Volume 0 (1981), Number 0 pp. 1–14 COMPUTER GRAPHICS *forum*

Partial Shape Matching using Transformation Parameter Similarity

Paul Guerrero[1] Thomas Auzinger[1] Michael Wimmer[1] Stefan Jeschke[2]

[1]Vienna University of Technology [2]IST Austria

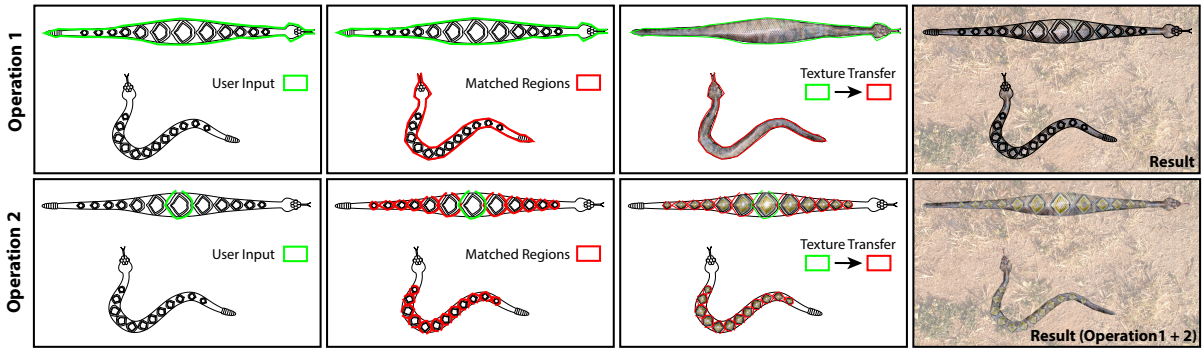

Figure 1: *Two snakes related by a highly non-rigid transformation can be matched with our method. A user selects a region (green) and similar regions (red) are automatically found, making it possible to transfer the texture inside the green region to the red regions. Two operations are performed and the results with an added background texture are shown in the last column.*

Abstract

In this paper, we present a method for non-rigid, partial shape matching in vector graphics. Given a user-specified query region in a 2D shape, similar regions are found, even if they are non-linearly distorted. Furthermore, a non-linear mapping is established between the query regions and these matches, which allows the automatic transfer of editing operations such as texturing. This is achieved by a two-step approach. First, point-wise correspondences between the query region and the whole shape are established. The transformation parameters of these correspondences are registered in an appropriate transformation space. For transformations between similar regions, these parameters form surfaces in transformation space, which are extracted in the second step of our method. The extracted regions may be related to the query region by a non-rigid transform, enabling non-rigid shape matching.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Hierarchy and geometric transforms I.4.8 [Computer Graphics]: Scene Analysis—Shape

1. Introduction

Geometric manipulation of 2D images or 3D models is a central component of many computer graphics applications. Throughout the years, these editing tools have become increasingly sophisticated and a recent research trend is the intelligent manipulation of many similar shapes with only a single interaction. For a wide range of applications, the ability to quickly select and edit model parts that are geometrically *similar* – but not *identical* – can considerably increase the efficiency of the editing work flow.

This inevitably leads to the field of shape matching which traditionally studies point correspondences. A point correspondence matches a point in a user-defined query region to similar points outside (target points) according to a local descriptor, such as SIFT [Low04] or Shape Contexts [BMP02]. In this paper, we refer to these point correspondences as *first-*

order similarities. In order to match a whole query region, different approaches exist. One is to find groups of points correspondences that preserve the pair-wise point distances of the query region in their respective target region, which is used in isometric shape matching [EK03, ASP*05, MS05, BBK06, PBB*13]. A different approach tries to find correspondences that have *similar transformations* between query and target points [GCO06, MGP06, MGP07] (cf. Figure 2a and 2b). We will refer the similarity between point correspondences as *second-order similarity*.

Given a query region, it is often desirable to identify non-linearly distorted regions, due to perspective, object bending, different object positions and orientations, etc. Unfortunately, requiring the matched point pairs to have an identical transform only lets us find *rigid* matches. In this work, we overcome this limitation by introducing the concept of *transformation parameter similarity* (TPS). We regard two regions as similar if they are related by approximately rigid first-order similarities and if the corresponding transformation parameters vary slowly over the region, i.e., have a small gradient. An example is given in Figure 2c, where the subregions are related by approximately rigid transforms, whereas globally, the two regions are transformed non-rigidly. Note that the rotation parameter varies slowly over the region. Our requirement of approximately rigid *local* transformations ensures that small-scale geometric detail is preserved and the region is not distorted beyond recognition.

Based on this concept, the main contribution of this paper is a novel method to find non-rigid matches to a user-specified query region in a vector image. Furthermore, we establish a dense mapping between the query region and all matched regions. This allows the transfer of editing operations between complex 2D shapes, such as painted strokes or textures (cf. Figure 1). Currently, our algorithm focuses on 2D shapes given by contour segments, but it could be generalized to 3D shapes given by 2D boundary meshes in future work.

2. Related Work

First-order point descriptors have been thoroughly researched and two main categories can be identified. Local descriptors, such as SIFT features [Low04] or shape contexts [BMP02] provide a matching of local shape features. Global descriptors, like Shape-DNA [RWP06] or the Heat Kernel Signature [SOG09], take whole shape into account, when establishing point correspondences, but are usually more expensive to compute and they are not applicable to our use case of 2D shapes. In this work, we use shape contexts to establish first-order similarities.

The matching of 3D shapes received a great deal of attention in graphics research and a wide range of second-order similarity concepts have been developed. The most prominent example is isometric shape matching [EK03, ASP*05, MS05, BBK06, PBB*13] which uses the distortion of geodesic distances on a shape's surface as similarity criterion. It allows all transformation as long as the pairwise geodesic distances of the query points are preserved in the matched region. In contrast to our method, large deformations over small areas, such as joints, are considered similar transformations (cf Figure 11c). Our similarity measure requires locally approximately rigid transformations but allows their gradual change over the region. This not only preserves local geometric detail, but can also handle large-scale shape changes, such as strong bending of thick objects over a large region, or a gradual change of scale, both of which are not considered similar by an isometric approach. Furthermore, our method does not require connected shape boundaries, which fits well to our use case of line-drawings or edge images, where we often have to deal with disjoint curves.

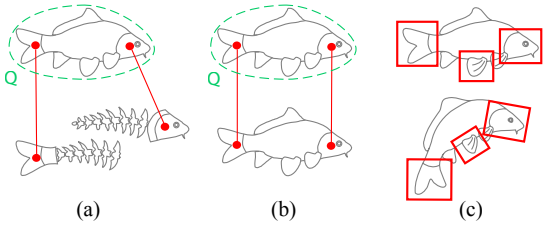

Figure 2: *Rigidly transformed regions can be found by searching for point correspondences with similar transformations. The correspondences in (b) have a similar transformation and belong to the same rigidly transformed region, in contrast to the correspondences in (a). The two fish in (c) are related by a non-rigid transform, whereas each pair of matching subregions is transformed rigidly.*

A different concept of second-order similarity was proposed by Zhang et al. [ZSCO*08], who computes a deformation energy to align the points of a query region with their matched counterparts. Due to its exponential complexity, only a very sparse set of points can be used and the geometry between them is ignored. Chang et al. [CZ08] match two articulated shapes by partitioning them into a small sets of subregions that are related by rigid transformations and optimize pairwise correspondences between the subregions. Smooth non-rigid transformations like bending cannot be handled. Berner et al. [BWM*11] infer a second-order similarity from a given set of shapes by identifying a low-dimensional subspace in the full space of all possible transformations between shape regions. To span the subspace, a large amount of similar shapes are required but usually not available for our use cases. Furthermore, rotations are not well presented in their space of transformations, which causes related non-rigid transformations, such as bending, to be handled poorly.

Various methods [GCO06, MGP06, MGP07] define second-order similarity based on transformation parameter similarity. However, the first two methods [GCO06, MGP06] can only match regions related by a rigid transformation,

while the third method [MGP06] is only suited for symmetrizing a single model and cannot be employed to match two different regions. Our approach can be seen as a generalization of these methods as our similarity measure is also based on transformation parameters. An in-depth description of this concept is given in Section 3.

In regard to the input of our method, i.e., a set of contour segments, several related methods were proposed in the last years. In Ferrari et al. [FJS10], pairs of adjacent segments are identified that potentially match parts of the query object contours. Each pair votes for a object position and scale and pairs that agree on their votes are used to build the outline of the matching objects. Lu et al. [LLA*09] use a similar voting scheme to capture the close-to-rigid relations between contour segments. Srinivasan et al. [SZS10] use shape histograms to encode the relations between contour segments similar to shape contexts. Ma et al. [ML11] matches small partial contour segments first and groups those with similar relative position and orientation to assemble the full contours. These four methods can handle small variations of the matching contours but fail for strong non-rigid transforms such as bending.

