respectively. Note that $\mathcal{D}'_i$ is known but $\mathcal{D}_i$ is unknown. The points of these domains are related by the unknown transformation $\mathbf{A}_i$:

$$\mathbf{x} = \mathbf{A}_i \mathbf{y}. \tag{2}$$

One way to recover $\mathbf{A}_i$ is to establish point correspondences and then set up a system of equations from Eq. (2). Since $\mathcal{D}_i$ is unknown, finding correspondences is practically impossible. Therefore we are interested in a direct method without solving the correspondence problem. For that purpose, let us notice that that the relation in Eq. (2) remains valid when a function $\omega : \mathbb{P}^n \to \mathbb{R}$ is acting on both sides of the equation [15]:

$$\omega(\mathbf{x}) = \omega(\mathbf{A}_i \mathbf{y}) \; . \tag{3}$$

We then integrate out individual point correspondences [15] yielding

$$\int_{\mathcal{D}_i} \omega(\mathbf{x}) d\mathbf{x} = |\mathbf{A}_i| \int_{\mathcal{D}'_i} \omega(\mathbf{A}_i \mathbf{y}) d\mathbf{y}, \tag{4}$$

where the integral transformation $\mathbf{x} = \mathbf{A}_i \mathbf{y}$, $d\mathbf{x} = |\mathbf{A}_i| d\mathbf{y}$ has been applied, and $|\mathbf{A}_i|$ is the Jacobian determinant. Based on Eq. (4), we can construct as many equations as needed by making use of a set of linearly independent functions $\{\omega_j\}_{j=1}^m$, $m \geq n(n+1)$. The solution of the resulting nonlinear system of equations provides the parameters of $\mathbf{A}_i$[15].

2.2 Solving for All Shapes Simultaneously

We have established relations between the i^{th} shape-pair, but we know neither the correspondence between the shapes nor the partitioning $\mathcal{D}_i$ of the *template*. Would these information available, a pairwise alignment could be recovered by any standard binary registration method. Unfortunately, that would require to solve a partial matching problem [8] between each *observation* shape and the *template*, which is far from trivial. Therefore we will sum equations for all shape domains $\mathcal{D}_i$ and solve the problem simultaneously, estimating all parameters in one system of equations. Thus Eq. (4) becomes

$$\sum_{i=1}^{\ell} \int_{\mathcal{D}_i} \omega_j(\mathbf{x}) d\mathbf{x} = \sum_{i=1}^{\ell} |\mathbf{A}_i| \int_{\mathcal{D}'_i} \omega_j(\mathbf{A}_i \mathbf{y}) d\mathbf{y} \; . \tag{5}$$

Let $\mathcal{D} := \cup_{i=1}^{\ell} \mathcal{D}_i$, where $\mathcal{D} = \{\mathbf{x} \in \mathbb{P}^n | \lambda(\mathbf{x}) \neq 0\}$ is the shape domain corresponding to the whole *template*. Therefore the left hand side of the above equation can be written as

$$\sum_{i=1}^{\ell} \int_{\mathcal{D}_i} \omega_j(\mathbf{x}) d\mathbf{x} = \int_{\bigcup_{i=1}^{\ell} \mathcal{D}_i} \omega_j(\mathbf{x}) d\mathbf{x} = \int_{\mathcal{D}} \omega_j(\mathbf{x}) d\mathbf{x} \; , \tag{6}$$

which can be computed directly from the input image without knowing the partitioning $\mathcal{D}_i$. The resulting system of equations has $\ell n(n+1)$ unknowns:

$$\int_{\mathcal{D}} \omega_j(\mathbf{x}) d\mathbf{x} = \sum_{i=1}^{\ell} |\mathbf{A}_i| \int_{\mathcal{D}'_i} \omega_j(\mathbf{A}_i \mathbf{y}) d\mathbf{y}, \quad j = 1, \ldots, m \; . \tag{7}$$

The solution of the system Eq. (7) provides all the unknown parameters of the overall deformation. Since each ω_j provides one equation, we need $m \geq \ell n(n+1)$ linearly independent functions to solve for ℓ linear transformations. In practice, $m > \ell n(n+1)$ yielding an over-determined system for which a least squares solution is obtained.

