Eurographics Symposium on Geometry Processing 2008
Pierre Alliez and Szymon Rusinkiewicz
(Guest Editors)

Volume 27 (2008), Number 5

Automatic Registration for Articulated Shapes

Will Chang Matthias Zwicker

University of California, San Diego

Abstract

We present an unsupervised algorithm for aligning a pair of shapes in the presence of significant articulated motion and missing data, while assuming no knowledge of a template, user-placed markers, segmentation, or the skeletal structure of the shape. We explicitly sample the motion, which gives a priori the set of possible rigid transformations between parts of the shapes. This transforms the problem into a discrete labeling problem, where the goal is to find an optimal assignment of transformations for aligning the shapes. We then apply graph cuts to optimize a novel cost function, which encodes a preference for a consistent motion assignment from both source to target and target to source. We demonstrate the robustness of our method by aligning several synthetic and real-world datasets.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling

1. Introduction

A key task in computer graphics is the acquisition and construction of 3D geometric models. As cheaper and faster range scanning devices are becoming available, it is getting easier to scan the shape of real world objects. An important post-processing task is to align the acquired scans from multiple viewpoints to a common pose. If the object stays rigid during the scanning process, it suffices to estimate a single rigid transformation for the alignment. However, if the object deforms, we must estimate a movement that now varies spatially across the object. Furthermore, there is usually a significant amount of missing surface data due to occlusion in the range scanning process.

In this paper, we present an algorithm that aligns a pair of shapes with significant motion and missing data. We focus on articulated objects, where the motion is composed of a few rigidly moving parts. We exploit this property to develop an algorithm that is able to register shapes with different connectivity and topology, while assuming no temporal coherence, no prior knowledge of a *template* shape, manually specified feature markers, nor any knowledge of the segmentation and skeletal structure of the model.

Our approach optimizes a novel cost function to evaluate the alignment between two surfaces. The optimization is divided into two steps: sampling the motion between the two surfaces, and applying graph cuts to assign the best spatially varying motion that aligns the shapes while preserving its structure. Our method is able to disambiguate between multiple possible assignments of similarly shaped parts given sufficient connectivity within the shape. We apply our approach to a number of real-world and synthetic datasets, and we demonstrate that we can align the shapes accurately. In addition, our algorithm is robust to missing data in both shapes, and the resulting alignment can fill in the missing parts of one shape with the other. Once this basic registration is completed, one can perform higher level tasks, such as constructing templates, interpolating shapes, automatically rigging skeletons, and building general shape models.

Our main contributions are the following:

- We formulate the registration problem as a label assignment problem, where the labels are rigid transformations that can be assigned to the surface.
- We develop a novel cost function for simultaneously solving a consistent assignment of forward transformations (from the source to the target) and backwards transformations (from the target to the source).
- We optimize this cost function using graph cuts, resulting in an assignment of transformations that aligns the shapes. In addition, contiguous regions with the same transformation give a segmentation of the shape into rigid parts.

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.
Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

2. Related Work

The surface registration problem is a classic, well-studied problem in computer graphics. The Iterative Closest Point (ICP) algorithm is a well-known solution for registering rigid surfaces [CM91, BM92]. Based on this algorithm, a variety of techniques have been developed to handle non-rigid surfaces [She00, CR03, HTB03, ARV07, BR07]. These resemble the ICP algorithm because they use the closest point mapping between the surfaces as a preliminary correspondence to optimize for a non-rigid alignment transformation. The limitation of these techniques is that they compensate for a local non-rigid warp and are not designed to handle significant changes in pose. Notably Chui and Rangarajan [CR03] use soft correspondence assignments and deterministic annealing to robustly handle outliers and a wider range of deformation, but the deformation is constrained to be smooth, unlike the piecewise discontinuous motion observed in articulated models. Compared to these techniques, our method is robust to changes in pose, because it can automatically align shapes where the parts have moved significantly.

A number of algorithms rely on user-placed feature markers to guide the registration. Allen and colleagues [ACP03] describe a template-based non-rigid registration algorithm to register a set of body scans. The example-based 3D scan completion work by Pauly et al. [PMG*05] also uses markers to optimize a per-vertex displacement aligning complete examples to partial range scans. A large body of work on mesh morphing solves the correspondence problem by finding a common base parameterization across multiple meshes of a common topological type [KR91, KCP92, KSK97, GSL*98, LSS*98, LDSS99, KSK00, OKT01, PSS01, TKO01, KS04]. Although these methods require a set of correspondence markers, our algorithm does not require any.

Several recent techniques exploit temporal coherence to align a sequence of range scans. Mitra and colleagues entirely avoid computing correspondences by modeling the shape as a continuously time-varying surface [MFO*07]. The continuous statistical optimization approach of Wand and colleagues [WJH*07] iteratively aligns a sequence of point clouds to assemble a deforming geometric model. Sagawa and colleagues [SOY07] rely on matching both texture and shape features to register a sequence of deforming textured range scans. Pekelny and Gotsman [PG08] additionally require a manual segmentation of the articulated object and its skeletal connectivity. They perform ICP for each segment to gradually accumulate the complete geometry from each additional frame of the scanned sequence. Our algorithm does not require the motion to be small between the shapes, and furthermore it does not depend on any prior knowledge of texture or segmentation.

A closely related work is the symmetrization algorithm by Mitra et al. [MGP07], where shape registration is performed as an application of symmetry detection. Although both our method and symmetrization use partial symmetry detection, the symmetrization algorithm does not address how to disambiguate similar parts in the same object.

The correlated correspondence algorithm by Anguelov and colleagues [ASP*04] is also closely related to our work. They also do not use user-placed markers, assume no temporal coherence or rough initial alignment, and have no prior knowledge of the object shape. With the modest requirement of a template shape, they find a good correspondence assignment in spite of significant changes in object pose. Our work differs from the correlated correspondence algorithm, because we find an optimal assignment of *transformations* rather than an explicit assignment of corresponding points. Therefore, our algorithm is able to assign transformations when no corresponding point is available, effectively removing the requirement of a template shape for the registration.

3. Registration Algorithm

In this section, we describe our approach for automatically registering pairs of articulated shapes with significant missing data. We interpret the registration problem as finding, for each point in one shape, a rotation and translation that moves it to the corresponding point in the other. A solution would be to formulate this as a continuous optimization problem and solve for a smoothly varying transformation field. However, this results in a difficult continuous non-linear optimization problem.