Several methods [CR03, ZD06, MZT*13] have been proposed to find a non-rigid matching between two sets of 2D points by iteratively optimizing correspondences and transformation. Chui et al. [CR03] use thin-plate splines to model the transformation and a soft-assign method [RCB97] to compute correspondences. Zheng et al. [ZD06] employ the thin-plate spline model as well, but use a local descriptor based on a points neighborhood structure to compute correspondences. Ma et al. [MZT*13] use an estimator that is robust to outliers to estimate a transformation modeled as a function lying in a reproducing kernel Hilbert space. Shape contexts are used to compute correspondences. These methods work well if there is an unambiguous (i.e. single best) solution for the matching problem. However, if there are multiple sets of matching points, their iterative optimization is prone to get stuck in local minima. See Section 7 for a comparison of the most recent method [MZT*13] with ours.

Second-order similarity has also been explored for region matching in the context of raster images. These methods not only consider geometric similarity as the aforementioned works, but also photometric similarity measures. PatchMatch [BSFG09] and its generalization [BSGF10] search for similar patches in an image and grow regions around the best matches. Only matches with similar transformation are used which limits this approach to rigid matching. Leordeanu et al. [LH05] use thin-plate spline bending energy as as a measure of second-order similarity. In a different approach, Cheng et al. [CZM*10] identify matches for a semi-automatically generated query region. The method performs well even in the presence of strong occlusion and significant background clutter. The two last methods are not applicable for strongly non-rigid transformations. Yücer et al. [YJHS12] define a mapping between image regions as a deformation with bounded biharmonic weights [JBPS11] that are modified to adapt to the local image content. However, their region alignment approach is limited to regions that have a single best match.

Although they are used for raster images, the following two methods by Cho et al. [CLL09] and HaCohen et al. [HSGL11] are conceptually the most comparable works to our method. These are also part of the comparisons with our method in Section 7. Cho et al. [CLL09] starts by generating an initial set of local point correspondences in an image. These are then hierarchically clustered using a dissimilarity measure between pairs of matches that is based on the difference between their corresponding transforms. Their difference measure is unsuitable for large non-rigid deformations, however, and does not contain a notion of spatial neighborhood between correspondences. Thus, smoothness of the transformations over the query region cannot be properly defined and either local noise or only partially matched regions are the consequence. We will show in Section 3 that a concept of spatial neighborhood in the similarity measure is fundamental to the solution of this problem.

HaCohen et al. [HSGL11] start with a query image and target image and use the Generalized PatchMatch method [BSGF10] to find the best match for a patch around each pixel in the query image. Matched patches are then aggregated to consistent regions, where neighboring patches are required to have highly similar transforms. Since the Generalized PatchMatch method finds only a single best match for each pixel and does not exploit second-order similarities, the method favors local patch similarity over finding large matching region. For scenes with repeating elements, i.e., when there is not a single best match for each local patch, this causes problems.

3. Transformation Parameter Similarity

Our concept of similarity is based on 'smoothly' changing transformation parameters of first-order point correspondences over a region. In the discourse of this similarity concept we introduce three central terms: a *transformation space* is the space of rigid transformation parameters, *first-order correspondences* connect two points with local neighborhoods related by a rigid transformation and a *transformation function* models the non-rigid transformation between two shape regions. In the following, we give formal definitions of these terms.

The *transformation space* is the space of rigid transformation parameters. In our method, we focus on 2D similarity transformations $S(t_x, t_y, s, r)$, parametrized by x- and y-translation t_x and t_y, uniform scaling s and rotation r, resulting in a four-dimensional transformation space with coordinates $\tau = (t_x, t_y, s, r) \in \mathbb{T}$. Since the scaling of the individual dimensions is arbitrary (e.g. it depends on using ra-

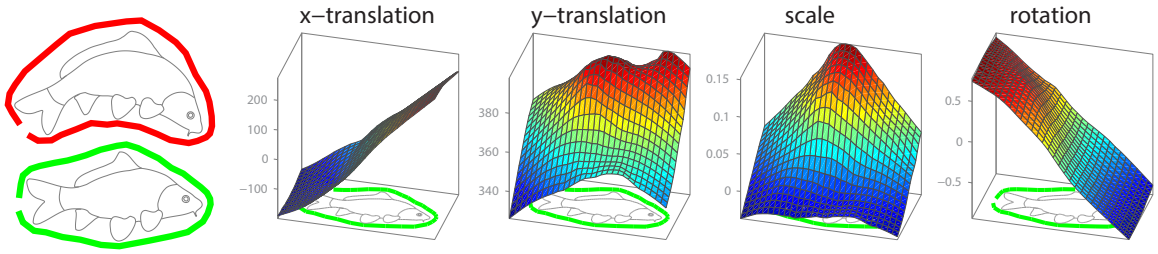

Figure 3: *Regions with high transformation parameter similarity are related by a smoothly changing transformation function. The green region on the left is related to the red region by the transformation function shown on the right. Each of the four transformation parameters of the function is shown in a separate plot. Note that there are no discontinuities or spikes in the gradient magnitude of the parameter functions.*

dians or degrees for the rotation), we normalize the dimensions of the transformation space. The following normalization factor is used throughout our method (unless noted otherwise): $(0.03b, 0.03b, 1.25, 45^\circ)$, where b is the length of the bounding box diagonal of the input shape. A translation by 3% of the bounding box diagonal is equivalent to a scaling by a factor of 1.25 and a rotation by 45°. Given parameters (t_x, t_y, s, r), the transformation of a point p is defined as $S(t_x, t_y, s, r) * p = S^L(t_x, t_y) * S^S(s) * S^R(r) * p$, where S^L, S^S and S^R are translation, scaling and rotation operations, respectively, and $*$ applies an operation to a point p.

A *first-order correspondence* (p, τ) matches two points in a vector image based on their local neighborhood, e.g. using a local descriptor. It is defined by the position of the first point p and the transformation parameters τ relating the local neighborhoods of the two points: $(p, \tau) = (p_x, p_y, t_x, t_y, s, r)$. The second point can be found as $\mathcal{S}(\tau) * p$. The set of all first-order correspondences of a shape is denoted $\boldsymbol{\mathcal{C}} \subset \mathbb{R}^2 \times \mathbb{T}$. Note that in the continous case, there are infinitely many first-order correspondences for any given shape. Due to the necessary discretization of the problem and associated numerical issues, only a finite noisy subset of $\boldsymbol{\mathcal{C}}$ can be found.

We model the transformation relating two shape regions $\mathbf{Q}, \mathbf{T} \subset \mathbb{R}^2$ with a *transformation function*:

$$f : \mathbf{Q} \to \mathbb{T} \tag{1}$$

The transformed region is $\mathbf{T} = \{S(f(p)) * p \mid p \in \mathbf{Q}\}$. A rigid transformation is described by a constant function, while all other functions describe non-rigid transformations. Each subset of correspondences $\mathcal{T} \subset \boldsymbol{\mathcal{C}}$ that contains one correspondence for each point in the region $\mathbf{Q}$ defines a transformation function that relates two shape regions with points having similar local neighborhoods. Figure 3 shows a transformation function relating the green region to the red region, i.e. transforming each point p in the green region with the transformation given by $f(p)$ yields the red region. The four dimensions of the function values are shown separately on the right. Since the red region is a smoothly bent version of the green region, the function does not exhibit discontinuities and the gradient magnitudes of its individual dimensions vary smoothly. Intuitively, transformation functions with low gradient magnitude in all parameters ensure that the transformed region is not distorted beyond recognition and retains local geometric detail, but still allow for strong non-rigid deformations over the whole region.

The Jacobian of the transformation function combines the gradients of the individual transformation parameters. The Frobenius norm of the Jacobian describes the magnitude of the combined gradients. More specifically, it is the norm of the vector of gradient magnitudes:

$$\|J_f\|_F = \left\| \left(\|\nabla f_{\tau_x}\|, \|\nabla f_{\tau_y}\|, \|\nabla f_{\tau_s}\|, \|\nabla f_{\tau_r}\| \right) \right\| \tag{2}$$

where J_f is the Jacobian of f, $\|\cdot\|_F$ is the Frobenius norm and ∇f_* are the gradients of the individual transformation parameters of S.

We define the transformation parameter similarity between two regions $\mathbf{Q}$ and $\mathbf{T}$ related by a transformation function f as the harmonic mean of the Jacobian magnitude $\|J_f\|_F$ over the entire function. The harmonic mean is used because a single large Jacobian magnitude in the transformation function is percieved as a strong discontinuity in the target region. Such a region is considered less similar than a region with large areas of average Jacobian magnitude.

Transformation functions that relate two shape regions are subsets of $\boldsymbol{\mathcal{C}}$. However, $\boldsymbol{\mathcal{C}}$ may contain multiple transformation functions, as well as correspondences that are not part of any transformation function. The goal of our method is to find subsets $\mathcal{T} \subset \boldsymbol{\mathcal{C}}$ that approximate functions with the properties described above. Each subset then describes a pair of regions with high transformation parameter similarity.