3 Numerical Implementation

Theoretically, any nonlinear function satisfying Eq. (3) could be used to construct the system of equations Eq. (7). In practice, however, the solution is obtained via iterative least-squares minimization algorithms, like the *Levenberg-Marquardt algorithm* [17], requiring a carefully chosen numerical scheme.

3.1 Normalization

First of all, the coordinates of both images are normalized into the unit hypercube $[-0.5, 0.5]^n$, *i.e.* $\cup_{i=1}^{\ell} \mathcal{D}_i' \mapsto [-0.5, 0.5]^n$ and $\mathcal{D} \mapsto [-0.5, 0.5]^n$. This is achieved by translating the origin into the center of the mass of the *template* and *observation* followed by an appropriate isotropic scaling with a common factor corresponding to the maximum size of the *template* and *observation.* Of course, the solution of the nonlinear system has to be unnormalized to get the right transformations between the original shapes. Denoting the normalizing transformations of the *template* and *observation* by $\mathbf{N}_t$, $\mathbf{N}_o$, respectively and the solutions by $\widetilde{\mathbf{A}}_i$, the true transformation is thus obtained as $\widehat{\mathbf{A}}_i = \mathbf{N}_t^{-1} \widetilde{\mathbf{A}}_i \mathbf{N}_o$ for all $i = 1, \ldots, \ell$.

Since a least-squares solution involves minimizing the algebraic error of Eq. (7), we expect an equal contribution from each equation in order to guarantee an unbiased error measure. This is achieved by normalizing the range of each ω_j into $[-1, 1]$. We found experimentally, that the transformations occurring during the least-squares minimization process do not transform the shapes out of a hyper-sphere with center in the origin and a radius $\sqrt{n}/2$ (*i.e.* the circumscribed hyper-sphere of the unit hyper-cube). Thus the normalization can be done by dividing the integrals in Eq. (7) with an appropriate constant c_j corresponding to the maximal magnitude of the integral over this domain:

$$c_j = \int_{\|\mathbf{x}\| \leq \frac{\sqrt{n}}{2}} |\omega_j(\mathbf{x})| d\mathbf{x} \ , \quad j = 1, \ldots, m. \tag{8}$$

3.2 Algorithmic Solution and Complexity

In practice, only a limited precision digital image is available, thus the integrals can only be *approximated* by a discrete sum over the foreground pixels introducing an inherent, although negligible error into our computation. The continuous domains $\mathcal{D}$ and $\mathcal{D}_i'$ are represented as finite sets of foreground pixels denoted by D and D_i'. Thus the discrete form of the normalized Eq. (7) becomes

$$\frac{1}{c_j} \sum_{\mathbf{x} \in D} \omega_j(\mathbf{N}_t \mathbf{x}) = \frac{1}{c_j} \sum_{i=1}^{\ell} |\mathbf{A}_i| \sum_{\mathbf{y} \in D_i'} \omega_j(\mathbf{N}_o \mathbf{A}_i \mathbf{y}) \ , \quad j = 1, \ldots, m \ . \tag{9}$$

The system of Eq. (9) is solved by iterative least squares minimization using the *Levenberg-Marquardt algorithm* [17], which requires the evaluation of the equations at every iteration step. Thus the time complexity of the algorithm is considerably decreased if the sums can be precomputed, hence avoiding scanning the image pixels at every iteration. Theoretically, an arbitrary set of ω functions could be used, as long as they generate linearly independent equations. It is shown in [15], however, that choosing a set of polynomial functions will result in a polynomial system of equations, where these sums become precomputed constants. According to these findings the following set of polynomes are adopted

$$\{\omega_j : \mathbb{P}^n \to \mathbb{R}\}_{j=1}^m = \{\mathbf{x} \mapsto x_1^{u_1} \dots x_n^{u_n} | u_k \in \mathbb{N},\ k = 1, \dots, n,\ 0 \leq \sum_{k=1}^n u_k \leq d\} , \quad (10)$$

where d is the maximum degree and the number of the polynomes is given by $m = \frac{1}{n!} \prod_{i=1}^n (d + i)$.