In our approach, we take advantage of the observation that articulated objects consist of a few rigidly moving parts. Our main idea is to first determine a finite set of significant motions between the mostly rigid parts of the pair of shapes. We show that it is possible to extract the significant motions even if large parts of the shapes are missing. We then find the registration by solving a label assignment problem, where each transformation corresponds to a label.

A main challenge of this approach is that many shapes contain several similar parts, as for example the four legs of a horse. Therefore, many significant transformations that lead to good alignments of parts will not lead to consistent global registration. We formulate a cost function for the labeling problem that prefers an assignment of transformations that keeps the shape intact and ensures that each part is mapped in a globally consistent manner. Adding this regularization enables us to apply graph cuts for solving the resulting optimization problem.

We describe how to determine the finite set of transformations describing the movement of the object in Section 3.1. We show how to formulate and optimize the cost function for assigning the transformations in Section 3.2. For convenience, we refer to the pair of shapes as $\mathcal{P}$ and $\mathcal{U}$. We assume that each shape $\mathcal{S}$ is represented as a triangle mesh, and we make use of the set of vertices $V_S = \{v_0, v_1, \ldots, v_n\} \in \mathcal{S}$ and the set of edges $E_S = \{(v_i, v_j) \mid v_i, v_j \in V_S\}$ between adjacent vertices.

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.

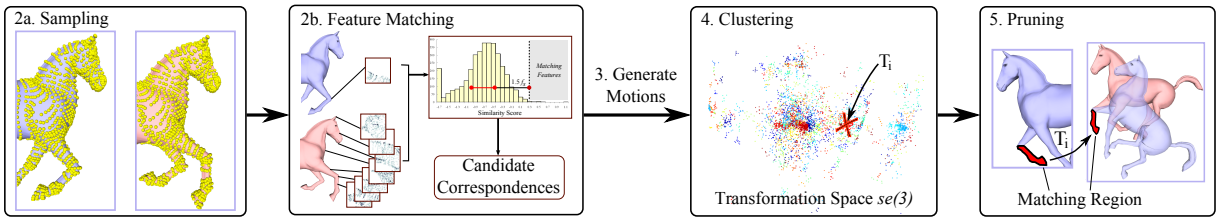


Figure 1: *Motion sampling overview. After precomputing coordinate frames and feature descriptors, we sample the shapes and compare spin image features to find candidate correspondences between the shapes. Using the precomputed coordinate frames, we generate a rigid transformation for each correspondence. Then, we cluster the resulting set of candidate transformations to obtain the final set of transformations. Optionally, we prune unnecessary transformations based on matching regions.*

3.1. Motion Sampling

In this section, we describe an algorithm to find a finite set of rigid transformations $\mathcal{T} = \{T \mid T \in SE(3)\}$ that describes the movement of each part of the shape. We follow the work of Mitra et al. for estimating this set [MGP06]. The algorithm has the following steps, illustrated in Figure 1:

1. Precompute per-vertex coordinate frames and feature descriptors
2. Sample the shapes and match similar features
3. Generate the motion for each match
4. Cluster the transformations
5. Prune transformations based on matching regions

The purpose of the feature matching, clustering, and selection steps is to narrow down on a concise set of transformations that describes the movement of all rigid parts of the shape. Here we discuss the specific design choices made in the implementation of this method, and we refer the reader to Mitra et al. for additional details.

Coordinate Frames. As a preprocessing step, we estimate a coordinate frame on each vertex of the shape. The frame (R,t) contains the 3D location (the position of the vertex, t) and three orthonormal vectors (collected in matrix R) consisting of the normal vector and the two principal curvature directions. The principal curvature directions are estimated by least-squares fitting of the second fundamental form [Rus04]. These frames are used in Step 3 to find a rigid transformation between a pair of corresponding points.

Feature Descriptor. Also in the preprocessing step, we compute at each vertex a "spin image" which describes the local geometry [Joh97]. A spin image is a histogram of the vertices where the bins are concentric rings stacked along the normal direction. This is visualized as a sweeping plane rotating about the normal direction, collecting vertices in a grid defined on the plane. Spin images have been successfully used for computing correspondences in range data and are robust to clutter and occlusion in static scenes. Since we assume that our object has articulated motion composed of several rigidly moving parts, the spin images work well as long as they are localized to small neighborhoods. We have also tried using multi-scale principal curvatures, but we found that spin images were more discriminative.

Sampling and Feature Matching. In these steps, we use the precomputed feature descriptors to find possible corresponding points between the source and target shapes. First, we randomly subsample the set of vertices in each shape, in order to reduce the number of comparisons during the matching step. Next, for a single point p in the source, we find corresponding points in the target by computing the similarity score (higher is better) from p's spin image to all of the spin images in the target (Figure 1). Collecting these scores into a histogram, we find the features that are significantly more similar than the rest—to such a degree that they are outliers in the histogram. We use a moderate threshold of $1.5 f_s + m_u$ to determine outliers, where m_u is the median of the upper half of the measurements, m_l is the median of the lower half of the measurements, and $f_s = m_u - m_l$. We repeat this matching process for all vertices in $\mathcal{P}$ to obtain a large set of correspondence "candidates."

Generating Motions. Now, for each correspondence candidate (p,u) where $p \in \mathcal{P}$ and $u \in \mathcal{U}$, we use the precomputed coordinate frames $T_p = (R_p, t_p)$, $T_u = (R_u, t_u)$ to generate a rigid transformation $T = (R,t)$ from p to u, given by

$$R = R_u R_p^\top, \quad t = t_u - R t_p.$$

Here, R is the rotation and t is the translation from T_p to the T_u, and $SE(3)$ is the space of all rigid transformations. As a result, we have a set of rigid transformations, where each transformation is associated with a single correspondence candidate.

Clustering Motions. In this step, we use a variant of the non-linear mean-shift framework to cluster the set of transformations [TSM05]. Mean-shift is a non-parametric clustering algorithm that finds peaks of the local density of a point set using gradient ascent. When mean-shift clustering is applied for rigid motions, the challenge is to define an appropriate distance measure for comparing transformations. The non-linear mean-shift approach defines the distance between any two transformations $X, Y \in SE(3)$ as

$$d(X,Y) = \| \log(X^{-1} Y) \|$$

where $\log(X)$ maps a transformation $X \in SE(3)$ to the corresponding element x in the Lie algebra $se(3)$, a 6D vector containing axis-angle rotation and linear velocity [TSM05].