4. Algorithm Overview

The input to our algorithm is a model of 2D shapes defined by contour segments. Such segments can be defined as polylines, obtained for example by vectorizing the output of an edge detector like Canny. Broken contours are not overly problematic here. However, the polyline geometry should not be too noisy (if so it can be gracefully smoothed) as the local curvature is used by our local shape descriptor. The

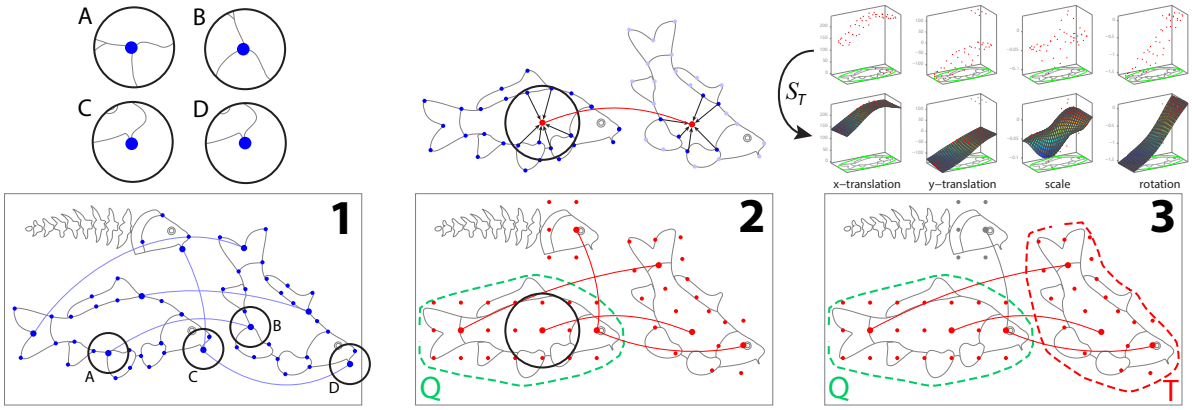

Figure 4: *The three main steps of our method. From left to right: 1) First-order correspondences $\mathcal{C}$ (blue dots) are computed. To avoid clutter, only some correspondences are shown with connecting lines. 2) Refined first-order correspondences $\mathcal{C}'$ (red dots) are computed by aggregating nearby unrefined correspondences. 3) Transformation functions $f_{\mathbf{T}}$ are found in transformation space, corresponding target regions* **T** *that are similar to the query region* **Q**.

second input is a user-provided 2D *query region*, defined by outlining or brushing on the canvas. The goal is to find all 2D shape regions that have a high similarity to the query region. Our method proceeds in three steps (cf. Figure 4 and Alogrithm 1).

In the first step, the first-order correspondences are computed using a local shape descriptor. We use a variant of Shape Contexts [BMP02], but any descriptor invariant to similarity transforms can be used as long as the resulting correspondences are reasonably accurate. Since this first step is interchangeable and just a minor contribution, we refer to the additional material for details.

In practice only a noisy subset of $\boldsymbol{\mathcal{C}}$ denoted $\mathcal{C}$ is found in the first step. We could find smooth transformation functions directly on this set of correspondences, but it is usually prohibitively large and may not cover all areas of the query region, degrading the stability of our method. To improve stability, we introduce an intermediate step that has three roles: it reduces the number of first-order correspondences to decrease computation time, generates correspondences that cover the query region more evenly and reduces noise. In this step, the query region is sampled regularly with a fixed number of points. For each of these points, we find stable correspondences by aggregating nearby noisy correspondences of $\mathcal{C}$ using the method of Mitra et al. [MGP06], as described in Section 5. The result is a smaller set of first-order correspondences $\mathcal{C}'$ that contains less noise and covers the query region more evenly.

In the final step, we search for sets of correspondences $\mathcal{T} \subset \mathcal{C}'$ that are smooth transformation functions (cf. Section 3). We use linkage clustering with weights based on approximate Jacobian magnitudes to identify these functions, as will be explained in Section 6. Each function corresponds to a region similar to the query region.

Algorithm 1: TPS Matching

Input : query region **Q**,
contour segments $\{P_j\}$

Output: sets of first-order correspondences $\{\mathcal{T}_k\}_{k=1}^{N}$
describing regions with similarities $\{TPS_k\}_{k=1}^{N}$

1 compute first-order correspondences $\mathcal{C}$ on contour segments $\{P_j\}$ using local shape descriptors
2 refine first-order correspondences $\mathcal{C}$ to get $\mathcal{C}'$ using Algorithm 2
3 the sets of correspondences $\{\mathcal{T}_k\}$ and their similarities $\{TPS_k\}$ are found as smooth transformation functions in $\mathcal{C}'$ using Algorithm 3

5. Improving First-Order Correspondences

The first-order correspondences found by the local shape descriptors are usually noisy and may not cover the query region evenly. This makes finding smooth transformation functions more difficult. Consequently, we refine the original set $\mathcal{C}$ of first-order correspondences into a new set $\mathcal{C}'$ having fewer elements with less noise that are evenly spread over the query region.

To create the new set of correspondences, we first determine the locations q of the refined correspondences (q, τ). The locations q are constructed by placing a fixed amount of points ($\sim$50 in our examples) regularly in the bounding box of the query region and only the points inside the query region are kept. We call these points *query points*.

Next, the transformation parameters τ for each query point are found. The method of Mitra et al. [MGP06] finds rigid region matches in a shape given a set of noisy first-order correspondences. We apply this method once for each query point q, using only the noisy correspondences near q as in-

put, as we will explain in detail below. This effectively gives the rigid matches of the local query point neighborhood. The rigid transformation parameter τ of each match is used to construct a refined correspondence (q,τ).

Given a query point q, noisy correspondences (p_i,τ_i') in the set $\mathcal{C}$ are weighted by the following function around q:

$$w_i = G(||p_i - q||)\, h_i' \frac{1}{d_i} \tag{3}$$

where G is a Gaussian with a radius r_s defined by the user. The Gaussian can be thought of as a smoothing kernel. Increasing the radius gives smoother results, since each refined first-order correspondence is based on a larger set of noisy first-order correspondences. We use a radius of $0.1b$ in all our examples, where b is the length of the bounding box diagonal of the input shape. Each noisy correspondence might have a confidence value h_i'. If no confidence values are available, a factor of one is used instead. d_i is the density of the first-order correspondences in $\mathcal{C}$ computed at each point p_i. This factor removes the bias that would otherwise be introduced due to different densities of first-order correspondences over the shape. A straightforward way to compute the density at a point p_i is to center G at p_i and accumulate $G(||p_i - p_j||)$ for all points $p_j \in \mathcal{C}$ with $i \neq j$.

With the weighted noisy correspondences near the query point q as input, we now proceed according to the method of Mitra et al. [MGP06]. Noisy correspondences with a weight smaller than one percent of the maximum weight are discarded. The transformation parameters τ_i' of the noisy correspondences weighted by w_i form a density distribution in transformation space $\mathbb{T}$. The dominant modes τ of this distribution correspond to transforms used by many noisy correspondences. Consequently, if such transforms are applied to the query point, it will be mapped to a neighborhood similar to the query region. Because no a-priori knowledge about the number of dominant modes is available, mean-shift clustering [CM02] with a kernel radius of one, due to the transformation space normalization, is used to find them.

The refined correspondences are the query points combined with each of their dominant modes:

$$\mathcal{C}' = \{(q,\tau)_j\} \tag{4}$$

where the index j runs over the dominant modes of all query points. Additionally, we use the magnitude h of the dominant modes as a confidence value for the refined correspondences. A higher magnitude means more noisy correspondences agree on the transformation parameters. The number of refined correspondences is the total number of dominant modes of all query points. The number of dominant modes for each query point depends on the complexity of the shape, as well as on the radius of the Gaussian in Equation 3. See Algorithm 2 for a summary of the refinement step.

Algorithm 2: Refine First-order Correspondences

Input : query region $\mathbf{Q}$,
noisy first-order correspondences $\mathcal{C}$
Output: refined first-order correspondences $\mathcal{C}'$

create query points $Q = \{q_k\}_{k=1}^{M}$ by sampling the query region $\mathbf{Q}$ in a regular grid
for $q_k \in Q$ **do**
  compute weights $\{w_i\}_{i=1}^{|\mathcal{C}|}$ using Equation 3
  compute set of transformation parameters $\{\tau\}_k$ as the dominant modes of the noisy correspondences $\mathcal{C}$ weighted by $\{w_i\}$ in transformation space $\mathbb{T}$
end
the refined correspondences $\mathcal{C}'$ are the pairs $\{(q,\tau)_j\}$ of all query points

6. Region Matching

We now proceed to the main step of our method: finding shape regions that have high transformation parameter similarity to the query region. A similar region is related to the query region by a transformation function with low Jacobian magnitude. Each refined first-order correspondence (q,τ) can be seen as a sample of a transformation function, where q is the function argument and τ the value. The set $\mathcal{C}'$ of refined first-order correspondences may contain samples for many transformation functions, as well as samples that are not part of any function. The goal in the final step is to identify subsets of first-order correspondences that approximate transformation functions with low Jacobian magnitude (cf. the additional material for an example). Each subset contains correspondences between the query region and a region similar to the query region.

