Int J Comput Vis (2012) 97:54–70
DOI 10.1007/s11263-011-0452-0

Feature-Based Deformable Surface Detection with Self-Occlusion Reasoning

Daniel Pizarro · Adrien Bartoli

Received: 25 September 2010 / Accepted: 11 April 2011 / Published online: 27 April 2011

Abstract This paper presents a method for detecting a textured deformed surface in an image. It uses (wide-baseline) point matches between a template and the input image. The main contribution of the paper is twofold. First, we propose a robust method based on local surface smoothness capable of discarding outliers from the set of point matches. Our method handles large proportions of outliers (beyond 70% with less than 15% of false positives) even when the surface self-occludes. Second, we propose a method to estimate a self-occlusion resistant warp from point matches. Our method allows us to realistically retexture the input image. A pixel-based (direct) registration approach is also proposed. Bootstrapped by our robust point-based method, it finely tunes the warp parameters using the value (intensity or color) of all the visible surface pixels. The proposed framework was tested with simulated and real data. Convincing results are shown for the detection and retexturing of deformed surfaces in challenging images.

Keywords Deformable surfaces · Image registration · Feature-based registration · Pixel-based refinement · Robust feature correspondence

1 Introduction

Detecting a textured deformed surface in a single image is an important fundamental and applied problem for video augmentation, Non-Rigid Structure-from-Motion, material deformation analysis, to name just a few. Detection is here to be understood as the ability to register an input image of the deformed surface to a template image. The template image is known and represents the surface of interest in a canonical shape, usually flat. Figure 1 shows an example of a template and an input image that we will use throughout the paper to illustrate various concepts. Compared to the registration and detection of rigid objects, for which robust, fast and mature methods such as Baker and Matthews (2004) were developed, the detection of deformable objects still lags behind, even though it received a growing attention over the past few years (Bartoli and Zisserman 2004; Chui and Rangarajan 2003; DeCarlo and Metaxas 1998; Gay-Bellile et al. 2010; Hilsmann and Eisert 2009; Pilet et al. 2008).

There are two kinds of approaches in the literature for the registration of deformable surfaces: the pixel-based (direct) approach, where the intensity discrepancy between the images is used as a cue to compute the deformation, and the feature-based approach, whereby features are detected, matched, and used to compute the deformation.

The pixel-based approach has lead to many effective registration methods for different kinds of deformable objects, such as human faces (DeCarlo and Metaxas 1998) or deformable surfaces (Bartoli and Zisserman 2004; Gay-Bellile et al. 2010; Hilsmann and Eisert 2009; Pilet et al. 2008). In Gay-Bellile et al. (2010) a robust and self-occlusion resistant registration method was presented for surfaces, while in Pizarro et al. (2008) a light-invariant color discrepancy measure was used to attenuate the effect of illumination change. However, pixel-based methods have limitations: being iterative and 'local' they generally heavily rely on the initialization. They thus are mostly used in off-line video processing since they cannot self-recover after failure. On the other

D. Pizarro (✉)
University of Alcala, Alcala de Henares, Spain
e-mail: pizarro@depeca.uah.es

A. Bartoli
Clermont Université, Clermont-Ferrand, France
e-mail: adrien.bartoli@gmail.com

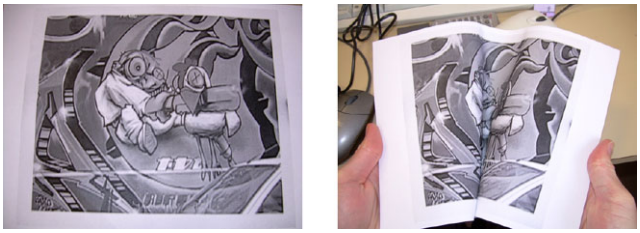


Fig. 1 Example of a template (*left*) and an input image (*right*) with deformations and a self-occlusion (from Bartoli 2008). This example is used to illustrate the different steps and concepts of our methods

hand, they can be bootstrapped, typically by a feature-based method.

In the feature-based approach, the registration is computed by minimizing the distance between matched features in the template and the input image. These methods do not generally need an initialization. However, their use has two main challenges. The first one is to detect erroneous matches (that any automatic feature matching method may make.) The second challenge is to handle surface self-occlusions since very few features are usually detected near the self-occlusion boundary. The first challenge has received attention in the literature. An iterative robust real-time feature-based surface detection method based on an M-estimator was proposed in Pilet et al. (2008). This method shows excellent capabilities for removing outliers from wide-baseline matches but, as it uses the global smoothness of the motion field, it is defeated by surface self-occlusions which are naturally 'unsmooth' along one direction. Some other methods such as Chui and Rangarajan (2003) were specifically designed to match points across soft deformations. They are generally slow and do not cope with self-occlusions either.

The state of the art is missing a deformed surface detection method that like Pilet et al. (2008) would be able to both handle wide-baseline and like Gay-Bellile et al. (2010) would be able to cope with self-occlusions. The keypoint-based method we propose in this paper satisfies these requirements. Our key idea to cope with self-occlusions is to model the motion field as being piecewise smooth instead of globally smooth as in previous work. We use local smoothness to filter out erroneous matches. A novel self-occlusion resistant warp estimator is then proposed, that estimates a piecewise smooth warp, as required. In more details, our contributions are:

- A method that detects and discard outliers (erroneous point matches) based on smoothness on the local scale only. Our method copes with large amounts of outliers, beyond 80% in our experiments.
- A self-occlusion detection method, that finds which parts of the template are self-occluded in the input image. Our method forms a partial self-occlusion map, containing the 'important' sub-part of the self-occlusion for subsequent warp estimation.
- A point-based warp estimation method that prevents the warp to fold in the presence of self-occlusions. This method combines a data term, a smoother and uses the partial self-occlusion map in a single round of convex, linear least squares optimization. Contrarily to previous work which discards warp smoothing in the orthogonal direction of the self-occlusion boundary, we force the warp to behave in a rigid affine manner over the self-occluded area. This leads to a simple modification of the usual bending energy term.
- A pixel-based warp estimation method that, bootstrapped by the point-based registration result, finely tunes the warp parameters. Our modified bending energy term is here employed directly in place of the usual one.

Note that parts of our method were published in a shorter conference version of this paper (Pizarro and Bartoli 2010).

This paper is organized as follows. In Sect. 2 the problem statement and background are presented. Our outlier rejection framework is then given in Sect. 3 followed by the self-occlusion resistant warp estimation in Sect. 4. A pixel-based registration step is proposed in Sect. 5 to refine the warp parameters. In Sect. 6, some results of automatic surface detection and retexturing are given. Conclusions are drawn in Sect. 7.

2 Problem Statement and Background

We first describe the inputs and hypotheses our algorithm uses, and then a generic image deformation model or *warp*, followed by a simple linear least squares point-based estimation method.

2.1 Inputs and Hypotheses

We assume that point matches were established between the template image $\mathcal{T}$ and an input image $\mathcal{I}$ ($\mathcal{T}(\mathbf{p})$ with $\mathbf{p} = (x \quad y)^\top$ and $\mathcal{I}(\mathbf{q})$ with $\mathbf{q} = (u \quad v)^\top$ are pixel colors). The template shows the surface unraveled (flat in most cases). The surface deformation and viewpoint change may be large in the input image. The surface may also undergo external occlusions and self-occlusions. There exist several keypoint detectors and descriptors such as SIFT (Lowe 2004) and SURF (Bay et al. 2008) for which a matching procedure will, despite the significant changes in appearance between the template and the input images, find matches. In Lepetit and Fua (2006) the authors proposed a wide-baseline feature matching method based on artificially synthesized texture exemplars for keypoints (this method was successfully used in Pilet et al. (2008) for deformable surface detection.) These methods usually produce a fair amount of correct matches or *inliers*, but the matches almost always also include erroneous matches or *outliers*.

The keypoints we use in this paper are detected and matched with the SURF method as Fig. 2 illustrates. Note

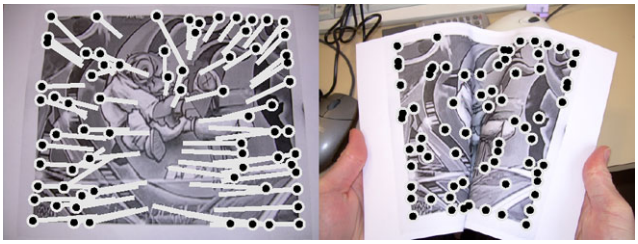


Fig. 2 Point matches between the template and the input images and motion vectors superimposed on the template image. These $N = 71$ point matches were obtained with the SURF method (Bay et al. 2008)

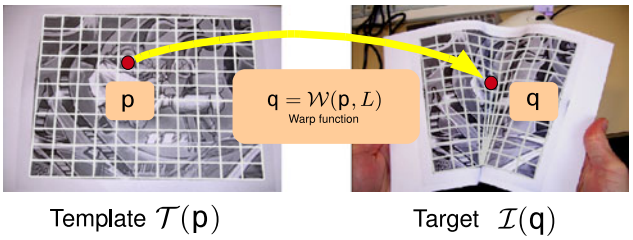


Fig. 3 Warp transfer of a template point $\mathbf{p}$ to a point $\mathbf{q}$ in the input image. Both images are overlaid with a *warp visualization grid*. This grid is fixed in the template and transformed to the input image to let one visualize the warp's behaviour

that our surface detection method is however independent of the type of keypoint detector being used. This provides two matched sets of N points each, denoted $C_\mathbf{p}$ and $C_\mathbf{q}$, which may contain outliers:

$$C_\mathbf{p} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}, \quad \mathbf{p}_i = (x_i \; y_i)^\top, \tag{1}$$

$$C_\mathbf{q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_N\}, \quad \mathbf{q}_i = (u_i \; v_i)^\top. \tag{2}$$

Point $\mathbf{p}_i$ in $\mathcal{T}$ is matched with point $\mathbf{q}_i$ in $\mathcal{I}$.

Our aim is to find the inlier subsets $G_\mathbf{p} \subset C_\mathbf{p}$ and $G_\mathbf{q} \subset C_\mathbf{q}$ of unknown size n. In addition, we search for the global warp function $\mathcal{W}(\mathbf{p}, L)$ parameterized by the matrix L, that minimizes a distance criterion between every point in $\mathbf{p}_i \in G_\mathbf{p}$ and its match $\mathbf{q}_i \in G_\mathbf{q}$. Our basic assumption is that, given the correct parameter vector, $\mathcal{W}$ maps any point in $\mathcal{T}$ to its match in $\mathcal{I}$, as Fig. 3 illustrates.