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.

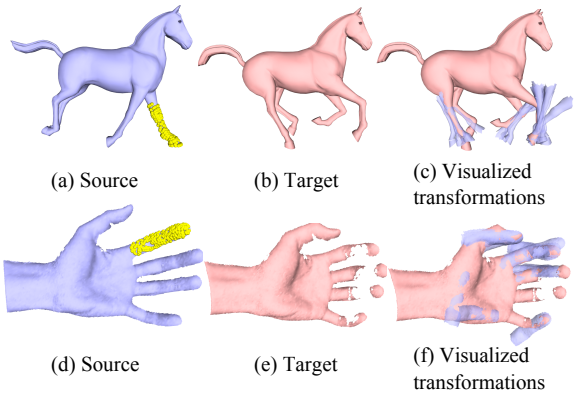


Figure 2: *Visualizing the set of estimated transformations $\mathcal{T}$ from the source (a,d) to the target (b,e). We manually select a region in the source and plot a transformed copy of the region for each associated transformation (c,f). We see that there are many transformations between different but similarly shaped parts. The graph cuts optimization assigns the transformations to map each part correctly.*

Considering the Lie group $SE(3)$ as a differentiable manifold, this distance $d(X,Y)$ corresponds to the length of the geodesic curve between $X^{-1}Y$ and the identity $e \in SE(3)$.

This distance measure is a natural definition that is based on the structure of a Lie group, but it is a non-linear function that is expensive to evaluate and cannot support a data structure for fast range queries. Therefore, we opt for an approximation

$$d'(X,Y) = \| -x+y \|$$

where $x,y \in se(3)$ are the axis-angle and linear velocity representations of $X,Y \in SE(3)$, respectively. We discuss in Appendix A that this is a reasonable approximation. Furthermore, since $d'(X,Y)$ is just Euclidean distance in the Lie algebra $se(3)$, it is faster to evaluate and supports fast approximate range queries using a k-d tree. This is particularly useful when we have a large number of rigid transformations to cluster.

To utilize the distance $d'(X,Y)$, we first map each transformation $T = (R,t) \in SE(3)$ to its corresponding element in $se(3)$. The scale of the rotations R usually differs from the scale of the translations t, so we perform zero mean, unit standard deviation normalization on the translations. As a result, the range of norms for the rotational and transational components are in $\approx [0,\pi]$. Finally, we construct a k-d tree and apply mean-shift clustering using the Epanechnikov kernel. As a result, we obtain a set of clusters, where each cluster is a group of similar transformations. Finally, we compute the cluster modes and collect all of them to obtain the final set of estimated transformations $\mathcal{T}$. A visualization of these transformations is shown in Figure 2.

Pruning. This is an optional step to further refine the set of transformations $\mathcal{T}$. Even after clustering, there are many spurious transformations resulting from incorrect correspondences. Therefore, we want to retain the best transformations and get rid of unnecessary ones.

Most reliable transformations will have the largest matching regions (Figure 1). Therefore, we find the matching region for each transformation, and retain only the top k transformations with the largest matching regions. However, since good transformations corresponding to very small regions may be potentially discarded, we found it acceptable to optionally skip this step if the feature matching is sufficiently accurate and discriminative.

In our implementation, to find the matching region, we start at a random cluster member (a pair of vertices on the source and target) and incrementally grow the region, only to the point where applying the transformation keeps the region within a small distance of the target. In addition, we refine the accuracy of the transformation during this time by performing ICP at regularly scheduled intervals. When there is missing data in the mesh, there will typically be boundary vertices, which are the vertices of edges that are adjacent to a single triangle. To help ICP to be more robust to missing data, we slightly modify the point selection step: in the case where the closest point u is on a boundary, we instead use the closest point on the tangent plane at u. This strategy helps to provide accurate transformations when there are large missing regions in the data.

Conclusion. The final output of the motion sampling is a finite set of rigid transformations $\mathcal{T} = \{T \mid T \in SE(3)\}$ of the best transformations determined by the clustering or selection step.

3.2. Graph Cuts Optimization

The next step in our method is to assign the transformations to the shapes. This is essentially a labeling problem, where the vertices of the shape are the nodes and the transformations are the labels assigned to each node. We first develop a cost function that measures the quality of assigning the transformations, and describe how we apply graph cuts to optimize this cost.

Cost Function. We expect that a good transformation assignment aligns the shapes well and results in a consistent deformation that preserves the connectivity of the shape. We express this preference as minimizing a cost function

$$\underset{\{T_p \in SE(3)\ \forall p \in \mathcal{P}\}}{\operatorname{argmin}} \sum_{p \in \mathcal{P}} D(p,T_p) + \sum_{(p,q) \in E_P} V(p,q,T_p,T_q) \quad (1)$$

where $D(p,T_p)$ represents a *data cost* measuring how well the application of rigid transformation T_p matches p to $\mathcal{U}$, E_P specifies the connectivity of $\mathcal{P}$, and $V(p,q,T_p,T_q)$ is a *smoothness cost* measuring the consistency of the deformation between a pair of points $(p,q) \in E_P$. This cost function is reminiscent of one developed by Allen et al. [ACP03], except that they consider T_p to be affine transformations. Stated purely in this form, minimizing this function results in a continuous, non-linear optimization problem that is generally difficult to minimize.

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.

