Feature-driven Deformation for Dense Correspondence

Deboshmita Ghosh, Andrei Sharf, and Nina Amenta

Department of Computer Science, University of California, Davis, California

ABSTRACT

Establishing reliable correspondences between object surfaces is a fundamental operation, required in many contexts such as cleaning up and completing imperfect captured data, texture and deformation transfer, shape-space analysis and exploration, and the automatic generation of realistic distributions of objects. We present a method for matching a template to a collection of possibly target meshes. Our method uses a very small number of user-placed landmarks, which we augment with automatically detected feature correspondences, found using spin images. We deform the template onto the data using an ICP-like framework, smoothing the noisy correspondences at each step so as to produce an averaged motion. The deformation uses a differential representation of the mesh, with which the deformation can be computed at each iteration by solving a sparse linear system.

We have applied our algorithm to a variety of data sets. Using only 11 landmarks between a template and one of the scans from the CEASAR data set, we are able to deform the template, and correctly identify and transfer distinctive features, which are not identified by user-supplied landmarks. We have also successfully established correspondences between several scans of monkey skulls, which have dangling triangles, non- manifold vertices, and self intersections. Our algorithm does not require a clean target mesh, and can even generate correspondence without trimming our extraneous pieces from the target mesh, such as scans of teeth.

Keywords: Dense correspondence, Template fitting

1. INTRODUCTION

As more and more real-world data is digitized, new opportunities arise to use assemblages of three-dimensional computer models. For instance, laser scans of faces[1] have been used to fit a three-dimensional model to a single photograph, and a collection of laser scans of and human bodies[2] has been used to generate crowds of digital extras which follow real distributions of physical features. The "shape spaces" implied by collections of models can be used for visualization and analysis[3] and provide a principled approach for applications such as deformation transfer[4] or scan completion.[5] Probabilistic models of the space of shape deformations implied by a collection of models are very important in medical imaging. Collections of digitized organs, especially brains, form the input to active shape models,[6] which can be used, among other things, for atlas-based segmentation of volumetric data.

Figure 1 shows a specific example: an assemblage of scanned crania from different species of monkeys. The scanned meshes evince many typical problems, such as holes (both large and small), dangling triangles (especially inside the eye sockets and noses), non-manifold vertices, and sometimes self-intersections; in other applications, we also have extraneous surfaces in the target meshes (see Figure 8). Establishing a dense set of point correspondences enables statistical analysis of the shape variation, which in this case can elucidate the relationship of morphology and evolutionary history. Minimizing user involvement in establishing the correspondences makes it possible to do large, statistically meaningful, analysis.

We consider the correspondence problem in this situation, in which the data provides an estimate of the surface of an object, but may not form a reliable manifold mesh. Like many others, we take the approach of fitting a *template mesh* to each of the input target meshes, so that the correspondences of each input data set with the template provides a multi-way correspondence between the data sets. The template includes relevant
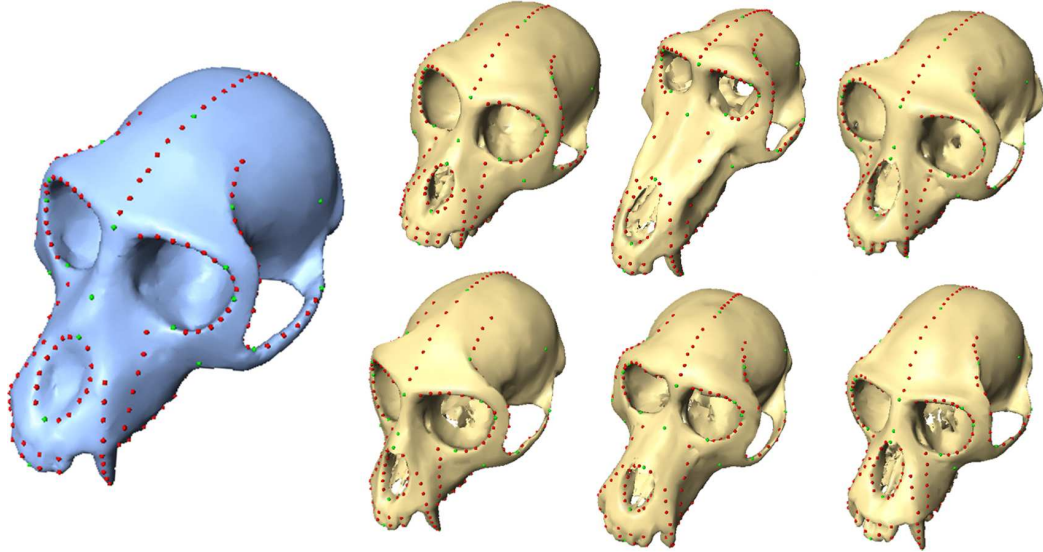
Figure 1. Establishing a dense correspondence between a template skull (left) and scanned monkey skulls, representing different species and sexes. Using only few user specified landmarks (green), we place many others (red) at corresponding locations on the target meshes. The target meshes have many holes and dangling triangles, and some self-intersections