The simple pseudo code of the algorithm is shown in Algorithm 1. Since a set of polynomial functions is applied to generate Eq. (9), the unknowns are eliminated from the sums [15]. Hence the algorithm has a linear time complexity: the complexity of constructing the system Eq. (9) is $\mathcal{O}(|D| + \sum_{i=1}^{\ell} |D_i'|)$; and the complexity of the solver itself is thus independent from the size of the input images.

Algorithm 1. Pseudo-code of the proposed algorithm.

```
Input : The binary template ($D$) and $\ell$ observation shapes ($D_i'$, $i = 1, \dots, \ell$)
Output: $\ell$ estimated linear transformations $\widehat{\mathbf{A}}_i$
1 Normalize the input coordinates by an appropriate similarity transformation $\mathbf{N}$
  into $[-0.5, 0.5]^n$ such that the center of mass becomes the origin.
2 Choose a set of $\omega_j : \mathbb{P}^n \to \mathbb{R}$ ($j = 1, \dots, m \geq \ell n(n+1)$) polynomial functions.
3 Construct the (over-determined) system of equations Eq. (9).
4 Find a least-squares solution of the system using a Levenberg-Marquardt
  algorithm. The solver is initialized with the parameters of the identity
  transformation.
5 Unnormalizing the solutions $\widetilde{\mathbf{A}}_i$ gives the parameters of the aligning
  transformation as $\widehat{\mathbf{A}}_i = \mathbf{N}_t^{-1} \widetilde{\mathbf{A}}_i \mathbf{N}_o$.
```

4 Affine Transformations

Herein we apply the registration framework to important classes of linear deformations: 2D and 3D affine, and 3D rigid body. 2D affine transformations are often used as a linear approximation of projective distortions. 3D rigid body transformation is important in many medical applications. In particular, when bony structures need to be aligned in CT volumes then this transformation should be considered due to the bio-mechanical properties of bones.

4.1 2D Affine Transformations

A 2D affine transformation has 6 parameters, hence $n = 2$ and we have 6ℓ unknowns. In order to obtain sufficiently many equations by using the set of ω functions described in Eq. (10), d has to be chosen such that

$$m = \frac{(d+1)(d+2)}{2} \geq 6\ell \quad \Rightarrow \quad d \geq \left\lceil \frac{\sqrt{1+48\ell}-3}{2} \right\rceil , \tag{11}$$

where $\lceil \cdot \rceil$ denotes the upper integer parts. Eq. (7) becomes for all $j = 1, \dots, m$

$$\int_{\mathcal{D}} x_1^{u_1} x_2^{u_2} d\mathbf{x} = \sum_{i=1}^{\ell} \int_{\mathcal{D}_i'} |\mathbf{A}_i| (a_{i11}y_1 + a_{i12}y_2 + a_{i13})^{u_1} (a_{i21}y_1 + a_{i22}y_2 + a_{i23})^{u_2} d\mathbf{y} \ , \tag{12}$$

where the Jacobian can be easily computed as $|\mathbf{A}_i| = a_{i11}a_{i22} - a_{i12}a_{i21}$.

4.2 3D Affine Transformations

The extension of the 2D case to 3D is rather straightforward. Here, the *template* parts undergo different 3D affine transformations, having a total of 12ℓ unknowns. In this case, d has to be chosen such that

$$m = \frac{(d+1)(d+2)(d+3)}{6} \geq 12\ell \quad \Rightarrow \quad d \geq \left\lceil \frac{c}{3} + \frac{1}{c} - 2 \right\rceil , \text{where}$$

$$c = \sqrt[3]{3\left(324\ell + \sqrt{(324\ell)^2 - 3}\right)}. \tag{13}$$

The Jacobian can be computed as in the 2D case.