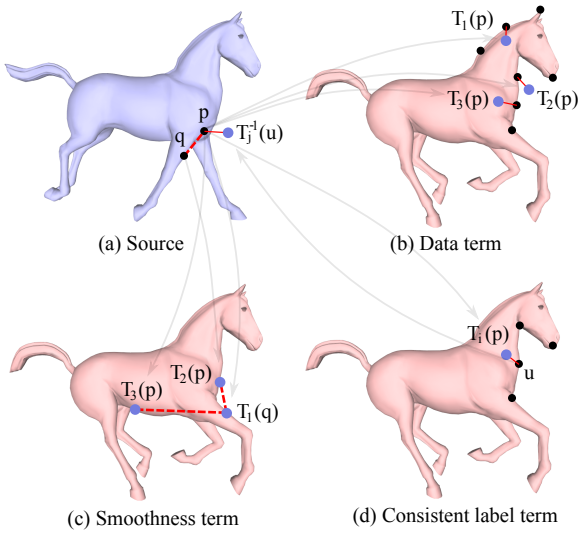


Figure 3: *Illustration of the cost function from the source (a) to the target (b,c,d). The data term (b) measures the distance of the transformed point $T(p)$ to the closest point on the target. The smoothness term (c) measures the change of length between neighboring points p,q after applying the transformations. The consistent label term (d) ensures that the transformation T_i assigned to p is consistent with T_j assigned to the closest point u, i.e. $T_i = T_j$.*

Our observation is that motion sampling gives *a priori* the set of possible transformations to assign to each vertex $p \in \mathcal{P}$. Instead of finding each T_p in the continuous space of rigid transformations, for each p we make a selection f_p of some transformation in a prescribed set $\mathcal{T}$. Thus, we transform the above continuous optimization problem to a discrete one:

$$\underset{\{f_p \in \mathcal{T}\ \forall p \in \mathcal{P}\}}{\operatorname{argmin}} \sum_{p \in \mathcal{P}} D(p, f_p) + \sum_{(p,q) \in E_P} V(p, q, f_p, f_q). \quad (2)$$

If we consider the transformations as labels to assign to each vertex, we see indeed that minimizing the above cost function corresponds to solving for an optimal labeling of the vertices. Therefore, we can use graph cuts for the optimization.

What remains is a definition of the data and smoothness terms in the objective function. For the data term, we provide a catalog of useful functions in Table 1. The point-to-point metric is the most basic matching function, illustrated in Figure 3b. Compared to the point-to-point metric, the point-to-plane error metric is more robust to missing data and the sampling of the surface. We prefer to use this metric when the sampling of the shape is very sparse. The hybrid metric is designed for missing data and measures a combination of the point-to-point and point-to-plane metrics, extrapolating the distance at the boundary of the shape. We choose this term when there is a substantial amount of missing data. Finally, the point-and-normal metric measures both the point-to-point distance and the difference of the normal vectors.

Point-to-point	$D_p(p, f_p) = \|f_p(p) - u\|$
Point-to-plane	$D_l(p, f_p) = \|(f_p(p) - u) \cdot n_u\|$
Hybrid	$D_h(p, f_p) = \begin{cases} D_l & \text{if } u \in \text{boundary} \\ D_p & \text{otherwise} \end{cases}$
Point-and-normal	$D_n(p, f_p) = \sqrt{D_h + \nu \|f_p(n_p) - n_u\|}$

Table 1: *A catalog of data cost functions. For a vertex p with normal n_p and a selected transformation f_p, $f_p(p)$ applies f_p to p (only rotation for normals), and $u \in \mathcal{U}$ is the closest point to $f_p(p)$ with associated surface normal n_u.*

Here, ν is an additional parameter controlling the influence of the normals. We use the point-and-normal term when precise normals are available.

For the smoothness term, the goal is to preserve the consistency of the shape. A possibility is to directly compare the labels or to compare the transformations between neighboring vertices. However, we do not want to penalize differing transformations due to potential joints in the articulated shape. Therefore, we preserve the *edge lengths* of neighboring points

$$V(p, q, f_p, f_q) = \Big| \|p - q\| - \|f_p(p) - f_q(q)\| \Big|. \quad (3)$$

If $f_p = f_q$ then we see that $V(p,q,f_p,f_q) = 0$, as is desirable for a smoothness function. In addition, this expression does not penalize the assignment f_p and f_q as long as it preserves the edge length $\|p-q\|$. This means that we can assign completely different transformations at neighbors p and q as long as it doesn't break the shape apart. As an example, in Figure 3c, we see that assigning T_2 to p preserves the edge length, while T_3 is penalized because it stretches the edge.

Graph Cuts. To apply graph cuts to minimize the objective in Equation 2, we construct an instance where the points $p \in \mathcal{P}$ are the sites, the edges $(p,q) \in E_P$ are neighbors, and the transformations $f \in \mathcal{T}$ are labels. We then solve the resulting multiple-label assignment problem using the α-expansion algorithm [BVZ01].

Symmetric Cost Function. In many cases swapping the inputs $\mathcal{P}$ and $\mathcal{U}$ gives different results, because in Equation 2 we are only optimizing for an assignment of the *forward* transformations from $\mathcal{P}$ to $\mathcal{U}$. To make maximum use of both shapes, we formulate a *symmetric* cost function, where we simultaneously solve for a consistent assignment of the forward transformations $\mathcal{T}$ to $\mathcal{P}$ and backward transformations $\mathcal{T}' = \{T^{-1} \mid T \in \mathcal{T}\}$ to $\mathcal{U}$. For convenience, we can assign forward transformation labels $f_u \in \mathcal{T}$ to points $u \in \mathcal{U}$, except that we apply the inverse transformation when evaluating the cost functions.

To construct a symmetric graph cuts instance, first we include the vertices and connectivity of both shapes $\mathcal{P}$ and $\mathcal{U}$ to simultaneously solve for the assignment in both directions. Next, we enforce corresponding points under a transformation assignment to have the same label. For example, in Figure 3d, we penalize the case where $T_i \neq T_j$. Formally, for each transformation $T \in \mathcal{T}$, we introduce a new edge

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.