shape features, which are preserved by the deformation, ensuring that features on one scan correspond correctly with features on another.

Our main contribution is the algorithm which fits a clean template mesh to the data. Our method explicitly analyzes the shapes of the template and the target meshes, and automatically moves similarly-shaped regions closer together while preserving the template shape in noisy regions, regions with holes, or regions with poor similarity.

Our algorithm has three steps. First, we initialize the placement of the template mesh using a very small number of user-placed landmarks. We then *automatically* find corresponding points on the template and target, using *spin images*,[7] a kind of local shape signature. We use these matches to deform the template mesh. We use the differential *moving frames* construction,[8] which allows us to compute the deformed mesh by solving two linear systems. This deformation brings the overall shape of the template closer to that of the target while maintaining its rigidity. Finally, we iteratively deform the template towards target data, again using moving frames, to bring nearby pairs of template and target points closer together.

2. RELATED WORK

The problem of finding a good mapping of a template onto target data is inherently non-linear, and whatever the exact formulation, the optimization must be carefully guided to avoid local minima. In medical imaging, there is a huge literature for fitting "atlases" to two- and three-dimensional image data; for a survey see.[9] This is generally done by computing a flow describing a smooth deformation from the atlas space to the image space, while locally minimizing data error. To avoid being caught in very poor local minima, a multi-resolution strategy is applied: a low resolution deformation is computed, and used to initialize a higher resolution deformation, and so on.

Captured surface data, which is not dense, and which often has complex topology, noise, missing parts, and perhaps self-intersections, presents a different set of difficulties. When an obvious planar parameterization is available, flow methods can still be applied. Most notably, Blanz and Vetter[1] used a method like this to compute correspondences between cylindrically parameterized faces; the data consisted of an RGB color and cylinder radius at each pixel.

Another approach would be to reconstruct the surface, clean it up, and then apply a method which establishes correspondences between closed, manifold meshes. In this framework, as in the volumetric case, it is possible to

use multi-resolution, although considerably more ingenuity is required. Wang et al.,[10] for instance, established correspondences between brain surface models by constructing a geodesic triangulation between sparse user-supplied landmarks, and subdividing it. Schreiner et al.[11] take an approach with the same flavor to the more difficult context of pairs of surfaces with the same, but complicated, topology and very different geometry. They supplement a few user-supplied landmarks with others found by extensive topological analysis, and build a multi-resolution mesh hierarchy using geodesic triangles.