2.2 Image Deformation Model

The image deformation model we use is a parametric warp. There is a large variety of such functions in the literature. Popular warps are those based on Radial Basis Functions (*e.g.* the Thin-Plate Spline (TPS) (Bookstein 1989)) and those based on the tensor-product, called Free-Form Deformations (FFD) (*e.g.* using the cubic B-spline (Rueckert et al. 1999)). Our algorithm is not specific to a particular type of warps, but in practice, the different steps of our algorithm use the two types of warps mentioned directly above.

In order to give a general framework (that includes the FFD and RBF warps amongst others) we use the generic model of Bartoli (2008). Let $\mathbf{p} \in \mathbb{R}^2$ be a point coordinate vector in the template image. The warp $\mathcal{W} : \mathbb{R}^2 \times \mathbb{R}^{l\times 2} \to \mathbb{R}^2$ maps 2D points from the template to the input image and depends on a set of l 2D control points $\mathbf{c}_1, \ldots, \mathbf{c}_l$ stacked in the parameter matrix $L \in \mathbb{R}^{l\times 2}$. The position of these control points specifies the behavior of the warp. The general parametric warp is defined as:

$$\mathcal{W}(\mathbf{p}, L) = L^\top \nu(\mathbf{p}), \tag{3}$$

with $\nu : \mathbb{R}^2 \to \mathbb{R}^l$ some nonlinear lifting function which, dotted with L, gives the warp's value. For a detailed explanation of the ν function for the TPS, see Bartoli (2008), where a feature-driven parameterization of the TPS is developed. In the FFD case we consider a regular grid of $l = m \times m$ control points, covering a specific area of the template image, and a modified grid of control points $\mathbf{s}_{i,j} = \mathbf{c}_{i+m(j-1)}$ stacked inside L. The warp function is:

$$\mathcal{W}_{\text{FFD}}(\mathbf{p}, L) = \sum_{k=0}^{3}\sum_{l=0}^{3} \mathcal{B}_k(v)\mathcal{B}_l(w)\mathbf{s}_{i+k,j+l}, \tag{4}$$

where $\mathcal{B}_k$, $\mathcal{B}_l$ are cubic B-spline interpolation coefficients evaluated at the normalized coordinates $(v \quad w)^\top$ of point $\mathbf{p}$ (expressed with respect to the 16 closest control points). The lifting function $\nu(\mathbf{p})$ is thus composed of the B-Spline coefficients for each point $\mathbf{p}$. The FFD has compact support by definition as the warped coordinates of every point is always driven by the position of only 16 control points.

2.3 Linear Least Squares Warp Estimation

The parameter vector L is chosen so as to minimize a cost function ϵ, composed of a data term ϵ_d, based on the average distance between the warped points in $G_\mathbf{p}$ and the matched points in $G_\mathbf{q}$, and a smoothing term ϵ_s that controls the smoothness of the motion field. The data term is thus a sum of squared residuals (MSR):

$$\epsilon_d(L) = \frac{1}{n}\sum_{i=1}^{n} \|\mathcal{W}(\mathbf{p}_i, L) - \mathbf{q}_i\|^2. \tag{5}$$

This can be rewritten in matrix form as:

$$\epsilon_d(L) = \frac{1}{n}\|AL - \Phi\|_F^2, \tag{6}$$

where $\|\cdot\|_F$ is the matrix Frobenius norm and:

$$A^\top = \begin{pmatrix} \nu(\mathbf{p}_1) & \cdots & \nu(\mathbf{p}_n) \end{pmatrix},$$

and:

$$\Phi^\top = \begin{pmatrix} \mathbf{q}_1 & \cdots & \mathbf{q}_n \end{pmatrix}.$$

We use the squared integral bending energy (approximated by the second derivatives of the warp), simply dubbed the *bending energy* as smoothing term[1] ϵ_s:

$$\epsilon_s(L) = \int_{\mathbb{R}^2} \left\| \frac{\partial^2 \mathcal{W}}{\partial \mathbf{p}^2}(\mathbf{p}, L) \right\|_F^2 d\mathbf{p}. \tag{7}$$

For the TPS the bending energy has a closed-form (Duchon 1976). For the other kinds of warps, the integral in (7) can be approximated by a Riemann sum and finite differences:

$$\epsilon_s(L) \approx \sum_{\mathbf{p}} \left\| \frac{\partial^2 \mathcal{W}}{\partial \mathbf{p}^2}(\mathbf{p}) \right\|_F^2 \Delta x \Delta y. \tag{8}$$

In all cases ϵ_s can be written as a quadratic function of L using a constant matrix Z:

$$\epsilon_s(L) = \|ZL\|_F^2. \tag{9}$$

Assembling the two terms with a smoothing weight μ to form the cost function gives:

$$\epsilon(L) = \frac{1}{n}\|AL - \Phi\|_F^2 + \mu^2 \|ZL\|_F^2. \tag{10}$$

Fixing the smoothing weight μ the optimal warp parameters L are given in closed-form as:

$$L = \left(A^\top A + n\mu^2 Z^\top Z\right)^{-1} A^\top \Phi = T\Phi. \tag{11}$$

The *influence matrix* T transforms the input image point coordinates in Φ in the warp parameter matrix L.

3 Outlier Rejection

We first describe our method to use piecewise smoothness to detect and reject outliers from the point matches. We then give an analysis of the method's computational complexity and a set of experiments that allows us to choose the method's parameters.

3.1 Proposed Algorithm

We propose to remove outliers from the set of matches $C_\mathbf{p} \leftrightarrow C_\mathbf{q}$ by assuming that the surface is locally smooth and that its local topology must thus be preserved.

Based on that assumption, we first form, for the set of template image points $C_\mathbf{p}$, a Delaunay triangulation $T = D(C_\mathbf{p})$. For each $\mathbf{p}_i \in C_\mathbf{p}$ there is a unique set, namely $Q^T(\mathbf{p}_i) = (\mathbf{p}_{i_1}, \ldots, \mathbf{p}_{i_{N(i)}})$, of $N(i)$ neighboring points induced by the triangulation T. The basic idea of our algorithm is to exploit that, if the surface is locally smooth, the set Q^T contains information about the position of $\mathbf{p}_i$ in the template, given also $\mathbf{q}_i$ and its neighbors (induced by T in the input image) denoted $Q^T(\mathbf{q}_i) = (\mathbf{q}_{i_1}, \ldots, \mathbf{q}_{i_{N(i)}})$.

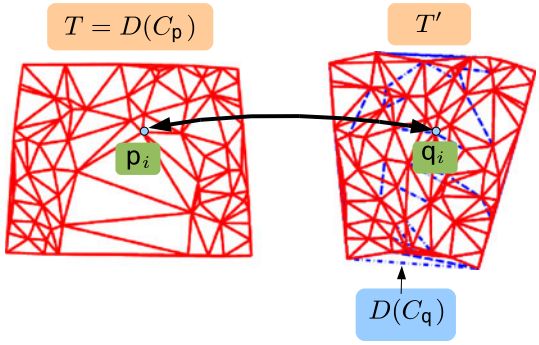


Fig. 4 The Delaunay triangulation $T = D(C_\mathbf{p})$ in the template is displayed as the left mesh in *solid red*. The induced triangulation T' in the input image is shown in *solid red* in the *right part* of the figure. Using *dashed blue lines* we show the differences between T' and $D(C_\mathbf{q})$

We denote T' the triangulation defined in the set $C_\mathbf{q}$ using the edges in T and the matching $C_\mathbf{p} \leftrightarrow C_\mathbf{q}$ to define the input image vertices.[2] It is interesting to note that, even in the absence of outliers, the triangulations T' and $D(C_\mathbf{q})$ do not necessarily coincide. This is due to surface deformation and viewpoint change (see Fig. 4 for an illustration.) We always take T as the reference triangulation to define the set of neighbors. For simplicity, we drop the reference to T from the neighbor sets of $\mathbf{p}_i$ and $\mathbf{q}_i$, simply writing them $Q(\mathbf{p}_i)$ and $Q(\mathbf{q}_i)$.

We propose to build a measure d_i in the template image of how near point $\mathbf{p}_i$ is to an estimate $\hat{\mathbf{p}}_i$ obtained from the sets $Q(\mathbf{p}_i)$ and $Q(\mathbf{q}_i)$ and point $\mathbf{q}_i$:

$$\hat{\mathbf{p}}_i = f(\mathbf{q}_i, Q(\mathbf{p}_i), Q(\mathbf{q}_i)). \tag{12}$$

To formulate this relationship we use a general warp function $\mathcal{W}$ depending on a set of parameters L:

$$\hat{\mathbf{p}}_i = \mathcal{W}(\mathbf{q}_i, L), \tag{13}$$

where L is obtained using the $N(i)$ matches in $Q(\mathbf{p}_i) \leftrightarrow Q(\mathbf{q}_i)$ and the influence matrix (11).

As will be seen in the experimental results that we report in Sect. 3.3 the choice of a deformable model for function f outperforms other choices such as a rigid projective transformation. We use the Euclidean distance $d_i = \|\mathbf{p}_i - \hat{\mathbf{p}}_i\|$ as a decision variable to know how likely the match $\mathbf{p}_i \leftrightarrow \mathbf{q}_i$ is to be an inlier. An illustration is given in Fig. 5.

Our statement is the following: we consider $\mathbf{p}_i \leftrightarrow \mathbf{q}_i$ as a probable inlier if $d_i < d_{TH}$ for some chosen threshold d_{TH}. However it must be noted that the reverse implication does

[1] Each term in the integral is the norm of a valence-3 tensor.

[2] In other words, T' is obtained by replacing every point $\mathbf{p}_i$ by $\mathbf{q}_i$ in T and keeping the edges.

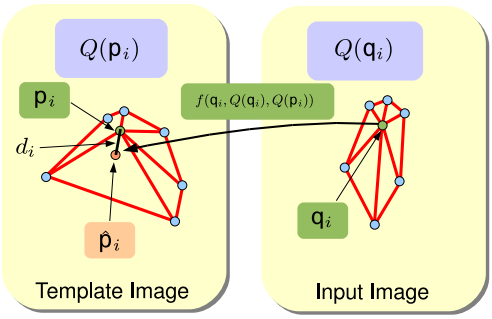


Fig. 5 Computation of the distance d_i. A low value for d_i implies that $\mathbf{p}_i \leftrightarrow \mathbf{q}_i$ is an inlier. The opposite is not always true, as we explain in the main text

not hold: if there were outliers in the set $Q(\mathbf{p}_i) \leftrightarrow Q(\mathbf{q}_i)$, the distance d_i would be affected, without giving a cue about the correctness of match $\mathbf{p}_i \leftrightarrow \mathbf{q}_i$. The statement implicitly considers that the probability that erroneous neighboring matches produce a small d_i is very small, given that $\mathbf{p}_i \leftrightarrow \mathbf{q}_i$ is an outlier, which in practice is very reasonable.