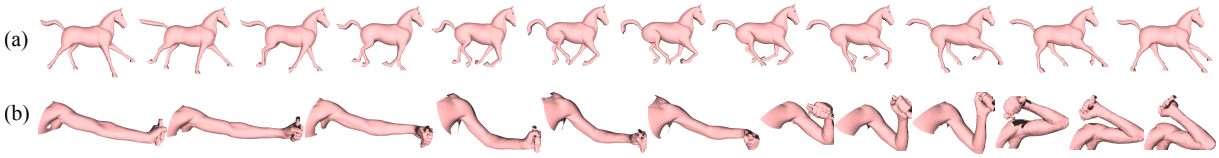
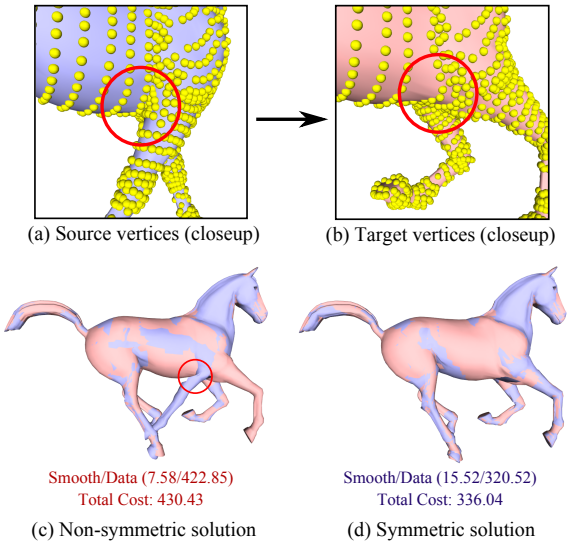


Figure 4: *Examples used for testing our algorithm. Shown are the twelve poses in the horse dataset (a) and the arm dataset (b).*



(a) Source vertices (closeup) (b) Target vertices (closeup)

(c) Non-symmetric solution (d) Symmetric solution

Figure 5: *Comparing non-symmetric and symmetric graph cuts. In this example, the skin joining the thigh and the torso is significantly stretched (a,b). The non-symmetric solution prefers to preserve the edge lengths in this region (c), resulting in a sub-optimal local minimum where the front right leg is aligned incorrectly. However, assigning consistent forwards and backwards transformations gives enough incentive to afford the stretch in the front right leg (d).*

$e_T = (p,u)$ for all $p \in \mathcal{P}$ and $u \in \mathcal{U}$ such that $u \in \mathcal{U}$ is the closest point to $T(p)$, or $p \in \mathcal{P}$ is the closest point to $T^{-1}(u)$. Let E_T be the set of all such edges. Then the *consistent label term* W is given by

$$\underset{\{f_p, f_u \in \mathcal{T}\ \forall p \in \mathcal{P}, u \in \mathcal{U}\}}{\operatorname{argmin}} \sum_{(p,u) \in E_T} W(p,u,f_p,f_u). \qquad (4)$$

This term serves to penalize assignments where the transformations disagree between corresponding points. Thus

$$W(p,u,f_p,f_u) = \begin{cases} c_W & f_p \neq f_u \text{ and } u \text{ is closest to } f_p(p) \text{ or,} \\ & f_p \neq f_u \text{ and } p \text{ is closest to } f_u(u), \\ 0 & \text{otherwise,} \end{cases}$$

where c_W is a constant penalty of an inconsistent label assignment. An example of the benefit of this term is shown in Figure 5.

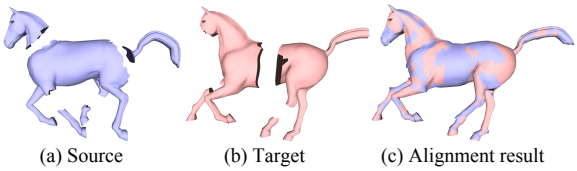
One caveat in using graph cuts is that the smoothness term in Equation 3 is not a semi-metric as required to obtain a strong local minimum. However, in practice we obtain good results with the α-expansion algorithm. In addition, lower cost solutions have consistently yielded qualitatively better results, which suggests that registration quality can be improved by using a stronger optimization technique.

4. Results

In this section we present results of aligning one synthetic dataset of a galloping horse mesh animation, one real-world dataset of human arm scans, and another real-world dataset of human hand scans [SP04, ACP03, WLG07]. Some examples used in our experiments are shown in Figure 4. To our knowledge, ours is the only algorithm which does not use any prior information about the shape or alignment, and is robust to both significant motion and missing data.

We were successfully able to automatically align most pairs of the 12 galloping horse examples (Figure 4a). To measure the quality of the alignment, we use the maximum symmetric Hausdorff distance, which is the maximum distance to the closest point for each vertex in both shapes. Out of a total 66 pairs, we obtained a good registration in 64 trials, where the distance was at most 5.6% of the bounding box. In only 2 examples, some legs were swapped, resulting in a maximum distance of 20% of the bounding box diagonal. This demonstrates that our algorithm can handle a wide range of motion and is particularly robust in this dataset.

Figure 7 shows an example of aligning two frames of this animation. Although our method has no input other than just the shapes, we obtain an accurate alignment with a registration error within 2% of the entire scene size (length of the shape's bounding box diagonal). All four legs in the source mesh have aligned to the correct leg in the target mesh, demonstrating that our optimization can map similar parts correctly. We see that the assigned transformations naturally give a high quality segmention of the mesh into rigid components (Figure 7e), which can be used to automatically create



(a) Source (b) Target (c) Alignment result

Figure 6: *Even after manually removing parts in both source and target, our method is able to align the meshes well.*

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.

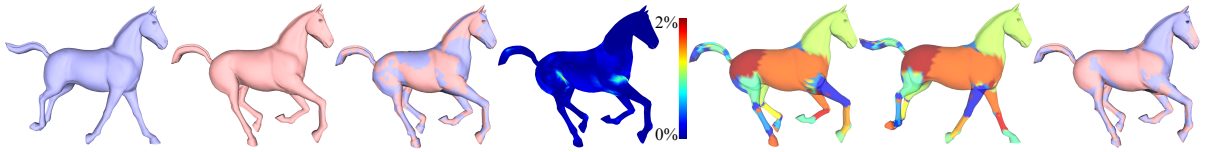


Figure 7: *Registration results for the synthetic horse dataset. Given a pair of meshes (a) and (b), our algorithm results in a close alignment (c). Notice that all four legs are matched to the correct part. We also obtain a low registration error in (d), which shows the per-vertex distance to the target shape as a percentage of the bounding box diagonal. In addition, our method gives a meaningful segmentation of the model into rigid parts that is consistent in both shapes (e). The last image (g) shows the result of aligning a 50% simplified version of the same pair. This demonstrates that our method performs well independent of the sampling and parameterization of the shapes.*