The alternative approach of fitting a closed, manifold template directly to "raw" captured data is very appealing, since it allows us to eliminate noise, fill holes, fix topological errors *and* establish correspondences, all with the same computation. For this approach there does not seem to be an obvious multi-resolution strategy, and to avoid local minima the optimization is usually initialized with more user-supplied landmark points. For instance, Allen et al.[2] use a template mesh to establish correspondences between the human scans in the CAESAR data set. This data comes with 76 landmarks, at most recognizable features of the model, establishing a very good initial correspondence. They used an iterative optimization, minimizing a combination of transformation smoothness, data fidelity and nearness of the landmark matches. Sumner and Popović[12] used a similar technique, on models which are geometrically more different, as part of their deformation transfer algorithm. Although fewer landmarks seem to be required, the quality of the correspondence need not be as exact in this application. In the SCAPE project,[13] a few user-placed landmarks were used, augmented with hundreds found automatically using the Correlated Correspondence algorithm,[14] to register human bodies in different poses. Like ours, the Correlated Correspondence algorithm automatically finds point correspondences using spin images[7] (discussed in more detail in Section 3.2). To resolve the difficult combinatorial problem of deciding which of the putative correspondences provided by the spin images are correct, they use a probabilistic belief-propagation model. We take a much simpler approach to using spin images; at the cost of a few more use-supplied landmarks, we can greatly limit the choices of possible correspondences, so that instead of resolving their inconsistencies combinatorially, we can simply average them.

Recently, Zhang et al.[15] introduced a similar approach of using deformable model for matching 3D shapes which differ in pose, scale, and geometric detail. Although, we use a similar technique, we find correspondence between similar shaped objects that are quite complex and can have non-manifold geometry. Huang et al.[16] also introduces a deformation based method for pairwise non-rigid registration of partially overlapping scans of the same object, in different poses. Unlike their method, we focus on generating multi-way correspondence between scans of several similarly-shaped objects.

Our method focuses on fitting a template to imperfect scanned data, possibly with complicated topology, and on reducing the number of user-supplied landmarks required.

3. BUILDING BLOCKS

Our method used two main tools: a differential deformation framework that preserves local rigidity, and a shape signature to automatically find matchings.

3.1 Mesh Deformation

The basic deformation engine of our method is based on the "moving frames" linear optimization described by Lipman et al.,[8] which produces an approximately "as rigid as possible" deformation. We define a local coordinate frame (Frenet frame) at each vertex x_i, with basis vectors b_i^1, b_i^2, b_i^3. Rather than storing the vertices x_i and their frames b_i explicitly, we store rotation matrices $\Gamma_{i,j}$ and displacement vectors $\alpha_{i,j}$ for every directed edge x_i, x_j, as described in Figure 2. Given some choice for one of the coordinate frames b_i, we can linearly solve for the others using the stored matrices $\Gamma_{i,j}$. And given the b_i, the $\alpha_{i,j}$, and the positions for a few of the x_i, we can linearly solve for the positions of the rest, giving a complete realization of the mesh.