4.3 3D Rigid-Body Transformations

An important special case of 3D linear deformations is the rigid-body transformation. This kind of transformations have six degree of freedom: $\alpha_1, \alpha_2, \alpha_3$ are the rotation angles and t_1, t_2, t_3 are the translations along the three coordinate axes. A similar set $\{\omega\}_{j=1}^{m}$ can be used as in Eq. (13), but we need fewer polynomes:

$$d \geq \left\lceil \frac{c}{3} + \frac{1}{c} - 2 \right\rceil , \text{ where } c = \sqrt[3]{3\left(27 + 162\ell + \sqrt{(27+162\ell)^2 - 3}\right)}. \tag{14}$$

Since a rigid-body transformation does not change the size of the objects, the Jacobian determinant equals to 1, hence it is omitted from the equations.

5 Experimental Results

The proposed method has been evaluated on 2D and 3D synthetic datasets. In the case of 2D transformations, the dataset consisted of 10 *template* objects. Synthetic observations were generated by first cutting each object into 2 parts in 4 different ways, resulting in 4 images for each *template*. Then 600 *observations* of size 700×700 were generated by applying randomly composed affine transformations to each of these images with the following parameter ranges: rotation angles of $[-\pi/4; \pi/4]$ and along both axes scaling factors from $[0.75; 1.25]$, skewing from $[-0.1; 0.1]$, and translations of $[-25; 25]$. In the 3D case, 10 *template* volumes were randomly cut into 2 parts by a plane, such that the smaller part is at least 20% of the original volume. By cutting each volume in five different ways, 50 volume images are obtained. Then random 3D affine transformations with similar parameters as in the 2D case (the only difference is that translations were chosen from [-10;10]) have been used to generate a total of 200 3D *observations* of size $250 \times 250 \times 250$.

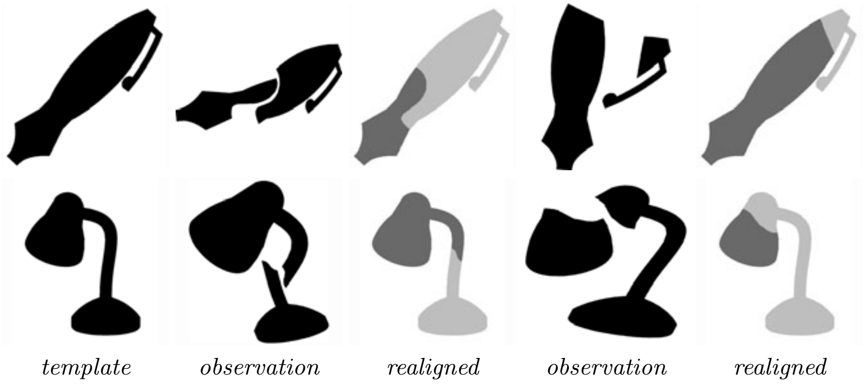


Fig. 1. Sample results on 2D synthetic images

For the evaluation of the results, we defined two kind of error measures: The first one (denoted by ϵ) measures the average distance between the true $\mathbf{A}_i$ and the estimated $\widehat{\mathbf{A}}_i$ transformation for all object. The second one is the absolute difference (denoted by δ) between the *template* and the *aligned* shapes:

$$\epsilon = \sum_{\mathbf{p} \in D'_i, 1 \leq i \leq \ell} \frac{\|(\mathbf{A}_i - \widehat{\mathbf{A}}_i)\mathbf{p}\|}{|D'|}, \quad \text{and} \quad \delta = \frac{|\widehat{D} \triangle D|}{|\widehat{D}| + |D|} \cdot 100\%, \tag{15}$$

where $\triangle$ means the symmetric difference, while $D' = \cup_{i=1}^{\ell} D'_i$ and $\widehat{D} = \cup_{i=1}^{\ell} \widehat{D}_i$ denote the set of pixels of the *observation* and *aligned* shape respectively. Intuitively, ϵ shows the average transformation error per pixel. Note that ϵ can only