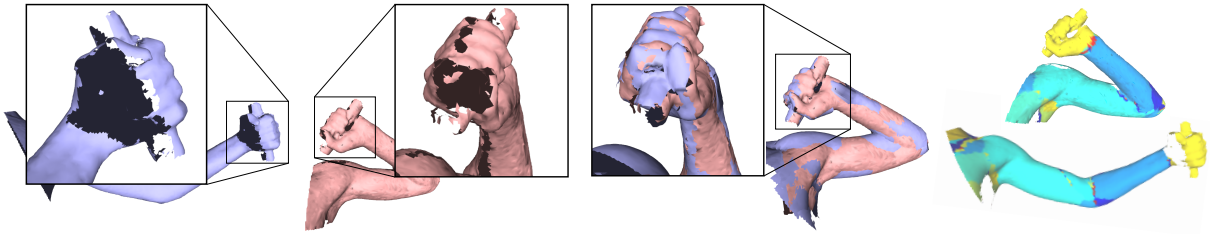


Figure 8: *Registration for an arm dataset pair. The source mesh (a) is aligned to the target mesh (b). The hand region is missing a significant amount of data in both meshes, but after alignment the surface of the hand is completed nicely (c). The assigned labels are shown in (d) for the source (bottom) and the target (top), and corresponding parts have the same label assignment. Also, the segmentation naturally corresponds to the different parts of the arm.*

a skeleton. In addition, the segmentation is consistent between both source and target shapes, as we can see in the figure where corresponding parts are colored with the same label. Finally, we obtain a good registration even when simplifing the meshes to 50%, demonstrating that our method is independent of the specific parameterization of the shape.

To test our algorithm on examples with significant missing data, we have manually removed parts of both the source and target meshes (Figure 6-abc). The holes were placed in different locations so that the alignment result would complete the entire shape. Even though the holes completely disconnect the meshes into several fragments, we obtain a good alignment result. This shows that our algorithm is robust to missing data and that it can be used for automatic shape completion using a set of incomplete examples.

For the arm scan dataset, our algorithm successfully aligned all pairs of the 12 examples (Figure 4b). The registration performed well even when there was significant missing data in both the forearm and the hand. In the example shown in Figure 8, notice that the aligned result has nicely completed the thumb and index finger region of the hand area (c). Just like the horse example, the assigned transformations segment the shape into meaningful rigid parts (d), which can be used to automatically create a skeleton.

Finally our algorithm was able to align many examples for the hand grasping dataset. This dataset is particularly challenging because of the occlusion in the fingers. Figure 9 shows two examples of aligning a pair of these scans. Despite significant missing data and movement in the fingers, our method successfully aligns all four fingers to the correct position. As a result, the holes in the fingers are filled in using data from the source shape.

Parameters. For the feature matching, the spin images were quite discriminative, so we limited the number of matching features for each vertex to 5-7 vertices. For the mean-shift clustering, since we take the cluster modes to be the estimated transformations, we set the mean-shift bandwidth to a small $h = 0.1$ in order to prevent over-smoothing of the transformations.

We use a fraction of the bounding box diagonal as parameter values because it naturally relates to the data and smoothness costs which measure distances between points. We set the error of an inconsistent label assignment c_W to 100 times the bounding box diagonal. Also, we set the closest point distance threshold to 0.5% of the bounding box diagonal for finding matching regions in the pruning step and for determining corresponding points in Equation 4.

The data cost that we use for each dataset is summarized in the rightmost column of Table 2. We set ν to be 2% of the bounding box diagonal for the point-and-normal metric. To control the relative influence of the data and smoothness terms, we multiply each with a constant weight c_D and c_V, respectively. In our experiments, we used $c_D = 1$ for all datasets, $c_V = 5$ for the arm and hand datasets, and $c_V = 10$

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.

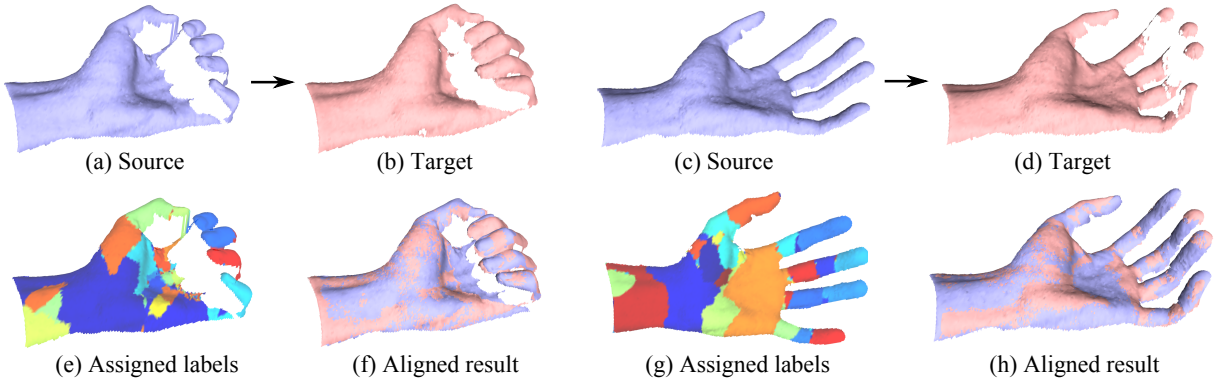


Figure 9: *Registration results for hand dataset examples. Although the hand is missing a significant amount of data in the fingers (b,d), our method is able to successfully align all four fingers (f,h). Notice that the segmentation produced in (e,g) roughly corresponds to the actual segments in the fingers of a hand.*

Dataset	# Vertices	# Samples	# Labels	Matching	Clustering	Verification	Graph Cuts	Data Cost
Horse	8431	4339	1500	2.1 min	3.0 sec	(skip) 1.6 sec	1.1 hr	D_n
Arm	11865	6094	1000	55.0 sec	0.9 sec	12.4 min	1.2 hr	D_p
Hand (Front)	8339	2945	1500	14.5 sec	0.7 sec	7.4 min	1.2 hr	D_l
Hand (Back)	6773	2669	1500	17.3 sec	0.9 sec	9.4 min	1.6 hr	D_l

Table 2: *Averaged performance and timing statistics for a typical subset of our experiments. The running time statistics were gathered from testing our implementation on a single core of a dual core 2.4 GHz Intel processor.*

for the horse dataset.