Each of the two linear systems described above has a unique solution. If we add more constraints to either one, by assigning orientations to some of the b_i or positions to some of the x_i, we get an over-constrained system which we can solve in a least-squares sense (note that since the constraints cannot be solved exactly, the resulting coordinate frames b_i may not be orthonormal; this does not appear to have a large effect). We can add weights to any of the constraints in these systems, so that the influences of user-placed landmarks, automatically
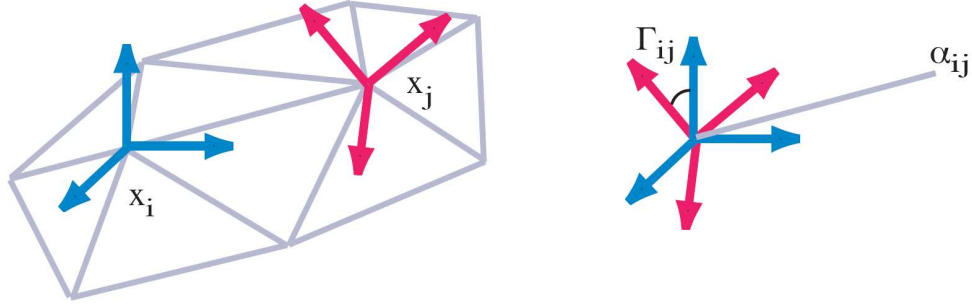
Figure 2. Definition of moving frames representation. Instead of storing x_i, x_j, and their respective coordinate frames (left), we store the rotation matrix $\Gamma_{i,j}$ between the two adjacent frames, and, at x_i, we store a vector $\alpha_{i,j}$, representing the position of x_j in the coordinate frame of x_i. We can reconstruct the original mesh from the Γs and αs by solving two linear systema, or, when we add displacement targets to the system, we can reconstruct a deformed version of the mesh.
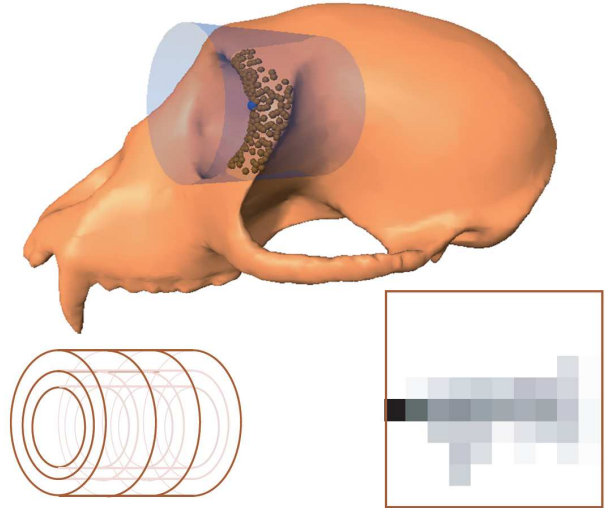

Figure 3. The spin-image at a point x_i takes into consideration the part of the mesh falling into a cylinder surrounding the surface normal. Vertices in the cylinder with similar-enough normals are accumulated into annulus-shaped bins (lower left), and the counts for each bin form a 2D histogram (lower right), the *spin image*, indexed by radius and height. Points on similarly-shaped patches of surface will have similar spin images (even if their normals are different).

chosen deformation targets, and the differential rigidity constraints provided by the framework can be balanced in different ways. Both the linear systems are very sparse and can be solved quickly.

As with other local differential frameworks, assigning many contradictory deformation targets can easily produce non-smooth deformations. Also, iterating this process - applying another deformation to a deformed mesh - can magnify the problem in areas where the deformation is not sufficiently smooth. Our algorithm copes with these limitations by making sure that deformation targets describe smooth fields of both rotations and translations.

3.2 Spin images

We use spin images,[7] a popular kind of localized shape signature, to select deformation targets. Spin images are high-dimensional vectors, summarizing the local shape of the surface around a vertex, in a format which is invariant with respect to rotations around the vertex normal. This ensures that even if the surface normals at two vertices are different, if there is a rigid motion which could align the local area around one vertex to the local area around the other, then the spin images at those vertices are the same. We use spin images of size 10x10 bins, where each bin is the size of the average edge length of the mesh.
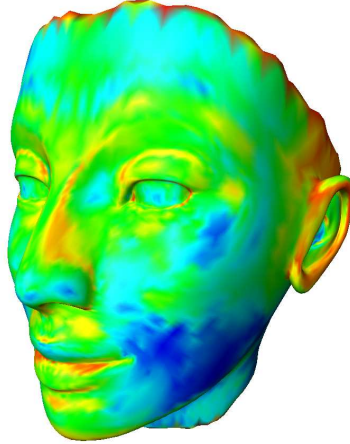
Figure 4. More distinctive vertices on the template mesh are red, less distinctive ones are blue. Distinctiveness is computed once for a given template. We only seek spin-image matches for distinctive vertices (red or yellow), since the others are likely to match any nearby vertex.

3.3 Initialization

Initially, we are given a clean, complete template mesh A and a target mesh B (possibly noisy, with holes, missing pieces, extra parts or topological errors such as self-intersections). The template can be constructed in various ways, for instance by using an artist-designed mesh, by manually cleaning up one of the input meshes, or, as in Figure 1, by using a dense set of user-supplied landmarks to co-register a subset of the meshes, computing a single average mesh, and then cleaning that up.