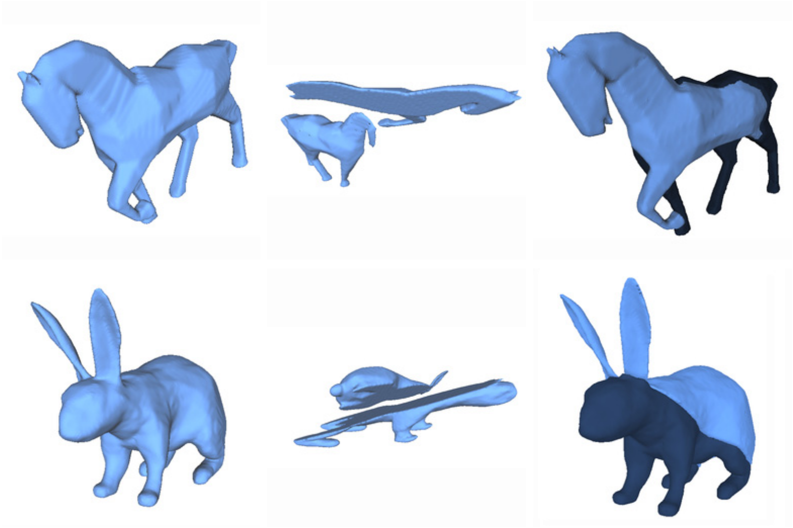


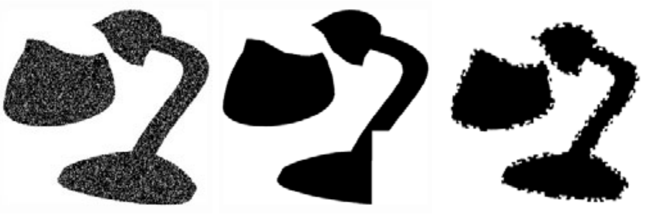
Fig. 2. Sample results on 3D synthetic images

be used when the true transformation is also known, while δ can always be computed. On the other hand, ϵ gives a better characterization of the transformation error as it directly evaluates the mistransformation. As a subjective evaluation measure, we found experimentally that a $\delta \leq 5\%$ in 2D and a $\delta \leq 10\%$ in 3D corresponds to a visually good alignment.

The proposed method was implemented in Matlab and ran under Linux with 3GHz CPU and 3GB memory. The typical runtime was under 3 seconds for 2D and 10 seconds for 3D shapes. Some results are shown in Fig. 1 and Fig. 2. Quantitative results in Table 1 clearly show that the proposed method provides almost perfect alignments in both 2D and 3D.

5.1 Robustness

In practice, segmentation never produces perfect shapes. Therefore, besides using various kind of real images inherently subject to such errors, we have also evaluated the robustness of the proposed approach against different type of segmentation errors. In the first testcase, $5\%, \ldots, 20\%$ of the foreground pixels has been removed from the *observation* before registration. In the second case, we occluded continuous square-shaped regions of size equal to $1\%, \ldots, 10\%$ of the shape. Finally, we randomly added or removed squares uniformly around the boundary of a total size $1\%, \ldots, 10\%$ of the shape. Note that we do not include cases where erroneous foreground regions appear as disconnected regions, because such false regions can be efficiently removed by appropriate morphological filtering. We therefore concentrate on cases where segmentation errors cannot be



(a) missing pixels (b) occlusion (c) boundary error

Fig. 3. Sample observations with various degradations

Table 1. Median of error measures achieved by the proposed method on the 2D and 3D synthetic datasets. The first two rows show the results without degradation while the rest contains the error values vs. various type of segmentation errors as shown in Fig. 3.

Without degradation	ϵ (pixel)				δ (%)			
2D affine transformations	0.11				0.13			
3D affine transformations	0.7				3.09			
(a) missing pixels	1%	5%	10%	20%	1%	5%	10%	20%
2D affine transformations	6.57	21.1	32.83	56.26	2.09	6.24	8.39	12.62
3D affine transformations	1.22	4.65	9.71	19.02	3.99	8.67	15.8	23.54
(b) size of occlusion	1%	2.5%	5%	10%	1%	2.5%	5%	10%
2D affine transformations	9.91	20.45	35.04	58.68	3.54	6.35	9.51	13.75
3D affine transformations	3.27	7.7	14.73	22.74	8.07	13.08	18.47	26.13
(c) size of boundary error	1%	2.5%	5%	10%	1%	2.5%	5%	10%
2D affine transformations	1.9	3.91	6.65	12.23	0.59	1	1.73	3.08
3D affine transformations	0.99	1.44	2.33	4.03	3.23	3.65	4.44	5.8