Performance. The statistics and running time of the experiments are summarized in Table 2. The most time-consuming portion of our algorithm is the graph cuts optimization, which depends on the number of sites as well as the number of labels. Since our graph cuts instance optimizes the assignment of all vertices in both source and target meshes, we simplify the meshes using the quadric error metric technique [GH97] to reduce the running time.

5. Discussion and Future Work

In this section we discuss the limitations of our method and point out several avenues for future work.

Motion Sampling. The quality of our registration result depends on how accurately we sample the transformations. A good sampling should have all the necessary transformations to align each part, while accurately narrowing down on a concise subset of transformations. In some horse examples, small parts such as hooves or legs were misaligned, because the correct transformation was not in the set of estimated transformations $\mathcal{T}$. In these cases, there were too few samples to extract these transformations accurately. In our experiments, we found that when the feature matching is noisy, the clusters found by our algorithm are spread out more uniformly in the transformation space. In this case, the clustering acts somewhat like a sampling strategy rather than finding densely clustered rigid components. It would be interesting to investigate other sampling strategies for the motion sampling step, such as adaptive sampling. Furthermore, it may be beneficial to adaptively sample the motion *during* the optimization to automatically refine the registration result.

A good extension, as also pointed out by Mitra et al. [MGP06], is to incorporate a method for extracting a continuous set of motions for non-rigid examples.

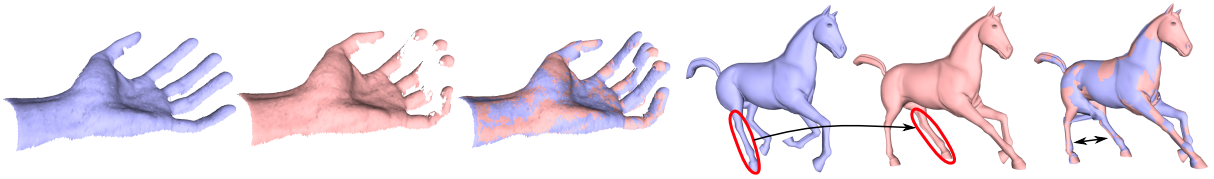
Another issue is that the assignment of transformations can disconnect or imperfectly match the boundaries between rigid components. In these cases, our method can be used as an initialization for an additional refinement step (such as non-rigid ICP) to obtain a smoother and more precise registration.

Finally, an open problem is to characterize exactly how much data is required for an accurate alignment. For example, the alignment in Figure 10-abc is less accurate. The main difficulty with missing data in our method is that there is not enough data to estimate the correct motion or to guide the optimization. It would be interesting to investigate what are the fundamental limitations for resolving missing data.

Optimization. In general, the optimal labeling problem is known to be NP-hard [BVZ01]. However, utilizing recent methods such as sequential tree-reweighted message passing (TRW-S) [Kol06] or Log-cut [LRB07] may improve both the quality and efficiency of the registration. Also, applying a coarse-to-fine optimization technique may help as well.

In our experiments, we found that often there is a trade-off between minimizing the smoothness and the data costs. If the smoothness weight is too high, then the optimization

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.



(a) Source (b) Target (c) Aligned result (d) Source (e) Target (f) Aligned result

Figure 10: *Typical errors in the registration. In the hand example (a,b,c), the missing data causes the fingers to become stretched, slightly disconnected, or misaligned. In the horse example (d,e,f), the rear legs are swapped.*

can prefer a badly aligned solution in favor of preserving the edge lengths of the mesh (e.g. Figure 5c). On the other hand, if the data weight is too high, the optimization may choose to break the mesh and map parts incorrectly. For example, in Figure 10-def, the rear right leg of the source is most similar to the rear left leg of the target. In this case, the low data cost managed to compensate for the penalty of the smoothness term when the rear legs were swapped. As an area of future work, it would be useful to determine the weights automatically. Another interesting idea is to include a "dummy label" to explicitly label vertices with no correspondence due to missing data.

Finally, since our method is currently limited to a pair of shapes, extending it to simultaneously align more than two shapes may help to resolve more missing data.

6. Conclusion

We have presented an automatic method for aligning a pair of articulated shapes in the presence of significant motion and missing data. We formulate the registration problem as assigning rigid transformations to each vertex of the shape. Our algorithm first determines a finite set of possible motions of the shape by sampling the motion. Then, it uses graph cuts to optimize a cost function, which measures the quality of an assignment for aligning the shapes and preserving the consistency of the transformed mesh. Furthermore, we develop a symmetric cost function for simultaneously obtaining a consistent assignment for both source to target and target to source. Our experimental results show that despite no prior knowledge of a template, user-placed markers, segmentation, or the skeletal structure, the algorithm is able to find good alignments between different poses of the shape. We also obtain a natural segmentation of the shape into rigid parts, given by contiguous regions with the same assigned label.

Acknowledgements. We wish to thank Bob Sumner for the galloping horse animation, Brett Allen for the body scans, and Thibaut Weise for the hand scan dataset. Additional thanks to Gilles Debunne for providing the libQGLViewer library, David M. Mount and Sunil Arya for providing the ANN library, and Yuri Boykov, Olga Veksler, Ramin Zabih for providing an implementation of their graph cuts algorithm. We also thank the anonymous reviewers and Toshiya Hachisuka, Wojciech Jarosz, Neel Joshi, and Wan-Yen Lo from the UCSD Graphics Lab for their valuable feedback.

References

[ACP03] ALLEN B., CURLESS B., POPOVIĆ Z.: The space of human body shapes: reconstruction and parameterization from range scans. In *ACM SIGGRAPH* (2003), pp. 587–594.

[ARV07] AMBERG B., ROMDHANI S., VETTER T.: Optimal step nonrigid icp algorithms for surface registration. In *CVPR* (2007).

[ASP*04] ANGUELOV D., SRINIVASAN P., PANG H.-C., KOLLER D., THRUN S., DAVIS J.: The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces. In *NIPS* (2004).

[BC04] BLANES S., CASAS F.: On the convergence and optimization of the baker–campbell–hausdorff formula. *Linear Algebra and its Applications 378* (2004), 135–158.

[BM92] BESL P. J., MCKAY H.: A method for registration of 3-d shapes. *PAMI 14*, 2 (1992), 239–256.

[BR07] BROWN B. J., RUSINKIEWICZ S.: Global nonrigid alignment of 3-d scans. In *ACM SIGGRAPH* (2007), p. 21.