Distinctiveness: If the spin image at a point x on A is very similar to the spin images of all of its neighbors, spin images are unlikely to identify a unique or likely deformation target for x. As a pre-processing step, we evaluate the "distinctiveness" of the spin image at each vertex of A.

We generate the spin images as described in Section 3.2. We then compute the correlation coefficient of the spin image of x_i and that of each of its neighbors x_j (we use a 5-ring neighborhood for all our examples). If the correlation coefficient is below δ, which means that the spin image of x_i is different from that of x_j, we increment a "distinctiveness count" at x_i. Finally we normalize the distinctiveness by the total number of neighbors. Figure 4 shows a visualization of the distribution of distinctiveness on a face template mesh.

Rough registration We use a small number of user-placed landmarks to get an initial rough deformation of template A to target B. We compute the alignment in a standard way,[17] by first rotating, translating and scaling the landmarks on A to minimize their sum-squared error from the landmarks on B,[18] and then deforming A so that its landmarks exactly coincide with B's, using a thin-plate spline (TPS), which provides a very smooth deformation.

3.4 Spin image deformation targets

Next, we choose deformation targets by matching similar nearby spin images. We use only the vertices of A with normalized distinctiveness count greater than some constant (0.7, for our datasets). For each selected vertex x, we find its candidate matches by finding the 20 closest vertices on the target mesh B; we discard any whose normals differ too much from that of x. If the distance of x to the closest possible target vertex is more than 5ϵ, where ϵ is the average edge length, or if the number of possible targets is too small, we discard x.

We then compute spin images for the potential targets on B. From these, we would like to select a vertex y which is both close to x and which has a similar spin image. To combine these criteria, we normalize both the Euclidean distances d, and the correlation coefficients c to lie between zero and one, and then score each vertex y by a linear combination, $\beta c + (1-\beta)(1-d)$. We have use 0.5 for the value of β in all our examples. The target vertex y with lowest score is chosen as the target matching for x.
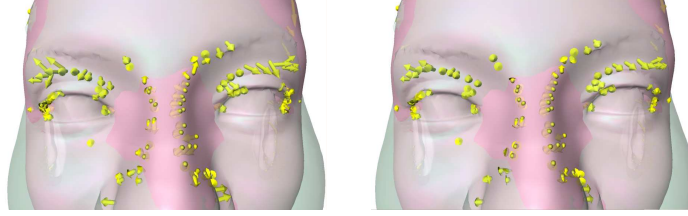
Figure 5. On the left, the original "best" spin image match for each vertex. On the right, the smoothed spin image translation vectors. Notice that the smooth vector field brings the ridges on the template towards the ridge on the target.

The target matchings are noisy, in the sense that neighboring template vertices on A can have very different targets vertices on B. For instance, spin images do a good job of matching ridge points to ridge points, but they do not discriminate well between points along a ridge. Each target vertex y provides both a rotation target (taking the normal at x to the normal at y), and a position target (making x coincident with y). To produce a smooth deformation, we smooth the rotation field and the translation vector field defined by the set of target matchings within 5-ring neighborhood of x. Figure 5 shows the target vectors before and after smoothing.

We add the smoothed targets to the moving frames optimization, and compute a deformed mesh A'. Since the set of targets is smooth, both in terms of coordinate frame rotation and in terms of vertex positions, the resulting deformation continues to be quite smooth. We recompute the rotation matrices $\Gamma_{i,j}$ at every edge, and the displacements $\alpha_{i,j}$ based on the mesh A', as described in Section 3.1, so that deformation continues from A'.

3.5 Deformation towards the target surface

The remaining step is to move points of the template in their normal direction (either inwards or outwards) until they lie on the target data. Because the target data may be incomplete, there will be template points that do not move onto the target surface, and these should smoothly deform so as to fill surface holes while retaining, as much as possible, the shape of the template.