Choosing the template image coordinates to evaluate the distance d_i is reasonable as we usually select the template to show the surface flat and fronto-parallel to the camera. In addition, the scale of the surface in the template image is known, which allows us to learn the threshold d_{TH} from simulated data (see Sect. 3.3). If the input image were used to compute d_i the threshold d_{TH} would have to changed accordingly to the arbitrary and unknown surface scale and deformation.

In order to tell apart inliers and outliers using d_i, we first compute an initial set of highly confident, strong inliers, which have a small d_i value that will give the subsets $G_\mathbf{p} \subset C_\mathbf{p}$ and $G_\mathbf{q} \subset C_\mathbf{q}$. The other matches are grouped in the complementary sets $B_\mathbf{p} = C_\mathbf{p} - G_\mathbf{p}$ and $B_\mathbf{q} = C_\mathbf{q} - G_\mathbf{q}$. We then use a second triangulation $\tilde{T} = D(G_\mathbf{p})$ using only the inliers in $G_\mathbf{p}$, which will be used to identify if the complementary sets $B_\mathbf{p} \leftrightarrow B_\mathbf{q}$ contain inliers surrounded by outliers. The test criterion is based on the distance d_i computed using the new triangulation. Each newly identified inlier $\mathbf{p}_k \leftrightarrow \mathbf{q}_k$ is then introduced in the sets $G_\mathbf{p} = \{G_\mathbf{p}, \mathbf{p}_k\}$ and $G_\mathbf{q} = \{G_\mathbf{q}, \mathbf{q}_k\}$. Once the test has been applied to the set of candidate inliers, the sets $G_\mathbf{p}$ and $G_\mathbf{q}$ contain the matches we eventually label as inliers.

Our algorithm can be summarized with the following steps:

1. Compute a Delaunay triangulation $T = D(C_\mathbf{p})$.
2. Compute d_i for all matches $i = 1, \ldots, N$ in $C_\mathbf{p} \leftrightarrow C_\mathbf{q}$.
3. Mark a match $\mathbf{p}_i \leftrightarrow \mathbf{q}_i$ as an inlier if $d_i < d_{TH}$. The set of strong inliers is denoted $G_\mathbf{p} \leftrightarrow G_\mathbf{q}$.
4. Compute a new Delaunay triangulation $\tilde{T} = D(G_\mathbf{p})$.
5. For each match $\mathbf{p}_k \leftrightarrow \mathbf{q}_k$ in $B_\mathbf{p}$ and $B_\mathbf{q}$:
   (a) Update the triangulation $\tilde{T}$ to $\tilde{T}^* = D(\{G_\mathbf{p}, \mathbf{p}_k\})$ and select the set of neighbors $Q(\mathbf{p}_k)$ and $Q(\mathbf{q}_k)$.
   (b) Compute d_k using the neighbor sets.
   (c) If $d_k < d_{TH}$, update the inlier sets $G_\mathbf{p} = \{G_\mathbf{p}, \mathbf{p}_k\}$ and $G_\mathbf{q} = \{G_\mathbf{q}, \mathbf{q}_k\}$.
6. Loop to step 4 until $G_\mathbf{p}$ and $G_\mathbf{q}$ stop growing.

Table 1 Computational complexity of our outlier rejection algorithm

Step	Content	Complexity
1	Compute $T = D(C_\mathbf{p})$	$\mathcal{O}(N \log N)$
2–3	Compute d_i, $G_\mathbf{p}$, $G_\mathbf{q}$	$\mathcal{O}(N)$
4	Compute $\tilde{T} = D(G_\mathbf{p})$	$\mathcal{O}(N_i \log N_i)$
5–6	Update $G_\mathbf{p}$ and $G_\mathbf{q}$	$\mathcal{O}(N_o \log N_i)$

3.2 Computational Complexity

The proposed method relies on the computation and refinement of Delaunay triangulations. The computational complexity for obtaining the Delaunay triangulation of a randomly distributed set of N points is bounded by $\mathcal{O}(N \log N)$. However the complexity of updating the triangulation by adding or removing a new point is $\mathcal{O}(\log N)$ (Lischinski 1994). Let N_i and N_o be the number of (strong) inliers and outliers inside the initial set of matches respectively. It can be shown that uniformly distributed sets of points have on average 6 neighbors in a Delaunay triangulation. We thus suppose that computing the set of distances d_i, $i = 1, \ldots, N$ can be done in $\mathcal{O}(N)$. An efficient way of computing d_i for the TPS warp is given in the Appendix. The computational complexity of each step of our algorithm is given in Table 1.

We assume that steps 4 and 5 involves to check only N_o points, as in most cases the majority of inliers is obtained from the previous steps. The number of times step 5 must be repeated is at most 3 in our simulations so it is constant and does not affect the asymptotic complexity. The overall algorithm complexity is thus $\mathcal{O}(N \log N)$.

3.3 Learning the Value of the Parameters

The proposed method has several parameters such as the threshold d_{TH} and the warp function chosen to compute the distance d_i. We report experimental results on simulated data that allow us to learn these parameters. We are interested in checking the performance of both rigid (projective) and deformable (*i.e.* TPS) warp functions as candidates to be used in function f from (12). It is expected that when the distance between keypoints is small enough, the rigid model works fine, as the surface can be locally approximated by a plane.

Our evaluation method is as follows. Given a template image showing the surface flat and fronto-parallel, we choose a set of four input images, showing different surface

deformations. By manually selecting a set of point matches between the template and each of the test input images, four different surfaces were generated using an FFD warp with $l = 12 \times 12$ control points and $\mu = 0.55$ (see Fig. 6). The test surfaces include two self-occlusions and two globally smooth deformations. By generating points randomly in the template image and obtaining the warped coordinates on each deformed surface it is possible to generate an arbitrary number of point matches for testing the proposed algorithm. To artificially generate outliers, we corrupt an arbitrary percentage of the generated matches with a uniform random distribution.

Figures 7a, 7b and 7c show the ROC (Receiver Operating Characteristic) curves of the experiments for d_{TH} and against $N = \{49, 100, 225\}$ matches. In the three cases the same experiment is shown for 30% and 50% outliers. By controlling the number of matches we implicitly change the distance between points, which affects the expected performance in the different possibilities for the warp. On each ROC curve one may see the performance of the following versions of the warp function f:

– 'Projective': a 2D homography.
– 'TPS 3×3': a TPS warp with $l = 3 \times 3$ centers.
– 'TPS 5×5': a TPS warp with $l = 5 \times 5$ centers.

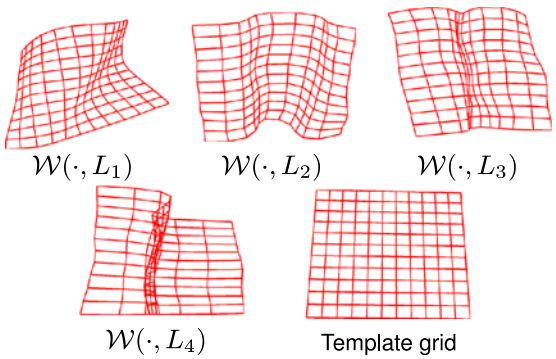


Fig. 6 Different shapes used in the evaluation tests

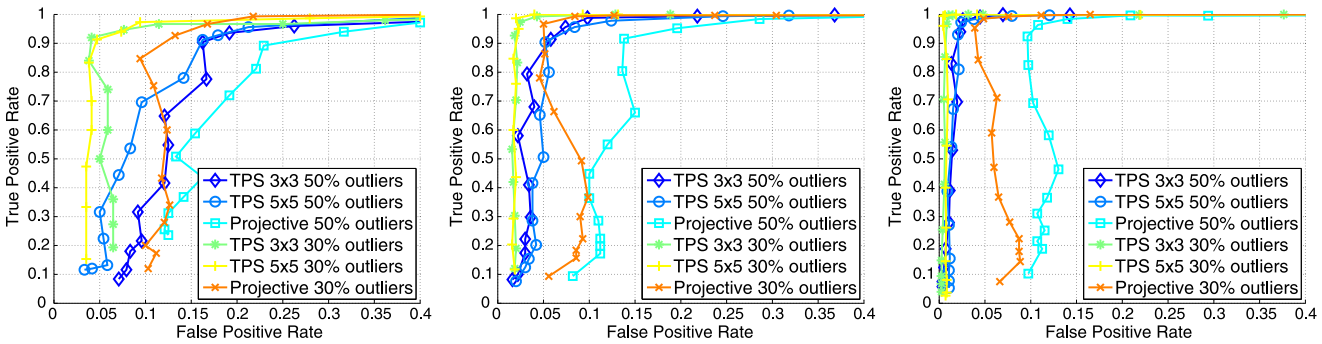
The ROC curves are used in this paper to establish a fair comparison between the different alternatives proposed. Each point in a ROC curve is obtained as the average TPR (True Positive Rate) and FPR (False Positive Rate) computed in the experiments using a particular value of d_{TH} to detect outliers. Ideally a perfect method should discard all outliers (TPR = 100%) without discarding useful inliers (FPR = 0%.) Therefore, the best d_{TH} that could be chosen in a single ROC curve can be found for the maximum possible TPR leaving the FPR below a reasonable value. The best TPR-FPR ratio can be used to compare different methods.

In the light of the results it is obvious that the larger the number of matches the better the results. The TPS warp clearly outperforms the homography in all tests. Surprisingly even with many points, the homography is far in terms of ROC from the TPS, which means that the deformable warp better captures the local properties of the motion field. With respect to the two different versions of the TPS, the one with 5×5 centers performs slightly better than the one with 3×3 centers. Despite the differences in all experiments the value of threshold $d_{TH} = 15$ pixels[3] always leads to a TPR of outlier detection above 90% producing always less than 15% of FPR.

Figures 8a and 8b show the TPR against the percentage of outliers for $d_{TH} = 15$ pixels. It can be observed how the proposed method is able to keep the TPR above 90% for 90% of outliers. However this high rate is at the cost of introducing too many false positives (more than 30%) for more than 70% outliers.