[BVZ01] BOYKOV Y., VEKSLER O., ZABIH R.: Fast approximate energy minimization via graph cuts. *PAMI 23*, 11 (2001), 1222–1239.

[CM91] CHEN Y., MEDIONI G.: Object modeling by registration of multiple range images. *Proceedings of the IEEE International Conference on Robotics and Automation 3* (1991), 2724–2729.

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

[GH97] GARLAND M., HECKBERT P. S.: Surface simplification using quadric error metrics. In *ACM SIGGRAPH* (1997), pp. 209–216.

[GSL*98] GREGORY A. D., STATE A., LIN M. C., MANOCHA D., LIVINGSTON M. A.: Feature-based surface decomposition for correspondence and morphing between polyhedra. In *CA '98: Proceedings of the Computer Animation* (1998), p. 64.

[HTB03] HÄHNEL D., THRUN S., BURGARD W.: An extension of the icp algorithm for modeling nonrigid objects with mobile robots. In *IJCAI* (2003), pp. 915–920.

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.

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

[KCP92] KENT J. R., CARLSON W. E., PARENT R. E.: Shape transformation for polyhedral objects. *ACM SIGGRAPH 26*, 2 (1992).

[Kol06] KOLMOGOROV V.: Convergent tree-reweighted message passing for energy minimization. *PAMI 28*, 10 (2006), 1568–1583.

[KR91] KAUL A., ROSSIGNAC J.: Solid-interpolating deformations: Construction and animation of pips. In *EUROGRAPHICS* (1991), pp. 493–505.

[KS04] KRAEVOY V., SHEFFER A.: Cross-parameterization and compatible remeshing of 3d models. *ACM SIGGRAPH 23*, 3 (2004), 861–869.

[KSK97] KANAI T., SUZUKI H., KIMURA F.: 3d geometric metamorphosis based on harmonic map. In *PG '97: Pacific Conference on Computer Graphics and Applications* (1997), p. 97.

[KSK00] KANAI T., SUZUKI H., KIMURA F.: Metamorphosis of arbitrary triangular meshes. *IEEE Comput. Graph. Appl. 20*, 2 (2000), 62–75.

[LDSS99] LEE A. W. F., DOBKIN D., SWELDENS W., SCHRÖDER P.: Multiresolution mesh morphing. In *ACM SIGGRAPH* (1999), pp. 343–350.

[LRB07] LEMPITSKY V., ROTHER C., BLAKE A.: Logcut – efficient graph cut optimization for markov random fields. In *ICCV* (2007).

[LSS*98] LEE A. W. F., SWELDENS W., SCHRÖDER P., COWSAR L., DOBKIN D.: Maps: multiresolution adaptive parameterization of surfaces. In *ACM SIGGRAPH* (1998), pp. 95–104.

[MFO*07] MITRA N. J., FLORY S., OVSJANIKOV M., GELFAND N., GUIBAS L. J., POTTMANN H.: Dynamic geometry registration. In *SGP* (2007), pp. 173–182.

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

[MGP07] MITRA N. J., GUIBAS L. J., PAULY M.: Symmetrization. In *ACM SIGGRAPH* (2007), p. 63.

[OKT01] OHBUCHI R., KOKOJIMA Y., TAKAHASHI S.: Blending shapes by using subdivision surfaces. *Computers & Graphics 25*, 1 (2001), 41–58.

[PG08] PEKELNY Y., GOTSMAN C.: Articulated object reconstruction and markerless motion capture from depth video. *EUROGRAPHICS 27*, 2 (2008).

[PMG*05] PAULY M., MITRA N. J., GIESEN J., GROSS M., GUIBAS L. J.: Example-based 3d scan completion. In *SGP* (2005), p. 23.

[PSS01] PRAUN E., SWELDENS W., SCHRÖDER P.: Consistent mesh parameterizations. In *ACM SIGGRAPH* (2001), pp. 179–184.

[Rus04] RUSINKIEWICZ S.: Estimating curvatures and their derivatives on triangle meshes. In *3DPVT* (2004), pp. 486–493.

[She00] SHELTON C. R.: Morphable surface models. *Int. J. Comput. Vision 38*, 1 (2000), 75–91.

[SOY07] SAGAWA R., OSAWA N., YAGI Y.: Deformable registration of textured range images by using texture and shape features. In *3DIM* (2007), pp. 65–72.

[SP04] SUMNER R. W., POPOVIĆ J.: Deformation transfer for triangle meshes. In *ACM SIGGRAPH* (2004), pp. 399–405.

[TKO01] TAKAHASHI S., KOKOJIMA Y., OHBUCHI R.: Explicit control of topological transitions in morphing shapes of 3d meshes. In *PG '01: Pacific Conference on Computer Graphics and Applications* (2001), p. 70.

[TSM05] TUZEL O., SUBBARAO R., MEER P.: Simultaneous multiple 3d motion estimation via mode finding on lie groups. In *ICCV* (2005), pp. 18–25.

[WJH*07] WAND M., JENKE P., HUANG Q., BOKELOH M., GUIBAS L. J., SCHILLING A.: Reconstruction of deforming geometry from time-varying point clouds. In *SGP* (2007), pp. 49–58.

[WLG07] WEISE T., LEIBE B., GOOL L. J. V.: Fast 3d scanning with automatic motion compensation. In *CVPR* (2007).

A. Approximated Distance Metric in $SE(3)$

The difference between $d(X,Y)$ and $d'(X,Y)$ can be expressed in terms of the Baker–Campbell–Hausdorff (BCH) formula:

$$\begin{aligned}\log(X^{-1}Y) &= \log(\exp(-x)\exp(y)) \\ &= \log(\exp(\mathrm{BCH}(-x,y))) \\ &= \mathrm{BCH}(-x,y) \\ &= -x+y+O([-x,y])\end{aligned}$$

where $[x,y] = xy - yx$ is the Lie bracket operator or commutator of the Lie algebra, and $BCH(x,y)$ is a series expansion based on nested iterated commutators. Blanes and Casas show that $x+y$ is a good approximation to $\log(e^x e^y)$ when the norm of the commutator $[x,y]$ is sufficiently small [BC04].

© 2008 The Author(s)
Journal compilation © 2008 The Eurographics Association and Blackwell Publishing Ltd.