To find these subsets, we first define a second-order similarity for each pair of first-order correspondences. The second-order similarity between two first-order correspondences $(q,\tau)_i$ and $(q,\tau)_j$ is based on a lower bound for the Jacobian magnitude that any function passing through both correspondences must have. A lower bound for the gradient magnitudes of the individual transformation parameters $\tau = (t_x, t_y, s, r)$ is the vector of difference quotients:

$$\mathbf{a}_{ij} = \frac{(t_x,t_y,s,r)_i - (t_x,t_y,s,r)_j}{\|q_i - q_j\|} \tag{5}$$

Since the Jacobian magnitude is the norm of the vector of gradient magnitudes (cf. Equation 2), the norm of $\mathbf{a}_{ij}$ is a lower bound for the Jacobian magnitude. We define the second-order similarity as the Gaussian of this lower bound:

$$a_{ij} = G(\|\mathbf{a}_{ij}\|) = G\left(\frac{\|\tau_i - \tau_j\|}{\|q_i - q_j\|}\right) \tag{6}$$

where a_{ij} is the second-order similarity between correspondence i and correspondence j. Note that the transformation

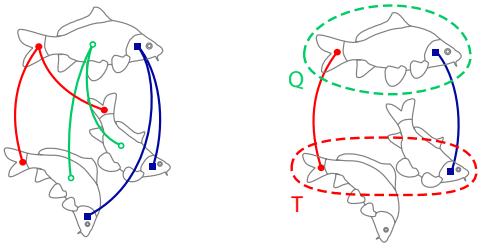

Figure 5: *Effects of an overly large neighborhood radius. Three fish are related by the correspondences shown on the left. The tail of the lower fish has similar transformation parameters as the head of fish in the middle. An overly large neighborhood radius would allow our method to merge correspondences from distant scene parts, while the center correspondences (green) are ignored. This causes an incorrect matching, as shown on the right.*

parameters τ are normalized, as explained in Section 3, otherwise the difference quotient would depend on the scale of the individual parameters. The radius of the Gaussian is set to one. Changing the radius would have the same effect as changing the normalization factors.

Given the second-order similarity for all pairs of correspondences, we find subsets of correspondences that approximate functions with low Jacobian magnitude. Hierarchical average linkage clustering with linkage weights based on the second-order similarity is used to find these subsets. We start with each correspondence in a separate set and iteratively merge neighboring sets having elements with high linkage weights. The linkage weights between correspondences $(q,\tau)_i$ and $(q,\tau)_j$ are defined as:

$$l_{ij} = a_{ij}\, h_i h_j n_{ij} \tag{7}$$

where h are confidence values for correspondences i and j, effectively favoring sets of correspondences with high confidence. The factor n_{ij} is a neighborhood weight that restricts merging operations to neighboring sets of correspondences:

$$n_{ij} = G(\|q_i - q_j\|) \tag{8}$$

G is a Gaussian with a radius r_n that controls the trade-off between the proximity of sets and their second-order similarity. A low radius will only allow sets of correspondences to be merged that have adjacent query points. This precludes finding shape regions with occlusions or regions missing correspondences due to noise or bad matching. Setting the radius too high might result in correspondences from distant parts of the query region being merged, ignoring the correspondences in between, as shown in Figure 5. Unless noted otherwise, we use a value of $0.003b$ in all our scenes, where b is the length of the bounding box diagonal of the input shape.

In each iteration, the two sets of correspondences $\mathcal{A}$ and $\mathcal{B} \subset \mathcal{C}'$ with the highest average of the linkage weights between their elements are merged. Each set contains samples of a transformation function that has low Jacobian magnitude when reconstructed from the samples. In some cases, such as branching structures that are matched to a non-branching query region, two sets of correspondences may get merged that have a large overlap in their query points. For example, two branches and the trunk of a tree may get matched to a query region that contains only a trunk and a single branch, if the trunk splits smoothly into the two branches. To pick only a single branch, we introduce a factor that encourages regions that contain exactly one correspondence of each query point q:

$$d_{\mathcal{AB}} = 2^{N^s_{\mathcal{AB}} - N^d_{\mathcal{AB}}} \tag{9}$$

where $N^d_{\mathcal{AB}}$ is the number of query points that are present in the correspondences of both sets and $N^s_{\mathcal{AB}}$ the number of query points present in only one of the sets. Two sets having a large overlap in the domain of the transformation function have a low fold-over factor. The final linkage weight between two sets of correspondences $\mathcal{A}$ and $\mathcal{B}$ is defined as:

$$l_{\mathcal{AB}} = \frac{\sum_{(i,j)_{\mathcal{AB}}} l_{ij} n_{ij}}{\sum_{(i,j)_{\mathcal{AB}}} n_{ij}} d_{\mathcal{AB}} \tag{10}$$

where $(i,j)_{\mathcal{AB}} = \{(i,j) \mid (q,\tau)_i \in \mathcal{A} \;\; (q,\tau)_j \in \mathcal{B}\}$. The algorithm stops merging the sets with highest linkage weight when this weight falls below a threshold. In our implementation we use one percent of the largest linkage weight.

Each resulting set $\mathcal{T} \subset \mathcal{C}'$ defines a transformation function $f_{\mathbf{T}}$ with low Jacobian magnitude that describes the transformation between the query region $\mathbf{Q}$ and a target region $\mathbf{T}$. Finally, the transformation parameter similarity of the two shape regions $\mathbf{Q}$ and $\mathbf{T}$ is approximated by the weighted harmonic mean (cf. Section 3) of the linkage weights between all pairs of correspondences in $\mathcal{T}$:

$$TPS_{\mathbf{QT}} = |\mathcal{T}| \left(\frac{\sum_{(i,j)_{\mathcal{T}}} l_{ij}^{-1} n_{ij}}{\sum_{(i,j)_{\mathcal{T}}} n_{ij}} \right)^{-1} \tag{11}$$

where $(i,j)_{\mathcal{T}}$ are the indices of the pairs of correspondences. See Algorithm 3 for a summary of the region matching step.

7. Results and Discussion

We implemented the described method in Matlab and tested it on a PC with 16 GB memory and a 3.4 GHz quad-core CPU. Key parts of the correspondence refinement step and the clustering step were implemented in CUDA or C++. The first step (cf. Section 4) creates first-order correspondences using local shape descriptors (cf. additional material). This step is computed once for each shape in a preprocess and is independent of the query region. Steps two and three are computed after a user selects a query region and depend mainly on the number of query points (usually $30-50$ in our examples). The linkage clustering described in Section 6 has a worst-case complexity of $O(N^3)$ in the number of refined first-order correspondences, but the average case has much lower complexity. Additionally, we discard pairs of refined