filtered out. See samples of these errors in Fig. 3. Table 1 shows that our method is quite robust whenever errors are uniformly distributed over the whole shape (first and third testcases). However, it becomes less stable in case of larger localized errors, like occlusion and disocclusion. This is a usual behavior of area-based methods because they are relying on quantities obtained by integrating over the object area. Thus large missing parts would drastically change these quantities resulting in false alignments. Nevertheless, in many application areas one can take images under controlled conditions which guarantees that observations are not occluded (*e.g.* medical imaging, industrial inspection).

5.2 Solving the Tangram Puzzle

Tangram is a dissection puzzle consisting of seven flat tiles (called *tans*), which are put together to form various shapes. The objective is to form a specific shape given only by its silhouette. Fig. 4 shows some examples of these shapes and the solutions found by our method.

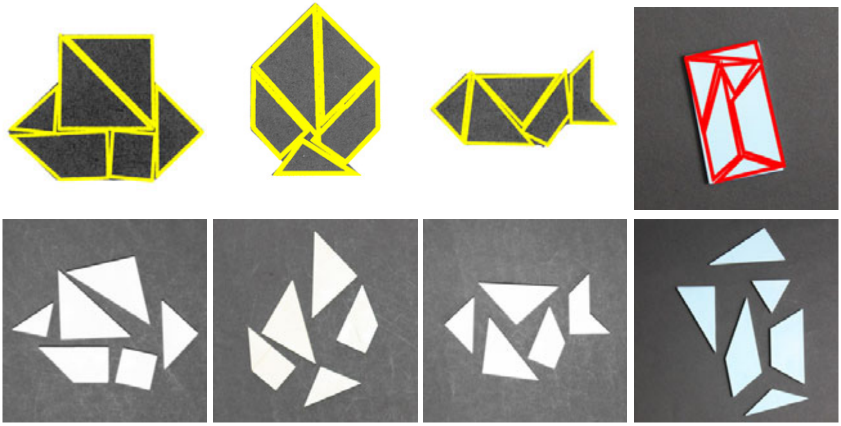


Fig. 4. Solutions of the Tangram puzzle. **Top:** Template images with overlayed contours of aligned fragments. **Bottom:** Observations.

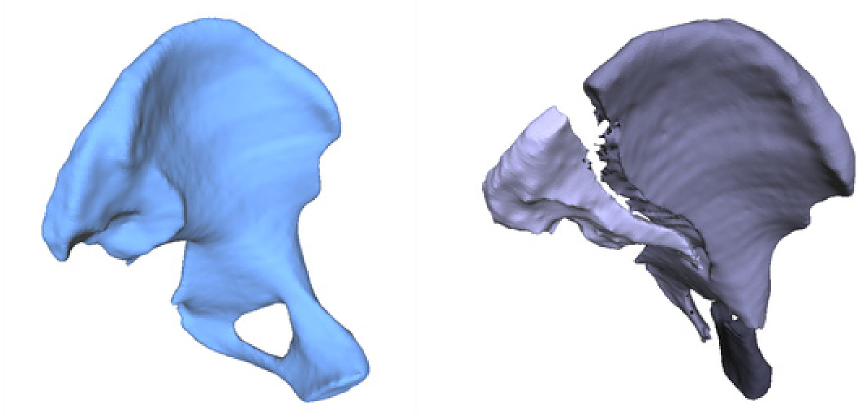
The images were taken with a digital camera, then they were thresholded and the resulting 2D shapes were realigned according to the *template*. The first three *templates* of Fig. 4 are more challenging as they are scanned versions of the printed shapes found in the Tangram manual, which are only approximate silhouettes of the final tile configurations. We have used the affine model as an approximation of the actual plane projective transformation acting between the shapes.

It is well known that the *Levenberg-Marquardt* algorithm finds a local minima close to the initialization. Finding a good initial configuration is largely application-dependent. For example, on these images a global optimization procedure (*e.g.* Spectral Gradient Method [18]) provided good initialization, from which *Levenberg-Marquardt* gives a better solution than starting from the identity transform.