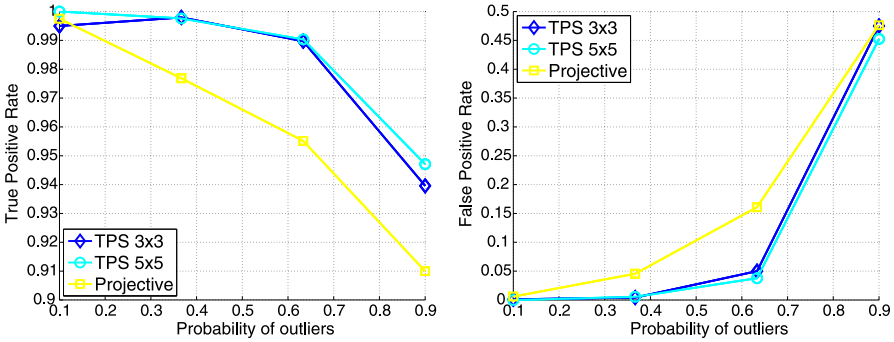
In Sect. 6, our outlier rejection method will be tested on real datasets. In these experiments we will use the TPS warp with 3×3 centers and $d_{TH} = 15$ pixels.

[3]The average point distance is 43, 35 and 21 pixels for the $N = 49$, $N = 100$ and $N = 225$ experiments, respectively.



(a) ROC curves for $N = 49$ matches (b) ROC curves for $N = 100$ matches (c) ROC curves for $N = 225$ matches

Fig. 7 ROC curves for the threshold d_{TH} against the different parameters of our outlier rejection method



(a) TPR vs p_o for $N = 200$ matches (b) FPR vs p_o for $N = 200$ matches

Fig. 8 Experiments comparing TPR and FPR versus the probability of outliers for $d_{TH} = 15$ pixels

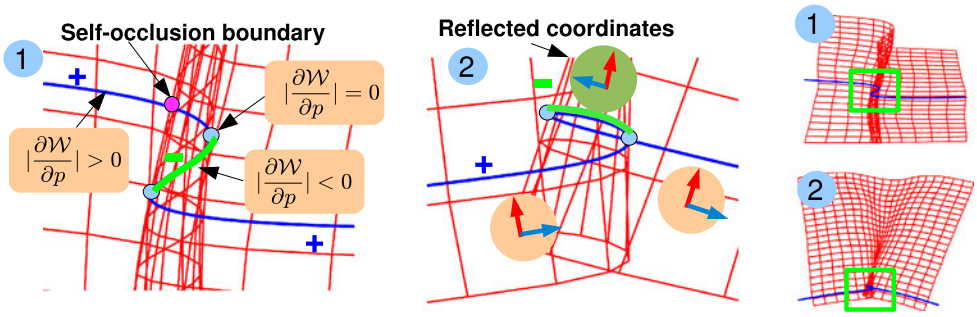


Fig. 9 Two examples of warp folds and their impact on the orientation of the local coordinates of the input image. The example labelled as 2 corresponds to the surface generated from the input image shown in Fig. 1

4 A Global Self-Occlusion Resistant Warp Estimator

This section presents the method we propose for handling warps with self-occlusions. Once the outlier-free set of matches $G_{\mathbf{p}} \leftrightarrow G_{\mathbf{q}}$ has been obtained, we compute the global warp $\mathcal{W}(\cdot, L)$ between the template image $\mathcal{T}$ and the input image $\mathcal{I}$. A warp is usually obtained by minimizing the average distance criterion between matches and a smoothness penalty as in Sect. 2.3. However, the resulting warp is constrained to be smooth, which implies that it can fold to comply with self-occlusions. Folds in regular warps make them many-to-many and we are interested in many-to-one warps in order to render convincing visual retexturing. In other words, the warp must 'collapse' the part of the template which has been self-occluded in the input image onto the self-occlusion boundary.

We propose a method with two stages. First, we compute what we call a *partial self-occlusion map*, that is, a sub- but sufficient part of the area in the template image $\mathcal{T}$ that suffers from self-occlusion in the input image. It is detected using an FFD warp computed from the matches $G_{\mathbf{p}} \leftrightarrow G_{\mathbf{q}}$. Second, a new FFD warp is obtained that does not fold in self-occluded areas. These two steps are described next. We then give a short experimental section showing how to learn the single method's parameter.

4.1 Finding the Self-Occlusion Map

The detection of self-occluded parts from a 2D warp presents several difficulties. Indeed, the warp can be quite complex, including multiple folds. The main question to answer is if an arbitrary point $\mathbf{p}$ in the template, warped to point $\mathbf{q} = \mathcal{W}(\mathbf{p}, L)$, is in a self-occluded zone or not.

Considering the warp as a 2D orientable manifold, the presence of a self-occlusion can be detected by looking at the orientation at each point of the warp. The notion of orientation for the warp coordinates can be obtained by considering the sign variations of the warp's Jacobian with respect to $\mathbf{p}$, denoted by $\frac{\partial \mathcal{W}}{\partial \mathbf{p}}$ (see Fig. 9 for an illustration). The warp's Jacobian matrix evaluated at point $\mathbf{p}$ represents a linear approximation of the warp for small variations around $\mathbf{p}$ and its warped coordinates $\mathbf{q}$. Taking the first order approximation of the warp in the vicinity of point $\mathbf{p}$:

$$\mathcal{W}(\mathbf{p} + \Delta\mathbf{p}, L) \approx \mathcal{W}(\mathbf{p}, L) + \frac{\partial \mathcal{W}}{\partial \mathbf{p}}(\mathbf{p}, L)\Delta\mathbf{p}, \tag{14}$$

where $\Delta\mathbf{p}$ is a small increment in the spatial coordinates. By renaming the term $\mathcal{W}(\mathbf{p} + \Delta\mathbf{p}, L) - \mathcal{W}(\mathbf{p}, L) = \Delta\mathbf{q}$ we get the following linear relationship:

$$\Delta\mathbf{q} \approx \frac{\partial \mathcal{W}}{\partial \mathbf{p}}(\mathbf{p}, L)\Delta\mathbf{p}. \tag{15}$$

Note that for the FFD warp the Jacobian matrix has a closed-form. According to (15), the warp's Jacobian matrix can be viewed as a general linear transformation between differential coordinates in the template ($\Delta\mathbf{p}$) and the input image ($\Delta\mathbf{q}$). The Jacobian $\eta(\mathbf{p}) = |\frac{\partial \mathcal{W}}{\partial \mathbf{p}}(\mathbf{p}, L)|$ gives us a clue about the orientation. If its sign is negative, it means that the warped coordinates $\mathbf{q}$ are locally reflected with respect to

those of $\mathbf{p}$. The reflection means that we are looking at the other side of the orientable surface defined by the warp at $\mathbf{q}$, and thus that the surface is self-occluded. If the sign of $\eta(\mathbf{p})$ is positive, it means that at point $\mathbf{p}$ the warp is not reflecting the coordinates and thus that point $\mathbf{q}$ could be visible. However, as the warp is a many-to-many transformation in case of self-occlusions, we cannot be sure that point $\mathbf{q}$ is visible unless we check for all points in the template region of interest $\mathcal{R}$ whose warped coordinates match up with $\mathbf{q}$ and $\eta(\mathbf{p}) > 0$. Figure 9 illustrates these concepts.

To sum up, independently of the nature of the warp configuration and the multiplicity of possible folds, the following statements are true for an orientable warp:

- A 'small' region around $\mathbf{p}$ with $\eta(\mathbf{p}) < 0$ is always self-occluded.
- A 'small' region around $\mathbf{p}$ with $\eta(\mathbf{p}) \geq 0$ can be either visible or self-occluded. If there are no other points $\mathbf{p}$ that have the same warped coordinates $\mathbf{q}$ then the point is not self-occluded. The reverse statement is not true as we need to establish a visibility order between points $\mathbf{r} \in \mathcal{R}$ that have $\eta(\mathbf{r}) \geq 0$ and result in the same warped coordinates $\mathbf{q}$.

We propose to build a partial self-occlusion map by detecting only those areas where $\eta(\mathbf{p}) \leq 0$, that we will refer to as self-occlusions, although we may be missing those points with $\eta(\mathbf{p}) > 0$ that may also be self-occluded. We show how to use this map to prevent the warp from folding without the need for finding all self-occluded pixels. We obtain the partial self-occlusion map by using a discretization approach. Given the template image $\mathcal{T}$, where the surface appears without deformation, we first build a two-dimensional regular partition $\mathcal{H}$ of the surface with $M \times M$ cells. Each cell is indexed as

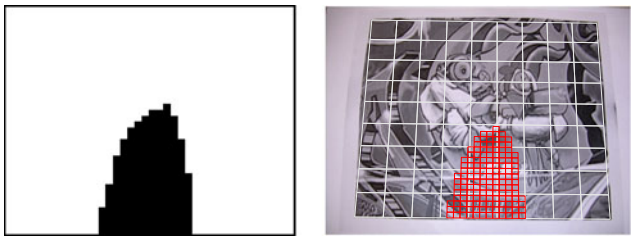
$$\mathcal{H}(\bar{\mathbf{p}}) \quad \bar{\mathbf{p}} = (\bar{x} \quad \bar{y})^\top \in \{1, \ldots, M\} \times \{1, \ldots, M\},$$

where $\bar{\mathbf{p}}$ are the cell coordinates corresponding to point $\mathbf{p}$ in the template image.[4] We define $\mathcal{H}(\bar{\mathbf{p}}) = 1$ if point $\mathbf{q}$ such that $\mathbf{p} \leftrightarrow \mathbf{q}$ is self-occluded (*i.e.* $\eta(\mathbf{p}) \leq 0$) in the input image $\mathcal{I}$ and $\mathcal{H}(\bar{\mathbf{p}}) = 0$ if $\mathbf{q}$ is visible. The partition $\mathcal{H}$ is the sought partial self-occlusion map. It can be viewed as a binary image where the self-occluded areas appear in black (see Fig. 10).