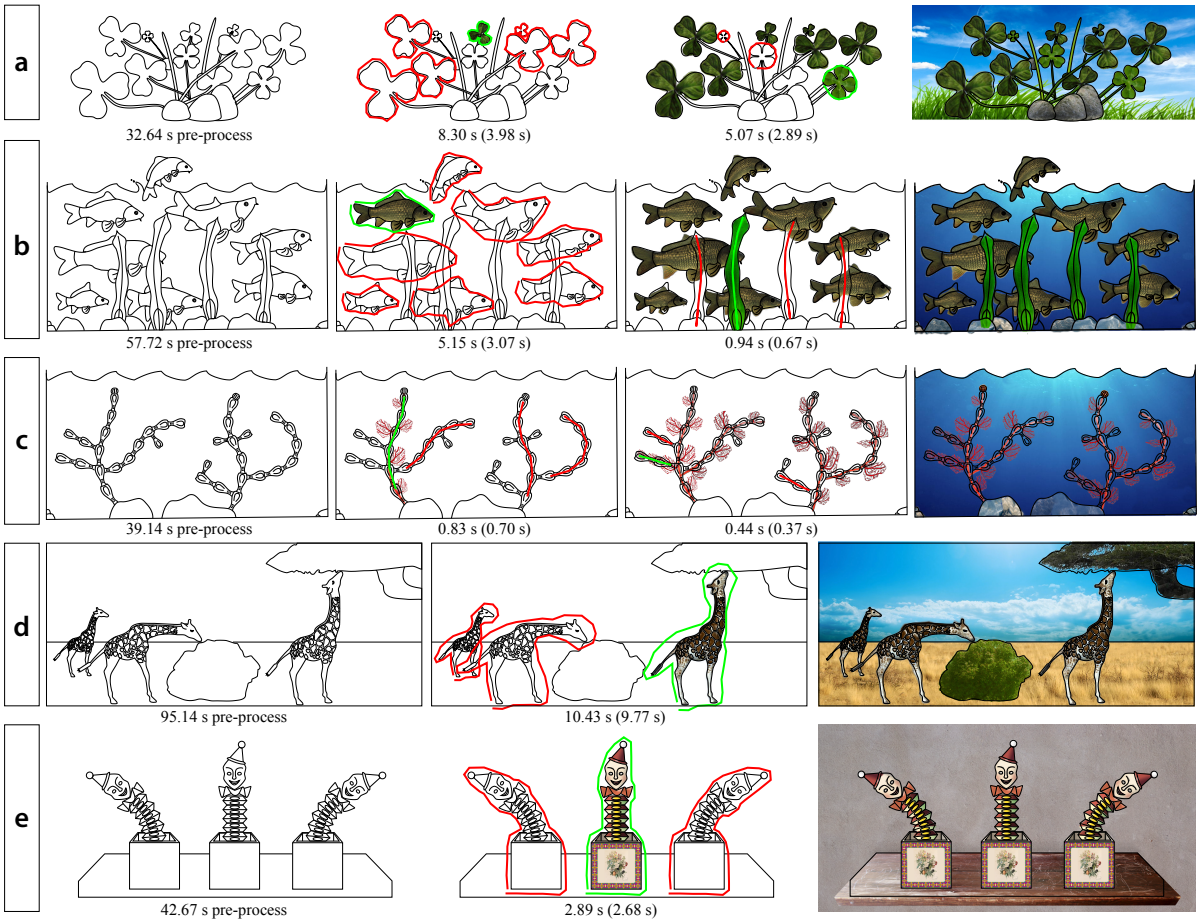

Figure 6: *Results of our method on five test shapes. The leftmost image shows the input shape. The next image shows one operation where region matches (red) are found for a user-defined query region (green). The texture from the query region is transferred to each such target region, as shown in the following image. The final result is shown in the last image with added background to give an impression of how an artist-created result might look like when using our method. Timings are provided below each image (see Section 7 for details).*

first-order correspondences with very low confidence (below one percent of the maximum confidence) before clustering. The computationally most intensive case for our method is a shape where all pairs of points have the same similarity, e.g., a circle. Since the stopping criterion of the clustering step would never be met, the clustering algorithm would need run through N iterations to merge all correspondences, searching for the best among $(N-k)^2$ correspondence pairs in step k.

The method is demonstrated on several vector images shown in Figures 6 and 7. The images are created by automatic contour finding from RGB-images with subsequent manual noise removal (h), manual drawing (c and i) or a combination of both (a,b,d,e,f,g). In each example, we perform 1-3 operations. An operation includes finding all red target regions matching the green query region with a TPS above a given threshold (which is selected interactively with a slider after the region matching step has finished) and then transferring the texture shown in the query region to each target region to visualize matching accuracy. Each texture has an alpha-channel and is added on top of the existing textures. The background is only included to give a better impression of how the final result of an artist-created image using our method might look like. The only input to our region matching method are the shape contours in the leftmost column.

In Figure 1, the small snake is related to the big snake by a transformation function with changing scale, angle and translation parameters. The skin and the pattern of the snake are transferred in two operations. Figures 6a and 7g show similar images with a significant amount of clutter. Nevertheless, it is possible to detect small objects like the leaf in Figure 7h or the small clovers in Figure 6a. In Figure 6b, all fish in the image have different proportions and additionally, some are bent. The contours of some fish have large gaps due to occluding sea weed. For the sea weed we use query points on a stroke instead of a region. The texture is transferred by extrapolating the texture coordinates of the matched (red)

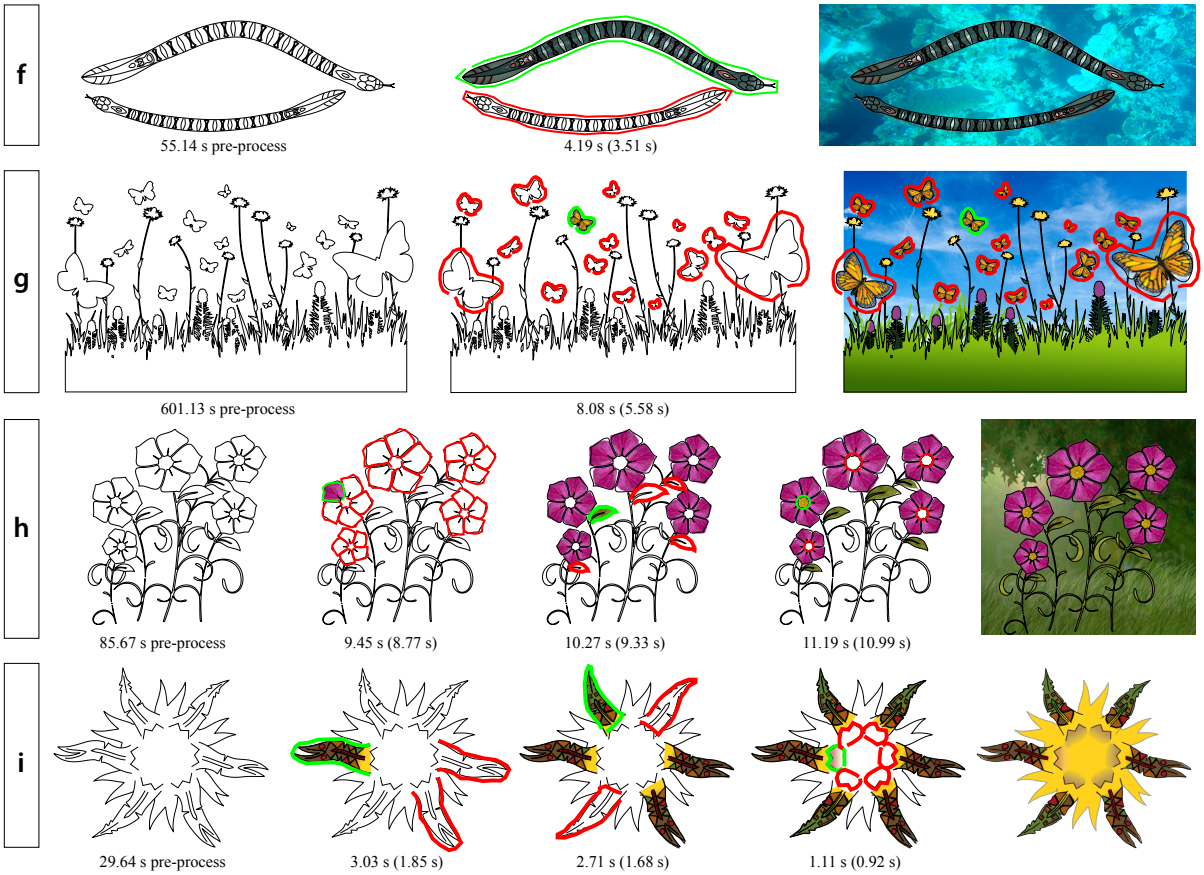

Figure 7: *Addtional results. The leftmost image shows the input shape. The next image shows one operation where region matches (red) are found for a user-defined query region (green). The texture from the query region is transferred to each such target region, as shown in the following image. The final result is shown in the last image with added background to give an impression of how an artist-created result might look like when using our method. Timings are provided below each image (see Section 7 for details).*

strokes. Figure 6c shows branching corals. Due to the thin structure of the corals, strokes are used here instead of regions. In 6d, giraffes in different poses are matched by our method. Figure 6e shows man-made objects with different amounts of bending (also note that the hat of the left harlekin is mirrored), while in 7f two snakes with a fine skin pattern are matched. In Figure 7g, eight different types of butterflies are matched in an image with a large amount of background clutter. The scale difference between butterflies has a factor of up to ten. Finally, Figure 7i shows a stylized sun with non-rigidly transformed rays being constructed in three operations (excluding the solid-colored background).

Detailed timings for each operation are shown in Figures 6 and 7 below the image of the corresponding operation. The time in parenthesis is for the refinement step, the time outside the parenthesis for the entire operation. Times for the first step of our method, computed as a pre-process, are shown below the input image. The largest factor for the computational demand is the number of refined correspondences, which mainly depend on scene clutter. For this reason, the operations in the flowers scene are computationally most expensive, followed by the butterflies and the clovers scene. In the giraffes example we used more query points than in the other examples to better capture the thin legs, making the operation more expensive.

Although the refinement step improves first-order correspondences considerably, the inlier percentage is still relatively small. Table 1 shows the inlier percentage for the first operation in all example images. In Figure 8 we show the distribution of refined first-order correspondences in the snakes image (Figure 1) in transformation space. The region matching step correctly finds the matching snake in the presence of a significant amount of outliers. Note how the matched region consists of high-confidence as well as low-confidence correspondences, precluding simple thresholding. See the supplementary material for additional and more detailed visualizations.