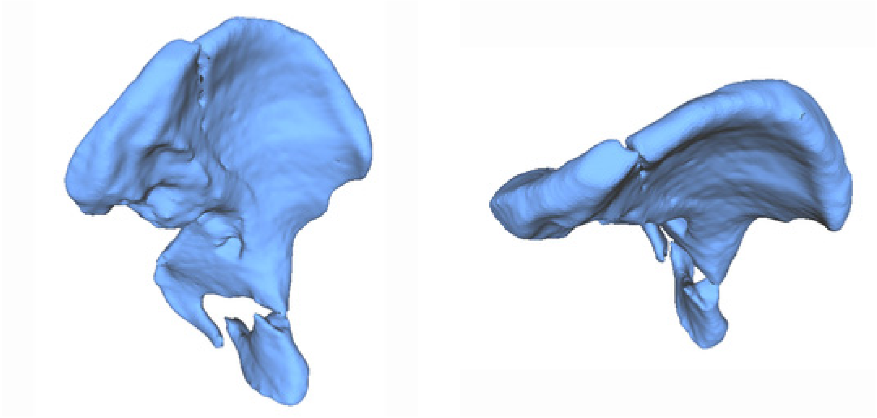
Finally, we note that some tiles are slightly overlapping in Fig. 4. This is because overlaps are invisible for the system of equations. Nevertheless, overlaps could be prevented by checking the transformed fragments at every iterations, but this is a rather time consuming procedure.

5.3 Realigning Bone Fractures

Complex bone fracture reduction frequently requires surgical care, especially when angulation or displacement of bone fragments are large. In such situations, computer aided surgical planning [5] is done before the actual surgery takes place, which allows to gather more information about the dislocation of the fragments and to arrange and analyze the surgical implants to be inserted. A crucial part of such a system is the relocation of bone fragments to their original anatomic position. Since the input data is typically a volume CT image, this



Template obtained by mirroring intact bones Observation



Realigned bone fragments

Fig. 5. Bone fracture reduction (CPU time was 15 sec. for these 1 megavoxel CT volumes)

repositioning has to be performed in 3D space which requires an expensive special 3D haptic device and quite a lot of manual work. Therefore automatic bone fracture reduction can save considerable time, providing experts with a rough alignment which can be manually fine-tuned according to anatomic requirements.

Since surgical planning involves the biomechanical analysis of the bone with implants, only rigid-body transformations are allowed. In [5], a classical ICP algorithm is used to realign fractures. Winkelbach *et al.* [6] proposed an approach for estimating the relative transformations between fragments of a broken cylindrical structure by using well known surface registration techniques, like 2D depth correlation and the ICP algorithm. In [7], registration is solved by using quadrature filter phase difference to estimate local displacements.

Herein, we apply our puzzle framework to reduce pelvic fractures using 3D rigid-body transformations. In cases of single side fractures, the *template* is simply obtained by mirroring intact bones of the patient. Fig. 5 shows a typical result for a pelvic fracture with three fragments. The main challenges are segmentation errors and, due to the variability of the human body, a slightly different *template*. In spite of these difficulties, the alignment of larger parts is quite accurate, only the small fragment has a noticeable alignment error. Since the error caused by a misplaced small piece is relatively low, the solver may not find the best transformation. If we could normalize the terms of Eq. (9) corresponding to each fragment, then the algebraic error would be better balanced and a precise alignment could be found. Unfortunately, this is impossible as we should know the partitioning of the *template* to compute proper normalizing constants. Since human verification and correction of the result is needed anyway in a real surgical planning system, these small errors are not critical and can be easily corrected.

6 Conclusion

A novel framework to solve the affine puzzle problem has been proposed and applied to 2D and 3D affine transformations. As opposed to classical solutions based on landmark extraction and correspondences, the proposed solution finds the aligning transformations without any additional information. Basically, the method consists in constructing a polynomial system of equations whose solution directly provides the unknown parameters. Obviously, the number of object fragments and strength of the deformation may influence the quality of the alignment: The more parts we have the more equations are required, which affects numerical stability. Furthermore, more parts allow more affine transformations yielding a stronger deformation. A completely random fragment-configuration corresponds to a complex deformation, for which a stable solution is difficult to achieve. On the other hand, when pieces are in relative order then a rather accurate solution is obtained. Note, that for the presented medical application, this is a realistic assumption due to physical constraints. Quantitative evaluations on both 2D and 3D synthetic datasets demonstrate the performance and robustness of the method and results obtained on real images confirm its relevance in various application domains.