The partial self-occlusion map $\mathcal{H}$ is in practice built by computing the warp's Jacobian at every cell of the map and by defining the following factor in terms of the Jacobian matrix's eigenvalues λ_1, λ_2:

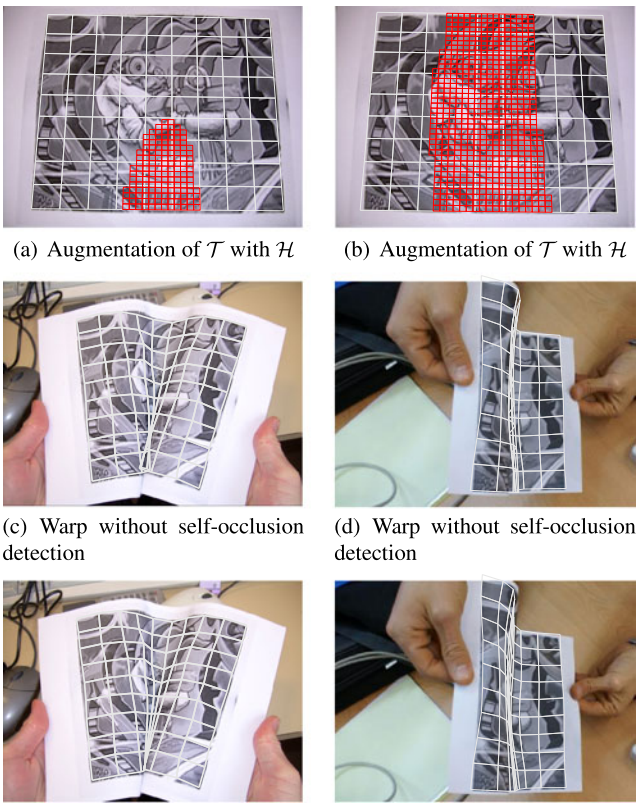
$$c = \text{sign}(\lambda_1)\,\text{sign}(\lambda_2)\,\min(|\lambda_1|, |\lambda_2|). \tag{16}$$

[4] We obtain $\mathbf{p}$ by simply scaling $\bar{\mathbf{p}}$ or by using a 2D homography if the template image needs rectification.



(a) Binary partial self-occlusion map $\mathcal{H}$ (b) Partial self-occlusion map $\mathcal{H}$ overlaid on the template $\mathcal{T}$

Fig. 10 An example of partial self-occlusion map



(a) Augmentation of $\mathcal{T}$ with $\mathcal{H}$ (b) Augmentation of $\mathcal{T}$ with $\mathcal{H}$

(c) Warp without self-occlusion detection (d) Warp without self-occlusion detection

(e) Warp with self-occlusion detection (f) Warp with self-occlusion detection

Fig. 11 Comparison between the warps obtained without and with the partial self-occlusion map $\mathcal{H}$

The factor c is a sensible choice as the smallest eigenvalue gives us, in pixel units, the amount of shrinking in the direction where the warp is collapsing due to self-occlusion, if any. A point $\mathbf{p}$ is marked as self-occluded if $c < \delta_H$ where δ_H is a small positive factor, which is chosen empirically to detect self-occlusions 'before' they occur in sequential processing (we have selected $\delta_H = 0.1$ pixels for all experiments.) Figures 11a and 11b show the map $\mathcal{H}$ obtained between two images for two different cases.

4.2 Self-Occlusion Resistant Warp Estimation

Once the self-occlusion map $\mathcal{H}$ has been obtained there are several ways to prevent the warp $\mathcal{W}(\cdot, L)$ between the sets $G_{\mathbf{p}} \leftrightarrow G_{\mathbf{q}}$ from folding.

In Gay-Bellile et al. (2010) the authors propose to use a 'shrinker', an extra term in the cost function which penalizes changes in the warp's local folds. Such a term is nonlinear in the warp parameters and requires an iterative algorithm to search for the minimum of the resulting cost function.

We propose a simple approach that forces an FFD warp to be quasi-affine in the regions marked in $\mathcal{H}$. We show that by over-smoothing the warp in the self-occluded areas, the warp behaves locally like a rigid transformation, shrinking rather than folding. We thus introduce a new cost function where the smoothing parameter is considerably larger in those areas where $\mathcal{H} = 1$ compared to the rest of the 'smooth' deformation areas. This approach, although simple, is very effective to avoid foldings and naturally shrinks the warp, allowing one to use linear least squares optimization. In Hilsmann and Eisert (2009) the authors propose a very similar approach, where the smoothness penalty for vertices in occluded regions of a triangular mesh is assigned a higher weight. The authors apply the idea to iteratively adjust the surface to optical flow measurements in a variational approach.

Although our method follows the same philosophy as Hilsmann and Eisert (2009), in contrast it provides a closed-form solution to the warp parameters given feature correspondences. This is an important step to bootstrap direct registration methods, where more sophisticated self-occlusion resistant cost functions can be used, such as the 'shrinker' term of Gay-Bellile et al. (2010). As we will show in Sect. 6, over-smoothing self-occluded areas performs slightly worst compared to using a 'shrinker' in direct approaches.

We modify the bending energy ϵ_s of (8) as follows:

$$\tilde{\epsilon}_s(L) = \sum_{\mathbf{p}} \kappa(\mathbf{p}) \left\| \frac{\partial^2 \mathcal{W}}{\partial \mathbf{p}^2}(\mathbf{p}, L) \right\|_F^2 \Delta x \Delta y, \tag{17}$$

where $\kappa(\mathbf{p})$ is:

$$\kappa(\mathbf{p}) = \begin{cases} K & \text{if } \mathcal{H}(\bar{\mathbf{p}}) = 1 \\ 1 & \text{otherwise.} \end{cases} \tag{18}$$

The constant parameter $K > 1$ is chosen using test experiments with various kinds of self-occlusions as explained below. The resulting smoothing term (17) can be formulated as a sum-of-squared linear combination Z_K of the warp parameters L:

$$\tilde{\epsilon}_s(L) = \| Z_K L \|_F^2. \tag{19}$$

The closed-form (11) can thus be readily used to obtain the warp by simply replacing Z by Z_K. It must be noted that in the absence of self-occlusion the smoothing term is the same as the conventional bending energy (8).

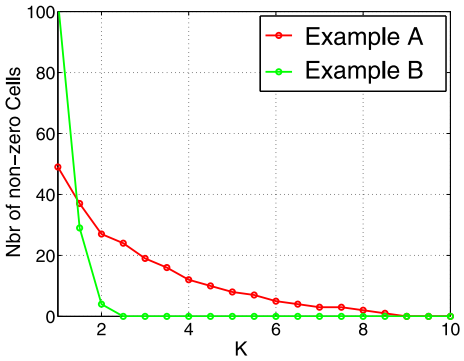


Fig. 12 Number of non-zero cells in $\tilde{\mathcal{H}}_K$ versus the value of K. Example A corresponds to Fig. 11c and Example B to Fig. 11d

Figures 11c and 11d show two different warps obtained without detecting self-occlusions, while Figs. 11e and 11f show the equivalent self-occlusion resistant warp obtained with our method. The partial self-occlusion map $\mathcal{H}$ is shown in Figs. 11a and 11b.

4.3 Selection of the Parameter K

In order to show the effect of the parameter K in the self-occluded areas we select two test examples affected with self-occlusion areas that are shown in Figs. 11c (Example A) and 11d (Example B). For both experiments we have computed the warp parameters L and a partial self-occlusion map $\mathcal{H}$.

We propose an experiment where several values are given to K (*i.e.* $K = 1, \ldots, 10$). For each value of K we compute new warp parameters L_K using the modified bending energy (18), and given $\mathcal{H}$. For each L_K we obtain a new partial self-occlusion map, namely $\tilde{\mathcal{H}}_K$, whose number of non-zero cells give us a way to measure the amount of folding produced by the warp parameters L_K. Each partial self-occlusion map $\tilde{\mathcal{H}}_K$ is obtained with a value $\delta_H = 0$, so that it does not 'anticipate' self-occlusions.

It is expected that by increasing K, the number of self-occluded (non-zero) cells tends to zero. In Fig. 12c we show the number of such cells in $\tilde{\mathcal{H}}_K$ versus the value of K for the two test cases. It can be observed that for $K > 10$ all cells get to zero, which means that the resulting warp did not fold in self-occluded areas. In all our experiments we choose the value $K = 20$.

5 Pixel-Based Registration Refinement

The feature-based approach presented in this paper allows us to detect a deformed instance of the template provided enough matches are given, while discarding outliers. The

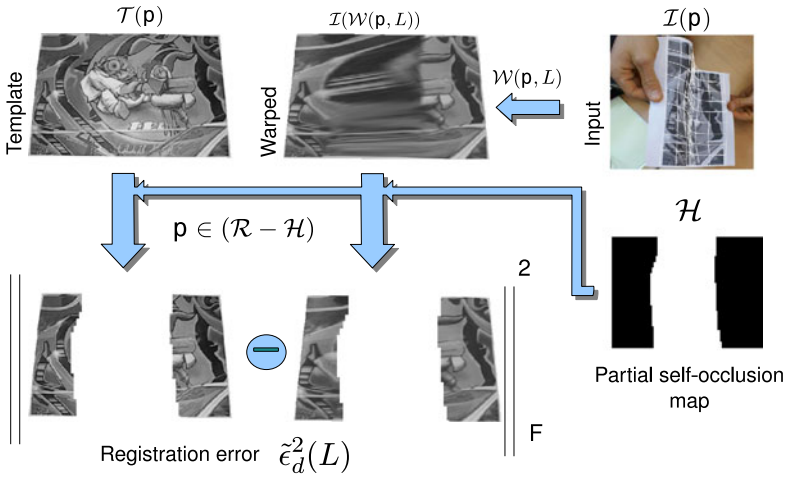


Fig. 13 Illustration of the pixel-based data term $\tilde{\epsilon}_d$, including the partial self-occlusion map $\mathcal{H}$

registration accuracy obviously depends on the keypoint detection and matching methods. Nevertheless, most of the methods often lose many points near self-occlusion boundaries or highly deformed areas. In these cases the accuracy of our proposed method decreases. We thus propose to use a pixel-based (direct) registration approach to refine the surface parameters given by our feature-based method. Pixel-based registration uses pixel value differences to obtain the warp parameters and therefore theoretically use all the information available in the image to compute an accurate registration. In Gay-Bellile et al. (2010), a pixel-based registration approach is proposed for deformable surface detection that includes self-occlusion reasoning. In contrast, the method we propose has a simpler mechanism to deal with self-occlusions, using our modified bending energy (17). In Sect. 6 we show the differences between both approaches.