We compare our method to the L_2-Estimator

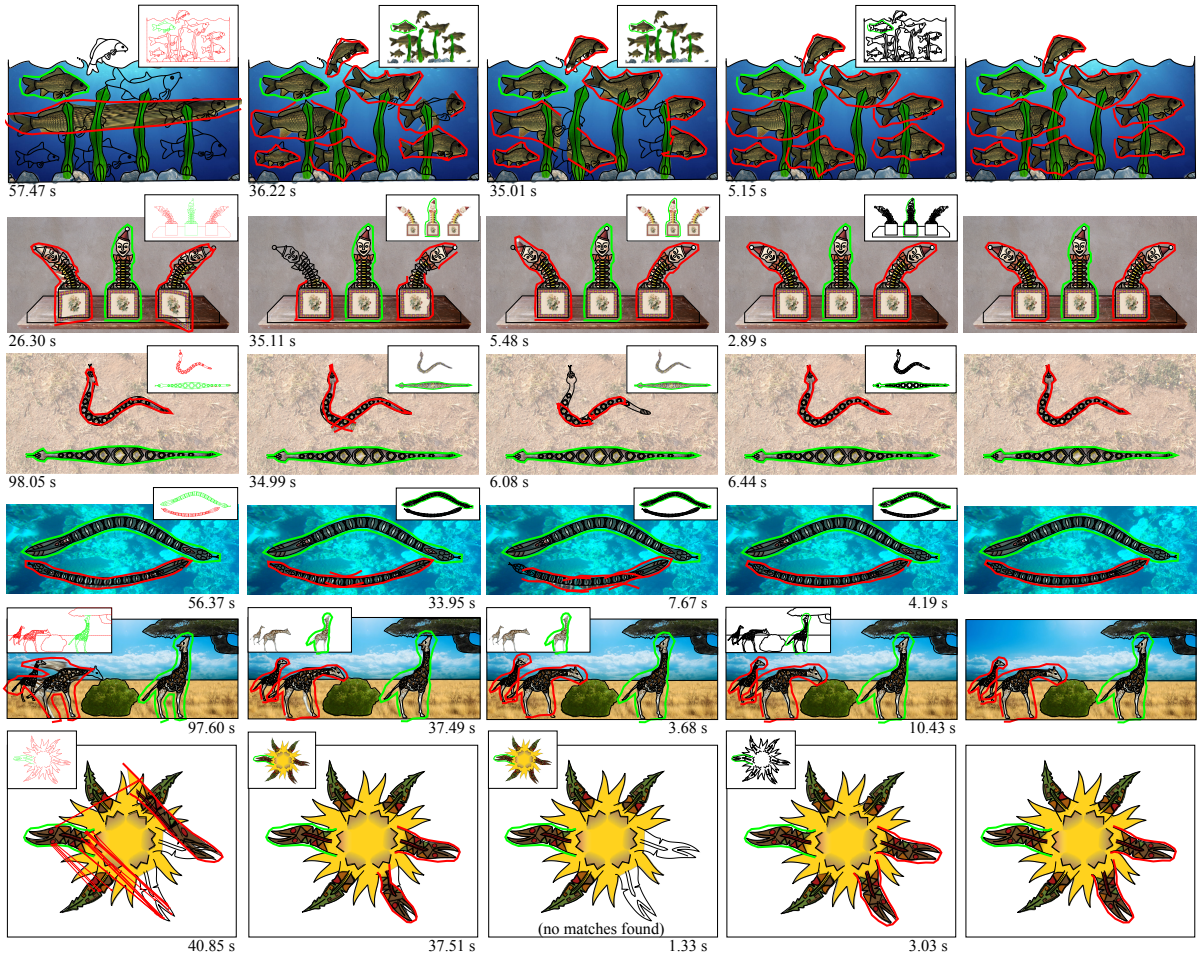

Figure 9: *Comparison to the L_2-Estimator (L2E) [MZT*13], Agglomerative Correspondence Clustering (ACC) [CLL09] and the Non-Rigid Dense Correspondence (NRDC) [HSGL11] methods. The red target regions are matches found for the green source region. A texture is transferred from source to target regions to visualize the matching accuracy. The input to the individual methods is shown as inset. A ground truth created by manually placing key correspondences is provided as reference. Note that L2E has problems with multiple matching regions, ACC has problems with regions related by strongly non-rigid transformations and NRDC has problem with occlusions, repeating patterns and regions without detailed texture. Our method can handle multiple matches, occlusions, strong non-rigid transformations and repeating patterns making it possible to correctly detect all similar regions.*

(L2E) [MZT*13] Agglomerative Correspondence Clustering (ACC) [CLL09] and the Non-Rigid Dense Correspondence (NRDC) method [HSGL11], three state-of-the-art methods for non-rigid region matching. For all three methods, we optimize the parameters for each image separately to provide the best matching. The correspondences found by these methods may not cover the entire query region, making it necessary to extrapolate the missing parts to establish a dense correspondence. For each image, we choose either Locally Linear Embedding [RS00], thin-plate splines or a Gaussian-weighted average of the transformation parameters for the extrapolation, whichever gives the best results. In our method, the correspondence refinement step makes a separate extrapolation unnecessary. To validate the results, we use a ground truth generated by manually placing a sparse set of correspondences at key locations of the shape (cf. Figure 9d). Since the L2E method works on point sets, we use evenly-spaced samples on the contour segments as input. The ACC and NRDC methods both require images as input. We transfer the texture in the query region to all target regions using the ground truth and use the resulting image as input. The area inside the query region is used as source image and the area outside the query region as target image. We also tried using rasterized contours as input, but using textures gave better results for all scenes. The input for each method is shown in the insets of Figure 9.

Algorithm 3: Find Smooth Transform. Functions in $\mathcal{C}'$

Input : refined first-order correspondences $\mathcal{C}'$
Output: sets of first-order correspondences $\{\mathcal{T}_k\}_{k=1}^{N}$ describing regions with similarities $\{TPS_k\}_{k=1}^{N}$

1 initialize sets of correspondences $\{\mathcal{T}_k\}$ as $\{\mathcal{T}_k = \{(q,\tau)_k \in \mathcal{C}'\} \mid k = 1 \ldots |\mathcal{C}'|\}$
2 compute linkage weights between all pairs $(\mathcal{T}_i, \mathcal{T}_j)$ using Equation 10
3 **while** *highest linkage weight is above threshold* **do**
4 merge correspondence sets $\mathcal{T}_{i^*}$ and $\mathcal{T}_{j^*}$ having the highest linkage weight: $\{\mathcal{T}_k\} = \left(\{\mathcal{T}_k\} \cup \{\mathcal{T}_{i^*} \cup \mathcal{T}_{j^*}\}\right) \setminus \{\mathcal{T}_{i^*}, \mathcal{T}_{j^*}\}$
5 update linkage weights for the merged correspondence set
6 **end**
7 compute similarities $\{TPS_k\}$ for each set in $\{\mathcal{T}_k\}$ using Equation 11

Result	corr. count	inlier %	weighted inlier %
1	2551	2.47	4.18
6a	6648	5.91	17.95
6b	7810	4.34	19.65
6c	278	7.91	7.39
6d	5664	2.31	23.98
6e	2028	5.42	16.60
7f	2999	1.87	4.62
7g	5668	12.35	29.69
7h	1911	55.78	88.17
7i	4005	2.42	14.02

Table 1: *Inlier percentage for the first operation in all examples shown in Figures 1, 6 and 7. Columns from left to right: total number of refined first-order correspondences, percentage of inliers thereof, percentage of inliers weighted by their confidence (see Section 5).*

The L2E method alternates between improving the transformation and the correspondence between the source and target point sets. The transformation is modeled as a function restricted to lie in a specific reproducing kernel Hilbert space and is estimated using their robust L_2 estimator. Correspondences are updated in each iteration using the Hungarian method to connect points with the most similar shape contexts. This method is designed for two point sets that have a single best match and exhibits two main problems in the presence of multiple best matches. First and most importantly, the method might get stuck in a local minimum if there are multiple basins of attraction, since one subset of the points might be initialized in one basin and a second subset in a different basin. The result of this behavior can be observed in rows 1, 5 and 6 of Figure 9a. Note that we only show one matched region in the first and last row to avoid clutter, since the other regions are distributed over a larger number of attractors resulting in an even larger distortion. In

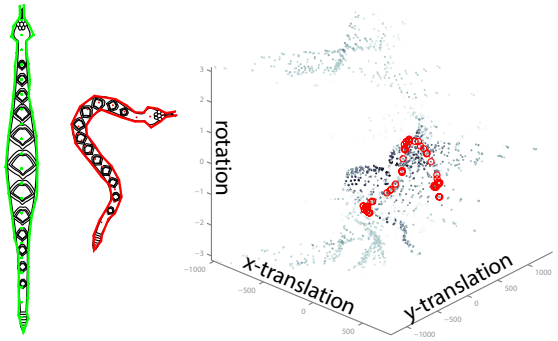

Figure 8: *Refined first-order correspondences of the snakes image (Figure 1) in transformation space. Only the rotation and translation dimensions are shown. Dark correspondences have high confidence, light correspondences have low confidence. Inliers corresponding to the red target region are encircled in red.*

the last row we additionally show the correspondences found for the second matched region as red lines. The second problem is that making the shape contexts scale and translation invariant is not trivial anymore if the matching regions are not centered in their point set. The authors use the average distance of the point set to the shape context center as scale and the direction of the shape context center to the point set center of mass as direction of a shape context. These approximations are only valid if both the source and target regions have approximately the same scale and are centered in their point sets. Using the contour normal and curvature information, as we do, is not possible, since the source points are transformed in each iteration and the contour information is invalidated. Badly rotated shape contexts result in the mismatched head of the harlekin in row 2 of Figure 9a.