References

1. Kong, W., Kimia, B.B.: On solving 2D and 3D puzzles using curve matching. In: Proceedings of the International Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii, vol. 2, pp. 1–8. IEEE, Los Alamitos (2001)
2. Papaioannou, G., Karabassi, E.A., Theoharis, T.: Reconstruction of three-dimensional objects through matching of their parts. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 114–124 (2002)
3. McBride, J.C., Kimia, B.B.: Archaeological fragment reconstruction using curve-matching. In: Proceedings of the International Conference on Computer Vision and Pattern Recognition Workshop, Madison, Wisconsin, USA, pp. 1–8. IEEE, Los Alamitos (2003)

4. Hill, D.L.G., Batchelor, P.G., Holden, M., Hawkes, D.J.: Medical image registration. Physics in Medicine and Biology 45, R1–R45 (2001)
5. Erdőhelyi, B., Varga, E.: Semi-automatic bone fracture reduction in surgical planning. In: Proceedings of the International Conference on Computer Assisted Radiology and Surgery. International Journal of Computer Assisted Radiology and Surgery, vol. 4, pp. 98–99. Springer, Berlin (2009)
6. Winkelbach, S., Westphal, R., Goesling, T.: Pose estimation of cylindrical fragments for semi-automatic bone fracture reduction. In: Michaelis, B., Krell, G. (eds.) DAGM 2003. LNCS, vol. 2781, pp. 556–573. Springer, Heidelberg (2003)
7. Pettersson, J., Knutsson, H., Borga, M.: Non-rigid registration for automatic fracture segmentation. In: Proceedings of International Conference on Image Processing, Atlanta, GA, USA, pp. 1185–1188. IEEE, Los Alamitos (2006)
8. Bronstein, A.M., Bronstein, M.M.: Regularized partial matching of rigid shapes. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 143–154. Springer, Heidelberg (2008)
9. Funkhouser, T., Shilane, P.: Partial matching of 3D shapes with priority-driven search. In: Proceedings of the Eurographics Symposium on Geometry Processing, Sardinia, Italy, Eurographics, ACM SIGGRAPH, pp. 1–12 (2006)
10. Reuter, M.: Hierarchical shape segmentation and registration via topological features of Laplace-Beltrami eigenfunctions. International Journal of Computer Vision 89, 287–308 (2010)
11. Rustamov, R.M.: Laplace-Beltrami eigenfunctions for deformation invariant shape representation. In: Proceedings of the Eurographics Symposium on Geometry Processing, Barcelona, Spain, Eurographics, ACM SIGGRAPH, pp. 1–9 (2007)
12. Besl, P.J., McKay, N.D.: A method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 14, 239–256 (1992)
13. Periaswamy, S., Farid, H.: Medical image registration with partial data. Medical Image Analysis 10, 452–464 (2006)
14. Feldmar, J., Ayache, N.: Rigid, affine and locally affine registration of free-form surfaces. International Journal of Computer Vision 18, 99–119 (1996)
15. Domokos, C., Kato, Z.: Parametric estimation of affine deformations of planar shapes. Pattern Recognition 43, 569–578 (2010)
16. Hagege, R.R., Francos, J.M.: Estimation of affine geometric transformations of several objects from global measurements. In: Proceedings of IEEE International Workshop on Multimedia Signal Processing, Rio de Janeiro, Brazil, pp. 1–5. IEEE, Los Alamitos (2009)
17. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. SIAM Journal on Applied Mathematics 11, 431–441 (1963)
18. Birgin, E.G., Martínes, J.M., Raydan, M.: SPG: Software for convex-constrained optimization. ACM Transactions on Mathematical Software 27, 340–349 (2001)