5.1 Defining the Cost Function

The pixel-based cost function we propose is composed of a data-term $\tilde{\epsilon}_d$, that compares intensity differences between the template and the warped image, and the smoothing term $\tilde{\epsilon}_s$ we proposed in (17):

$$\tilde{\epsilon}^2(L) = \tilde{\epsilon}_d^2(L) + \tilde{\mu}^2 \tilde{\epsilon}_s(L), \tag{20}$$

with:

$$\tilde{\epsilon}_d^2 = \sum_{\mathbf{p} \in (\mathcal{R} - \mathcal{H})} \|\mathcal{T}(\mathbf{p}) - \mathcal{I}(\mathcal{W}(\mathbf{p}, L))\|^2, \tag{21}$$

where $\mathcal{R}$ is the template region of interest (usually the whole template image.) The Euclidean norm used to compare pixel intensities or colors can be replaced by a robust loss function such as the one used in Gay-Bellile et al. (2010) so as to robustify the cost against external occlusions and other unmodeled phenomena. The region $\mathcal{R} - \mathcal{H}$ represents all pixel positions in the template region of interest that are not self-occluded (*i.e.* where $\mathcal{H}(\bar{\mathbf{p}}) = 0$). In those positions where the self-occlusion map is non-zero the warped image $\mathcal{I}(\mathcal{W}(\mathbf{p}, L))$ is not well defined (see Fig. 13).

5.2 Minimizing the Cost Function

A straightforward way to obtain the minimum of the cost (20) is to use the Gauss-Newton iterative optimization approach. The warp parameters are additively updated at each iteration (*i.e.* $L = L + \Delta$) for some increment Δ. The Gauss-Newton approximation of $\tilde{\epsilon}_d$ is given by:

$$\tilde{\epsilon}_d^2(L + \Delta) \approx \sum_{\mathbf{p} \in (\mathcal{R} - \mathcal{H})} \|\mathcal{T}(\mathbf{p}) - \mathcal{I}(\mathcal{W}(\mathbf{p}, L)) - g(\mathbf{p}, L)^\top \delta\|^2, \tag{22}$$

where $\delta = \text{vect}(\Delta)$ is the update vector.[5] The gradient vector $g(\mathbf{p}, L)$ factors in the image gradient vector $\nabla\mathcal{I}$ and the warp's Jacobian matrix $\frac{\partial \mathcal{W}}{\partial L}$, which is linear with respect to L and constant. The Gauss-Newton method repeatedly solves a linear least squares minimization problem in Δ. The normal equations are formed by stacking all gradient vectors into matrix $\mathcal{J}$ and all image residuals in vector D, giving the following approximation of the data term:

$$\tilde{\epsilon}_d^2(L + \Delta) \approx (\mathcal{J}\delta - D)^\top (\mathcal{J}\delta - D). \tag{23}$$

According to (19) the smoothing term can be directly expressed as the following quadratic form in the warp parameters (in vectorized form $\ell = \text{vect}(L)$):

$$\tilde{\epsilon}_s^2(L + \Delta) = \Big((\ell + \delta)^\top \mathcal{K} (\ell + \delta)\Big), \tag{24}$$

[5] vect refers to the column-wise vectorization of a matrix.

where $\mathcal{K} = \text{diag}(Z_K, Z_K)$. Finally the complete cost function (20) is approximated by a quadratic function in δ:

$$(\mathcal{J}\delta - D)^\top(\mathcal{J}\delta - D) + \tilde{\mu}^2\left((\ell+\delta)^\top\mathcal{K}(\ell+\delta)\right). \tag{25}$$

The minimum of (25) with respect to δ can be computed by nullifying its partial derivatives, as the solution to the system:

$$H\delta = B, \tag{26}$$

with:

$$H = \mathcal{J}^\top\mathcal{J} + \tilde{\mu}^2\mathcal{K}, \quad B = \mathcal{J}^\top D - \tilde{\mu}^2\ell. \tag{27}$$

Matrix $\mathcal{J}$ must be recomputed at each iteration as it depends on the warp parameters, which implies that the Hessian H must be recomputed and inverted too. The inversion process can be done in a very efficient manner as matrix H is sparse due to the limited support of the bicubic B-Spline basis function. Since the Gauss-Newton approach does not necessarily guarantee that the registration error decreases at each iteration we provide the algorithm with a stopping condition when the error increases. This condition can also be imposed with a region-trust algorithm like Levenberg-Marquardt.

A brief algorithmic view of our registration method is now given:

1. Initialize the warp parameters L using the feature-based method presented in Sect. 3 (which also give the partial self-occlusion map $\mathcal{H}$).
2. Precompute the bending energy matrix $\mathcal{K}$ and the warp's Jacobian matrix $\frac{\partial\mathcal{W}}{\partial L}$, set $n = 1$.
3. **Repeat**
   (a) Compute the warped image $\mathcal{I}(\mathcal{W}(\mathbf{p}, L))$ and its gradient vectors $\nabla\mathcal{I}$; form $\mathcal{J}$ and D.
   (b) Compute H and B from (27).
   (c) Use a sparse solver[6] to solve the linear system $H\delta = B$.
   (d) Reshape δ to Δ and temporally update parameters $L = L + \Delta$, set $n = n + 1$.
   (e) **If** the error $\tilde{\epsilon}^2$ increases with updated parameters **then** reject the update and exit.
   (f) **If** $\|\delta\| < \varepsilon$ **or** $n \geq n_{max}$ **then** exit.

This iterative algorithm may stop when the incremental step δ is smaller than a predefined value ε, typically 10^{-8}.

6 Experimental Results

This section shows the performance of both the outlier rejection method and the self-occlusion resistant warp estimation for three datasets. Each dataset has a template image and a sparse set of views of the deformed surface, taken under different viewpoints and with different deformations, including self-occlusions. We then give implementation and timing details.

6.1 Detection and Retexturing Results

The *Comics dataset* has 6 images of size 968×648, including the template. The images present different kinds of deformations including self-occluded surfaces and strong deformations due to paper creasing. Figure 14a shows the template image with a warp visualization grid. Figures 14c to 14e show the result of automatic surface detection based only on keypoints, for different viewpoints, with strong self-occlusions in Figs. 14d and 14e. Figures 14f to 14h show the surface retextured. Below each image we give the number of matched keypoints N, the number of outliers detected N_o, the registration error ϵ_d (5) in terms of transferred point distances (in pixels) and the registration (photometric) error ϵ_r (in pixel intensity units $[0 - 255]$), corresponding to the mean difference between pixel values:

$$\epsilon_r = \frac{1}{|\mathcal{R} - \mathcal{H}|}\sum_{\mathbf{p}\in(\mathcal{R}-\mathcal{H})}\|\mathcal{T}(\mathbf{p}) - \mathcal{I}(\mathcal{W}(\mathbf{p}, L))\|. \tag{28}$$

The same experiment is shown in Figs. 14i to 14n after the pixel-based registration refinement step based on the idea of over-smoothing the self-occluded pixels using a modified bending energy term. Figures 14o to 14t show the results of pixel-based registration using the shrinker term proposed in Gay-Bellile et al. (2010). For this last batch of figures, we only show the registration error ϵ_r.

In all images displayed in Fig. 14 the registration error is small and the retextured images are visually convincing even with self-occlusions. However, only a few keypoints were detected near the self-occluded parts, which produced artifacts, especially in Fig. 14e. The pixel-based refinement step is able to reduce the registration error ϵ_r in all cases, creating a more visually convincing registration (see Fig. 15 for a close-up view of the retexturing for one image of the Comics dataset). It can be observed that the pixel-based registration using a shrinker term has a lower registration error and gives visually more convincing results.

Figures 16a and 16b show surface detection with strong deformations created by creasing the surface. As it can be observed, some parts of the mesh fail to fit the surface, producing greater registration errors than in Fig. 14. The errors are produced by the lack of keypoints near some complex foldings created in the paper. Still, in those parts where enough points are found the retextured images shown in Figs. 16c and 16d are visually convincing. As in Fig. 14 the pixel-based refined versions of Figs. 16e to 16l are able to reduce the registration error and are more visually convincing. The pixel-based registration approach based on using a

[6] We use the 'mldivide' Matlab function.

Fig. 14 Results of the proposed framework on the Comics dataset (see main text for details)

Fig. 15 Close-up view of Fig. 14g (*left*) where only keypoints were used for registration, Fig. 14m (*middle*) where pixel-based registration was used including over-smoothing penalty to self-occlusions and Fig. 14s where a shrinker term is used in self-occlusions

shrinker term is slightly better than the approach using the over-smoothing penalty in all cases.

It can be seen that the top-left hand corner of the paper surface shown in Fig. 16b is not registered properly even after pixel-based refinement. This example is especially difficult. The feature-based approach suffers from the lack of features and smoothes the surface. Consequently, the self-occlusion map estimated from the feature-based solution does not detect the corner of the surface as a self-occluded part. Then, the pixel-based approach is not able to tell the difference between a self-occlusion and an external occlusion for that part of the warp. In Gay-Bellile et al. (2010) the authors show that a direct registration approach with a shrinker term is able to adapt to a video sequence where a similar bending occurs in the corner of a paper. In that case the continuous sequence allows the self-occlusion map to be estimated iteratively at each frame, making the surface to shrink properly at the corner.

The *T-Shirt dataset* has 3 images of size 968 × 648 of a textured t-shirt: the template and 2 input images with deformations. Figures 17a and 17b show the template image augmented with the surface mesh and the retextured template respectively. Figures 17c to 17f show the surface detected with the vector $[N, N_o, \epsilon_d, \epsilon_r]$, as for the previous dataset. Figures 17g to 17n show the result of pixel-based registration refinement and the new registration error ϵ_r with both the over-smoothing penalty and a shrinker term. In both cases pixel-based registration decreases the registration error. In this dataset there is no visible difference between the shrinker and the over-smoothing terms.

The *Graffiti dataset* has 4 images of size 320 × 640, extracted from a video sequence and processed separately. Figures 18a to 18d show the template and 3 input images, and the warp visualization grid. As in the other datasets we caption each Fig. of the retextured image with $[N, N_o, \epsilon_d, \epsilon_r]$. This dataset has no self-occlusion and shows accurate registration results without the need for pixel-based refinement.