The ACC method clusters first-order correspondences based on the absolute difference between their transformations. This means that sparse correspondences on a strongly non-rigid region, like on the 'neck' of the harlekin or the body of the snake in Figure 9b, are considered dissimilar and the matched region breaks apart. Additionally, the correspondences found by the method are sparse and only cover part of the query region. In contrast, our method uses the *gradient* of transformation parameters instead of the absolute difference, making it possible to correctly identify regions defined by a sparse set of first-order correspondences (cf. Figure 9c).

The NRDC method constructs first-order correspondences by finding a single best match for each $12x12$ pixel patch in an image using Generalized Patch Match [BSGF10] and then grows regions by merging adjacent correspondences based on the absolute difference of their transformations. Since the correspondences densely cover the query region, using the absolute difference does not create the same problems as in the ACC method. However, only merging adjacent correspondences precludes the handling of occlu-

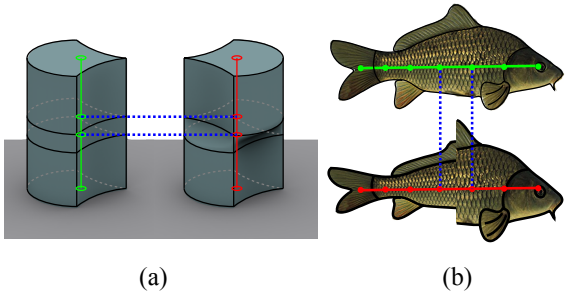

Figure 10: *Projection error used with transformations having many free parameters (note that this figure is only meant to illustrate a potential problem, we did not implement 3D transformations or non-uniform scaling). In both examples, the projection error between the two correspondences shown as blue dotted lines would be zero, even though their transformation is clearly different. In (a) the green and red strokes at the center axis of the two extruded shapes would be matched even though the second shape has a large discontinuity in its rotation around the center axis and in (b) the strokes inside the fish would be matched, even though the bottom fish has a large discontinuity in its vertical scaling.*

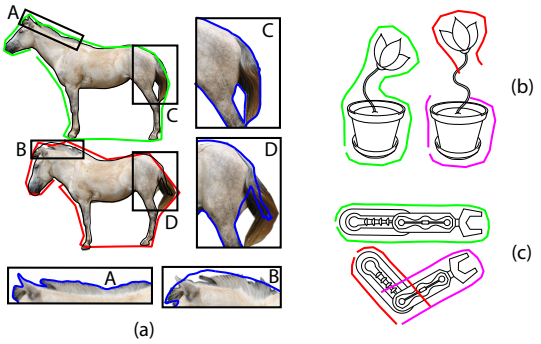

Figure 11: *Limitations of our method. Example (a) shows the result of matching two shape regions with partially dissimilar geometry, as shown in the insets. The matches in the dissimilar parts are not accurate, but similar parts of the shape are still matched correctly. In example (b), the stems of the flowers do not have any distinctive features in common. Consequently, no first-order correspondences are found for the stems and our method can only match the flower blossom and pot. Example (c) shows two regions that are related by a discontinuous transformation function. Our method only matches its continuous subsets.*

sions, as shown in Figure 9c, first row. Also, the single best match for each image patch is found based on local information only. Each element of a repeating pattern, such as the pattern on the snake skin (cf. Figure 9c, third and fourth row) is locally similar to all other elements, causing mismatches that can only be resolved when using global information, i.e. the position of the pattern element on the snake. Our method finds several matches for each query point and chooses a subset of these matches based on global information, correctly matching regions with repeating patterns. Finally, since NRDC densely computes correspondences for each pixel based on a relatively small neighborhood, it only works on regions having detailed texture. Less detailed textures (see bottom row of Figure 9c) can not be handled.

Relation to the Projection Error The projection error is a standard method of measuring second-order similarity. Our method can be adapted to use the projection error by changing the Jacobian magnitude in Equation 5 to the projection error as defined in Equation 1 of Cho et al. [CLL09] or Equation 1 of HaCohen et al. [HSGL11]. However, we argue that the TPS has two main advantages over the projection error when computing the similarity between two first-order correspondences:

First, the projection error is computed in the 2D space of positions, whereas transformations usually have a much larger number of free parameters (four in our case), and information is necessarily lost in the projection of the transformation parameters into the 2D space. Consequently, the distance in the 2D space does not always correlate with the distance of the transformation parameters. Figure 12a shows two snakes with clearly different scales, but the second-order similarity based on projection error does not have enough information to distinguish between the two scales and incorrectly merges them. This problem gets worse as the number of transformation parameters increases. For example, when using non-uniform scaling or when extending the method to 3D input, completely different transformations might have zero projection error. We illustrate this problem in Figure 10. In contrast, since our method directly compares the transformation parameters, no information is lost and the scale difference in Figure 12b is correctly detected.

Second, TPS is gradient-based, whereas the projection error uses an absolute distance, making it less suitable to handle distributions of query points with varying densities. The projection error gets smaller as the density of correspondences in a region increases, even though the transformation of the region remains the same. Figure 12c-f shows two butterflies that are related by a non-rigid transformation. To accurately match the small feelers, they have to be sampled more densely. Consequently the projection error in the feelers is smaller than in the remainder of the body, causing either incorrectly merged correspondences in the feelers due to a small projection error (Figure 12d), or failure to merge correspondences in the body due to a large projection error (Figure 12e). This depends on the value of the normalization factor described in Section 3. The TPS is independent of sampling density, as it uses the gradient over the query region and can find a good match for both feelers and body using the standard value of the normalization factor (see Figure 12f).

Possible Extension to 3D Geometry One interesting direction for future work is the extension of our method to 3D geometry. In this case, to find first-order correspondences, we would need 3D surface descriptors such as Spin Im-

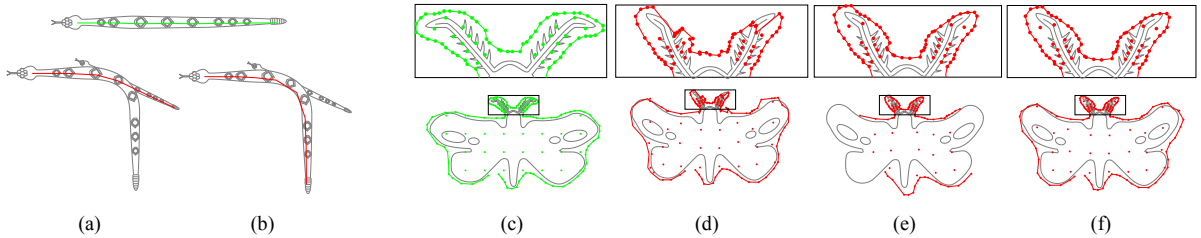

Figure 12: *Two problems when using a second-order similarity based on the projection error. In (a) and (b), a snake is matched to two snakes having different scales. The similarity based on the projection error (a) fails to separate the two snakes, whereas our similarity measure (b) can correctly separate the snakes based on the scaling parameter of their transformations. The butterfly in (c) is matched non-rigidly to the butterfly shown in (d)-(f). The similarity based on the projection error either incorrectly merges correspondences on the feelers (d) or fails to match the body (e), depending on parmeter settings. Our method (f) can correctly match body and feelers using standard parameters.*

ages [Joh97], Shape-DNA [RWP06] or Heat Kernel Signatures [SOG09] instead of our 2D descriptors. Using 3D descriptors might reduce false correspondences, since the local neighborhood of a point on a 3D surface is typically more discriminative than the local neighborhood on a 2D surface. Additionally, the transformation space $\mathbb{T}$ would be higher-dimensional, but this would pose no problem, as all employed techniques are well suited for high-dimensional data and have a linear dependence on the number of transformation parameters, both in execution time and memory consumption. A larger number of dimension in $\mathbb{T}$ might even be beneficial, since the correspondences in $\mathbb{T}$ are sparser, effectively reducing clutter in the transformation space. Intuitively, the transformation parameters should be smooth in the 3D case as well, however this would need to be shown empirically on a number of examples.

An interesting design choice would be the placement of query points. Query points could be placed either in the volume or on the surface of a 3D object. The first option would be analogous to our current method and would treat 3D regions as solid objects with a volume that cannot have strong discontinuities in its transformation. The second option would treat regions as 2D hulls and not constrain their volumes in any way, allowing for different kinds of transformations, such as a smooth and detail-preserving unfolding of open meshes. When using the second option, we would need to change the neighborhood weight in Equation 8 to approximate the geodesic distance on the object surface. The differences to isometric shape matching discussed in Section 2 would also hold for a 3D extension of our method, such as the ability of our method to handle smooth changes in scale or more generally transformations that do not preserve pairwise distances between surface points.

The main challenge in implementing the volumetric version of our method might be performance, since the number of query points in the volume might get quite large. Another challenge is finding good local descriptors that provide reliable transformation parameters.

Limitations In the remainder of this section, we will discuss limitations of our method. First, due to our definition of similarity, our method can only match shape regions related by a smooth transformation function. Regions like joints have large gradients in the transformation parameters and can therefore not be matched, as illustrated in Figure 11c. Consequently, our method is better suited for organic shapes or shapes that deform smoothly. Note that some of these smooth deformations like a gradual change in scale (cf. Figure 1) cannot be handled by other popular similarity measures such as isometry (as discussed in Section 2).

Second, similar regions need to have distinctive local features that can be matched by local descriptors. Otherwise first-order correspondences cannot be found. A failure case is shown in Figure 11b. The stems of the two flowers cannot be matched correctly, because locally, all parts of the two stems have equally similar geometry. Another reason for missing correspondences is dissimilar geometry. Figure 11a shows two horses with geometry that is mostly similar, except for the neck and tail as shown in the insets. These parts are missing first-order correspondences. In our implementation we try to infer missing correspondences from parts that have been matched correctly, but shape geometry in the missing parts cannot be taken into account.

8. Conclusion

We have demonstrated that partial non-rigid shape matches can be found in 2D shapes by searching for sets of correspondences with smoothly changing scaling, rotation and translation parameters. We call the similarity measure resulting from this notion ‘Transformation Parameter Similarity’. Our method can match non-rigid objects that are similar under this measure and establish a dense mapping between pairs of matching objects, even if the regions are only given as cluttered line segments. We have shown that this mapping can be used to transfer a texture between two regions. Additional operations like simultaneous deformation of similar shapes are possible using our method and we would like to

explore these applications in future work. Another interesting future direction is to expand the definition of similarity to include branching shapes, which are represented by branching surfaces in transformation space. This would allow texturing branching structures like trees for example.

References

[ASP*05] ANGUELOV D., SRINIVASAN P., PANG H.-C., KOLLER D., THRUN S., DAVIS J.: The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces. *Adv. in Neural Inf. Proc. Systems 17* (2005), 33–40. 2

[BBK06] BRONSTEIN A. M., BRONSTEIN M. M., KIMMEL R.: Generalized multidimensional scaling. *Proceedings of the National Academy of Science* (2006), 1168–1172. 2

[BMP02] BELONGIE S., MALIK J., PUZICHA J.: Shape matching and object recognition using shape contexts. *IEEE PAMI 24* (2002), 509–522. 1, 2, 5

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D. B.: Patchmatch. *ACM TOG 28* (2009), 24:1–24:11. 3

[BSGF10] BARNES C., SHECHTMAN E., GOLDMAN D. B., FINKELSTEIN A.: The generalized patchmatch correspondence algorithm. ECCV'10, pp. 29–43. 3, 11

[BWM*11] BERNER A., WAND M., MITRA N. J., MEWES D., SEIDEL H.-P.: Shape analysis with subspace symmetries. *CGF 30*, 2 (2011). Eurographics. 2

[CLL09] CHO M., LEE J., LEE K. M.: Feature correspondence and deformable object matching via agglomerative correspondence clustering. In *ICCV* (2009), pp. 1280–1287. 3, 10, 12

[CM02] COMANICIU D., MEER P.: Mean shift: a robust approach toward feature space analysis. *IEEE PAMI 24*, 5 (2002), 603–619. 6

[CR03] CHUI H., RANGARAJAN A.: A new point matching algorithm for non-rigid registration. *Comput. Vis. Image Underst. 89*, 2-3 (2003), 114–141. 3

[CZ08] CHANG W., ZWICKER M.: Automatic registration for articulated shapes. In *SGP* (2008), pp. 1459–1468. 2

[CZM*10] CHENG M.-M., ZHANG F.-L., MITRA N. J., HUANG X., HU S.-M.: Repfinder: finding approximately repeated scene elements for image editing. *ACM TOG 29* (2010), 83:1–83:8. 3

[EK03] ELAD A., KIMMEL R.: On bending invariant signatures for surfaces. *IEEE PAMI 25*, 10 (2003), 1285 – 1295. 2

[FJS10] FERRARI V., JURIE F., SCHMID C.: From images to shape models for object detection. *IJCV 87* (2010), 284–303. 3

[GCO06] GAL R., COHEN-OR D.: Salient geometric features for partial shape matching and similarity. *ACM TOG 25*, 1 (2006), 130–150. 2

[HSGL11] HACOHEN Y., SHECHTMAN E., GOLDMAN D. B., LISCHINSKI D.: Non-rigid dense correspondence with applications for image enhancement. *ACM TOG 30*, 4 (2011), 70:1–70:9. 3, 10, 12

[JBPS11] JACOBSON A., BARAN I., POPOVIĆ J., SORKINE O.: Bounded biharmonic weights for real-time deformation. *ACM TOG 30*, 4 (2011), 78:1–78:8. 3

[Joh97] JOHNSON A.: *Spin-Images: A Representation for 3-D Surface Matching*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, August 1997. 13

[LH05] LEORDEANU M., HEBERT M.: A spectral technique for correspondence problems using pairwise constraints. In *ICCV* (2005), vol. 2, pp. 1482–1489 Vol. 2. 3

[LLA*09] LU C., LATECKI L. J., ADLURU N., YANG X., LING H.: Shape guided contour grouping with particle filters. In *ICCV* (2009), IEEE, pp. 2288–2295. 3

[Low04] LOWE D. G.: Distinctive image features from scale-invariant keypoints. *Int. J. C. Vision 60* (2004), 91–110. 1, 2

[MGP06] MITRA N. J., GUIBAS L. J., PAULY M.: Partial and approximate symmetry detection for 3d geometry. *ACM TOG 25* (2006), 560–568. 2, 3, 5, 6

[MGP07] MITRA N. J., GUIBAS L., PAULY M.: Symmetrization. *ACM TOG 26*, 3 (2007), 1–8. 2

[ML11] MA T., LATECKI L.: From partial shape matching through local deformation to robust global shape similarity for object detection. In *CVPR* (2011), pp. 1441 –1448. 3

[MS05] MÉMOLI F., SAPIRO G.: A theoretical and computational framework for isometry invariant recognition of point cloud data. *Found. Comput. Math. 5* (July 2005), 313–347. 2

[MZT*13] MA J., ZHAO J., TIAN J., TU Z., YUILLE A.: Robust estimation of nonrigid transformation for point set registration. 2147–2154. 3, 10

[PBB*13] POKRASS J., BRONSTEIN A. M., BRONSTEIN M. M., SPRECHMANN P., SAPIRO G.: Sparse modeling of intrinsic correspondences. *CGF 32*, 2pt4 (2013), 459–468. 2

[RCB97] RANGARAJAN A., CHUI H., BOOKSTEIN F.: The soft-assign procrustes matching algorithm. In *Information Processing in Medical Imaging*, vol. 1230 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1997, pp. 29–42. 3

[RS00] ROWEIS S. T., SAUL L. K.: Nonlinear dimensionality reduction by locally linear embedding. *Science 290* (2000), 2323–2326. 10

[RWP06] REUTER M., WOLTER F.-E., PEINECKE N.: Laplace-beltrami spectra as "shape-dna" of surfaces and solids. *Computer-Aided Design 38*, 4 (2006), 342–366. 2, 13

[SOG09] SUN J., OVSJANIKOV M., GUIBAS L.: A concise and provably informative multi-scale signature based on heat diffusion. In *SGP* (2009), pp. 1383–1392. 2, 13

[SZS10] SRINIVASAN P., ZHU Q., SHI J.: Many-to-one contour matching for describing and discriminating object shape. In *IEEE CVPR* (2010), pp. 1673–1680. 3

[YJHS12] YÜCER K., JACOBSON A., HORNUNG A., SORKINE O.: Transfusive image manipulation. *ACM TOG 31*, 6 (2012), 176:1–176:9. 3

[ZD06] ZHENG Y., DOERMANN D.: Robust point matching for nonrigid shapes by preserving local neighborhood structures. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 28*, 4 (2006), 643–649. 3

[ZSCO*08] ZHANG H., SHEFFER A., COHEN-OR D., ZHOU Q., VAN KAICK O., TAGLIASACCHI A.: Deformation-driven shape correspondence. *SGP 27*, 5 (2008), 1431–1439. 2