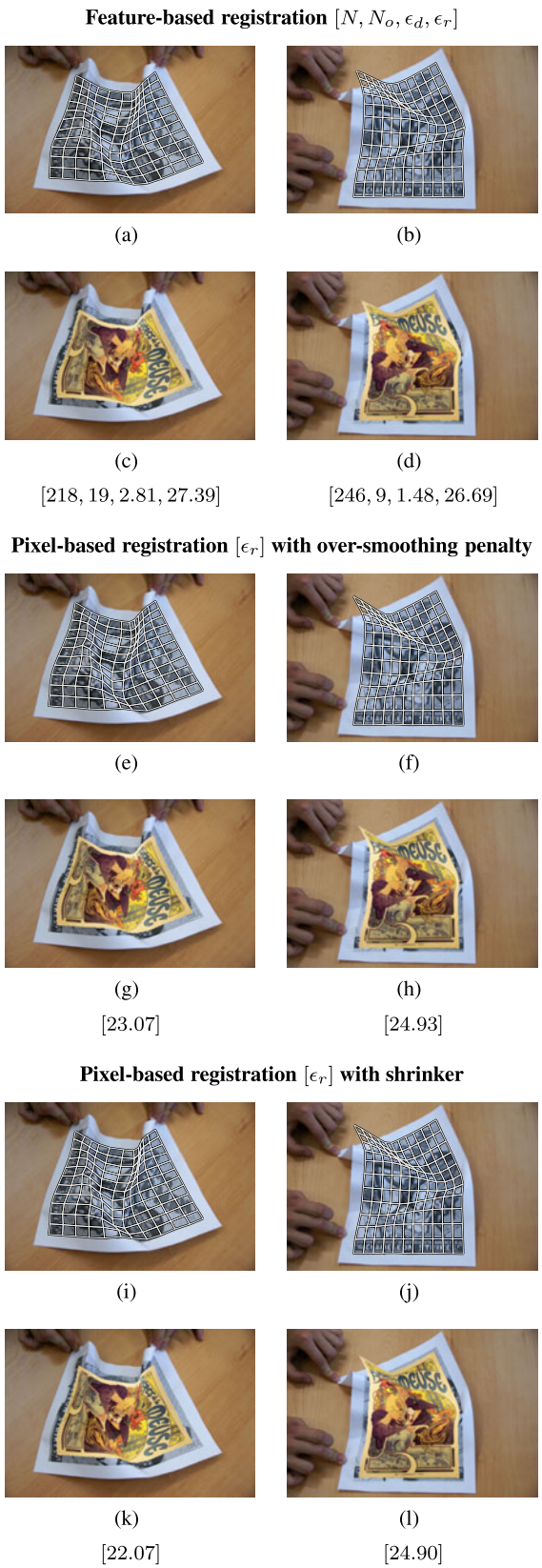


Fig. 16 Results of the proposed framework on the Comics dataset with creased paper (see main text for details)

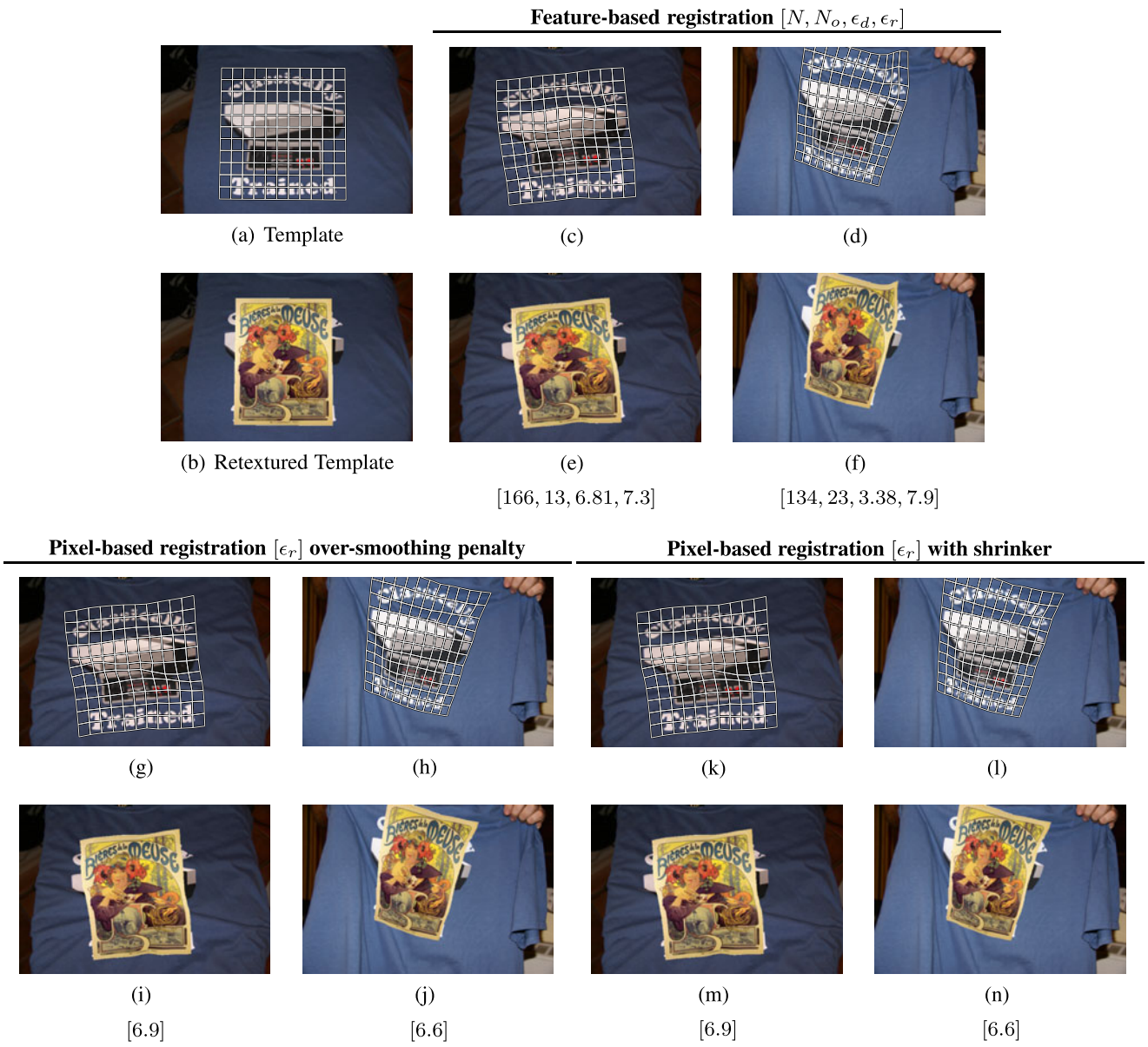


Fig. 17 Results of the proposed framework on the T-Shirt dataset (see main text for details)

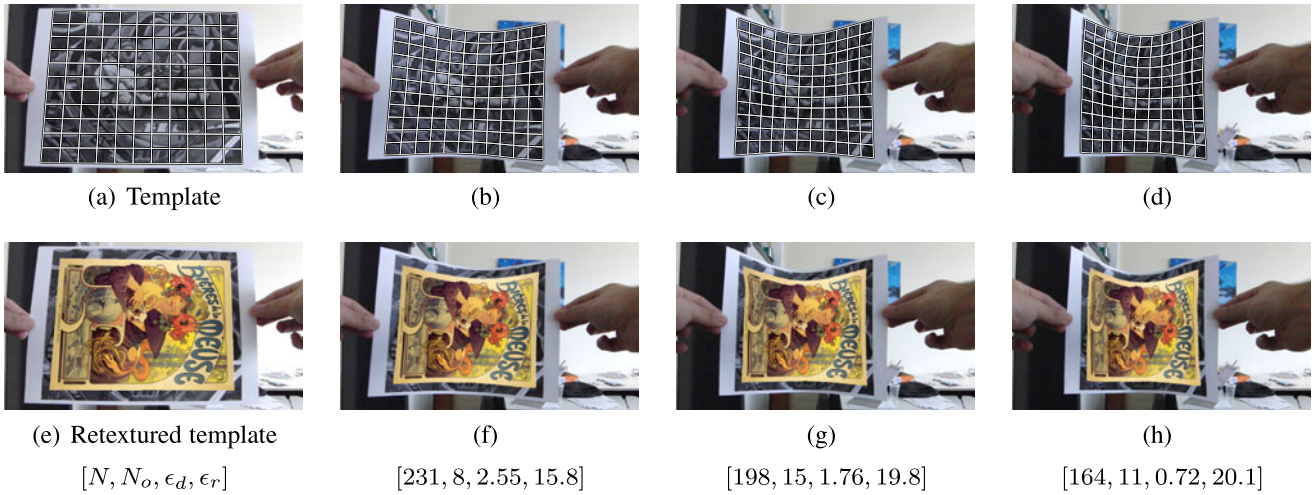
In Fig. 19 we highlight two of the experiments where the number of outliers found between the template and input images is high. We render the matches for both template and input images and show motion vectors superimposed on the template. The percentage of outliers is indicated in the caption of Fig. 19. The values oscillate between 15%–20% of outliers.

6.2 Software Implementation and Timing

All experiments were carried out using a Matlab implementation of our algorithms. We give timing details for some of the experimental conditions given in the paper in Table 2. The computer used for the test was a medium performance Intel Core 2 Duo CPU at 2.2 GHz. We believe that an implementation in nearly interactive rate could be achieved with an optimized code in a compiled programming language such as C++ with the same platform and algorithms. The bottleneck is clearly the pixel-based refinement. It could be made much faster using a number of improvements such as coarse-to-fine processing.

7 Conclusions

This paper has presented the first method for detecting a deformed surface in a single image based on point matches

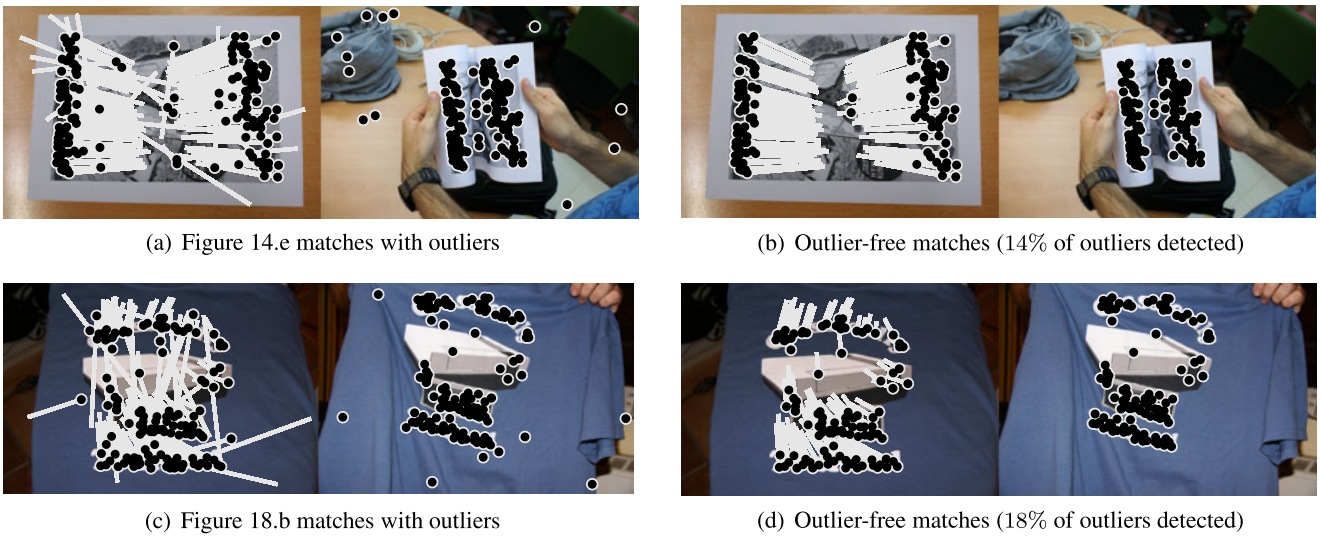


(a) Template (b) (c) (d)

(e) Retextured template (f) (g) (h)

$[N, N_o, \epsilon_d, \epsilon_r]$ $[231, 8, 2.55, 15.8]$ $[198, 15, 1.76, 19.8]$ $[164, 11, 0.72, 20.1]$

Fig. 18 Results of the proposed framework on the Graffiti dataset (see main text for details)



(a) Figure 14.e matches with outliers (b) Outlier-free matches (14% of outliers detected)

(c) Figure 18.b matches with outliers (d) Outlier-free matches (18% of outliers detected)

Fig. 19 Matches and motion vectors superimposed for the experiments shown in Figs. 14e and 18b

Table 2 Timing performance of our Matlab software implementation for the different experiments

Experiment	Phase	Time
Comics dataset, Fig. 14d	Outlier detection	1.94 secs
	Keypoint-based registration	3.01 secs
	Pixel-based registration (15 iterations)	16.98 secs
T-Shirt dataset, Fig. 17c	Outlier detection	0.94 secs
	Keypoint-based registration	2.95 secs
	Pixel-based registration (15 iterations)	12.98 secs
Graffiti dataset, Fig. 18d	Outlier detection	1.78 secs
	Keypoint-based registration	2.86 secs

to a template that handles both wide-baseline and self-occlusions. The paper contributes with a simple and effective method for automatically discarding outliers, based on the assumption that the surface to detect is locally smooth. Outlier removal is based on using relationships between neighboring points. The experimental evaluation confirmed that our method is reliable enough even with a high amount of outliers in the matches (beyond 70% with a TPR of more

than 90% and a FPR of less than 15%) which is more than necessary for real applications. Our outlier rejection method has the advantage over other similar works that it can be applied to self-occluded surfaces. The other contributions of the paper is a simple method to detect self-occlusions and to compute a self-occlusion resistant warp function—based on either the point matches or a pixel-based cost. This method allows us to perform convincing automatic retexturing. We showed how bootstrapping our pixel-based registration engine with a point-based solution leads to a robust wide-baseline algorithm giving accurate results. We believe that, even more than in rigid registration where fewer parameters are sought, combining feature-based and pixel-based methods is a sensible way of solving deformable registration problems.

Acknowledgements This work has been partly supported by the Spanish Ministry of Science and Innovation under projects VISNU (ref. TIN2009-08984) and SDTEAM-UAH (ref. TIN2008-06856-C05-05) and the French ANR through the HFIBMR Project. We thank the anonymous reviewers for their constructive feedback and helpful suggestions.

Appendix: Efficient Computation of the Distances d_i with a TPS

The computation of the predicted point $\hat{\mathbf{p}}_i$ and consequently of the value of d_i can be greatly simplified if the function f in (12) is chosen to be a TPS. Following the feature-driven parameterization of the TPS (Bartoli 2008), the parametric warp shown in (3) is simplified into the following expression:

$$\mathcal{W}(\mathbf{p}, L) = L^\top \nu(\mathbf{q}) = L^\top \mathcal{E}_\lambda^\top \mathbf{l}_\mathbf{p}, \tag{29}$$

where $\mathcal{E}_\lambda$ is an $(l+3)\times l$ matrix that depends on the template control points coordinates $\mathbf{c}_1^p, \ldots, \mathbf{c}_l^p$ (usually chosen on a regular grid) and on the parameter $\lambda \in \mathbb{R}$, modeling the strength of internal smoothing.

The vector $\mathbf{l}_\mathbf{p}$ with $l+3$ components is defined as follows:

$$\mathbf{l}_\mathbf{p} = \left(\rho(\|\mathbf{p}-\mathbf{c}_1^p\|^2) \;\cdots\; \rho(\|\mathbf{p}-\mathbf{c}_l^p\|^2)\; \mathbf{p}^\top \; 1\right), \tag{30}$$

where $\rho(d) = d\log(d)$ is the TPS kernel function. Using (29), the matrix A used for computing the influence matrix is transformed into the following expression:

$$A^\top = \mathcal{E}_\lambda^\top \left(\mathbf{l}_{\mathbf{p}_1}, \ldots, \mathbf{l}_{\mathbf{p}_n}\right). \tag{31}$$

By defining $\bar{\mathcal{E}}_\lambda$ as the first l rows of $\mathcal{E}_\lambda$, the bending energy ϵ_s can be written as:

$$\epsilon_s = 8\pi \left\| \sqrt{\bar{\mathcal{E}}_\lambda} L \right\|_F^2, \tag{32}$$

which means that $Z^\top Z = 8\pi\bar{\mathcal{E}}_\lambda$ in the influence matrix. The warp function and also the influence matrix shown in (12) are simplified since $\mathcal{E}_\lambda$ can be precomputed.

In the problem of computing each distance d_i we have two neighboring sets $Q(\mathbf{p}_i)$ and $Q(\mathbf{q}_i)$ of points $\mathbf{p}_i$ and $\mathbf{q}_i$ respectively. In this case we define a TPS warp function $\mathbf{p} = \mathcal{W}(\mathbf{q}, L)$ that goes from the input image to the template. Therefore matrix L includes control point coordinates in the coordinates of $\mathbf{p}_i$ and the predefined control points used in $\mathcal{E}_\lambda$ are defined as $\mathbf{c}_1^q, \ldots, \mathbf{c}_n^q$ in the coordinates of $\mathbf{q}_i$.

We propose the following algorithm to efficiently compute the distance d_i:

1. Distribute the control points $\mathbf{c}_1^q, \ldots, \mathbf{c}_l^q$ uniformly in the interval $[0,1]\times[0,1]$ and compute $\mathcal{E}_\lambda$ and $\bar{\mathcal{E}}_\lambda$.
2. For all $i = 1, \ldots, N$
   (a) Normalize the set $Q(\mathbf{q}_i)$ so that it lies inside the interval $[0,1]\times[0,1]$, name it $\bar{Q}(\mathbf{q}_i)$.
   (b) Compute the warp parameters L using the influence matrix (3), the matches $Q(\mathbf{p}_i) \leftrightarrow \bar{Q}(\mathbf{q}_i)$ and the value of $\mathcal{E}_\lambda$ and $\bar{\mathcal{E}}_\lambda$ previously computed.
   (c) Warp point $\mathbf{q}_i$ to get the predicted point $\hat{\mathbf{p}}_i$ using (29).
   (d) Compute the distance $d_i = \|\hat{\mathbf{p}}_i - \mathbf{p}_i\|$.

References

Baker, S., & Matthews, I. (2004). Lucas-Kanade 20 years on: a unifying framework. *International Journal of Computer Vision*, *56*(3), 221–255.

Bartoli, A. (2008). Maximizing the predictivity of smooth deformable image warps through cross-validation. *Journal of Mathematical Imaging and Vision*, *31*(2), 133–145.

Bartoli, A., & Zisserman, A. (2004). Direct estimation of non-rigid registrations. In *British machine vision conference* (vol. 2, pp. 899–908).

Bay, H., Ess, A., Tuytelaars, T., & Van Gool, L. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, *110*(3), 346–359.

Bookstein, F. L. (1989). Principal warps: thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *11*(6), 567–585.

Chui, H., & Rangarajan, A. (2003). A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, *89*(2–3), 114–141.

DeCarlo, D., & Metaxas, D. (1998). Deformable model-based shape and motion analysis from images using motion residual error. In *International conference on computer vision* (vol. 39, pp. 380–393).

Duchon, J. (1976). Interpolation des fonctions de deux variables suivant le principe de la flexion des plaques minces. *RAIRO Analyse Numérique*, *10*, 5–12.

Gay-Bellile, V., Bartoli, A., & Sayd, P. (2010). Direct estimation of nonrigid registrations with image-based self-occlusion reasoning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(1), 87–104.

Hilsmann, A., & Eisert, P. (2009). Tracking and retexturing cloth for real-time virtual clothing applications. In *Computer vision/computer graphics collaboration techniques: 4th international conference*, MIRAGE 2009, Rocquencourt, France, May 4–6, 2009 (pp. 94–105). New York: Springer.

Lepetit, V., & Fua, P. (2006). Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(9), 1465–1479.

Lischinski, D. (1994). Incremental delaunay triangulation. *Graphics Gems*, *IV*, 47–59.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110.

Pilet, J., Lepetit, V., & Fua, P. (2008). Fast non-rigid surface detection, registration and realistic augmentation. *International Journal of Computer Vision*, *76*(2), 109–122.

Pizarro, D., & Bartoli, A. (2010). Feature-based deformable surface detection with self-occlusion reasoning. In *Electronic proceedings of the fifth international symposium on 3D data processing, visualization and transmission*, 3DPVT'10.

Pizarro, D., Peyras, J., & Bartoli, A. (2008). Light-invariant fitting of active appearance models. In *Proc. IEEE conf. on computer vision and pattern recognition*, CVPR 2008, Anchorage, Alaska, USA (pp. 1–6).

Rueckert, D., Sonoda, L. I., Hayes, C., Hill, D. L. G., Leach, M. O., & Hawkes, D. J. (1999). Nonrigid registation using free-form deformations: application to breast MR images. *IEEE Transactions on Medical Imaging*, *18*(8), 712–721.