For this step we iterate steps of matching and deformation, similar to the ICP algorithm. We are very careful to retain a smooth deformation as we iterate. Recall that our deformation engine solves for the deformation in two steps, solving a linear system for rotations and another for translations. Changing the rotation between coordinate frames implicitly changes the shape of the mesh. So matchings found using the original mesh are not necessarily good matches after the rotations have been updated. This motivates the following structure for each iteration:

1. Find matches between template vertices and points on the target surface.
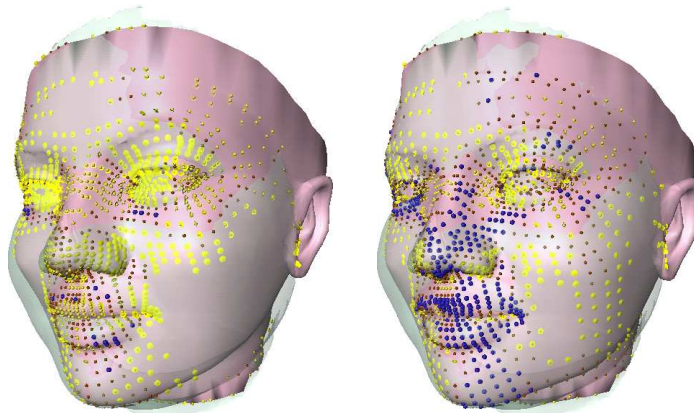

Figure 6. The template and the target data after zero (left) and twenty (right) iterations of normal matching and motion. Targets are yellow, anchors are red and statics are blue.

2. For each match, compute the rotation from the template vertex to the surface point.

3. Smooth this field of rotations

4. Solve for the frame rotations as described in Section 3.1.

5. Use these new frame rotations, and the static vertex positions, to solve for new positions for the template vertices.

6. Using these new vertex positions, find new matches on the target data.

7. Smooth the vector field of translations implied by the new matching.

8. Use the vector field as translation targets and solve for translational motions of the frames.

For both the frame rotations and the vertex positions, we cautiously choose the matches in which we have the most confidence. Let x be a vertex of the template which had not yet been moved onto the target data. At each iteration, x shoots a ray in its normal direction and intersects B in a point y. If y is within distance 0.5ϵ, where ϵ is the average edge length, and the surface normal at y is close enough to the surface normal at x, this is taken as an acceptable good target match.

At each iteration the template gets closer to the target. We remove vertices that are within distance 0.05ϵ to its target matching from the system, by making them either "statics", or "anchors". Statics are vertices that have moved close enough to the target and are removed from the linear system. They are surrounded by anchors, which are vertices in the system with very high constraint weights. Anchors maintain seamless transitions between vertices that moves and vertices that remain static. As more and more vertices move to their final position, the number of statics and anchors increases at each iteration (Figure 6).
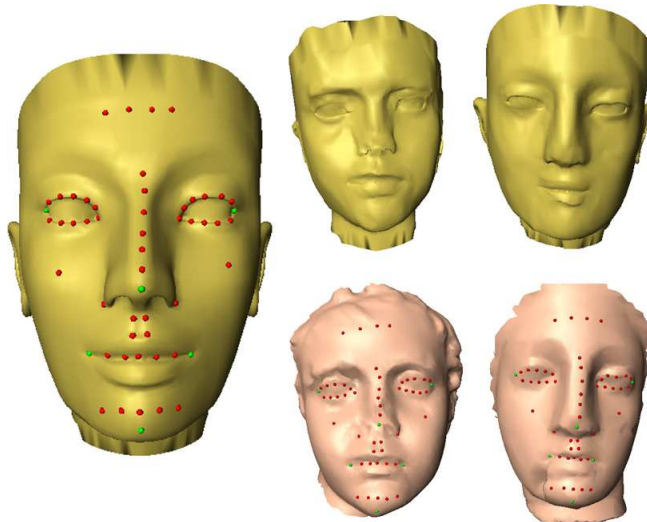
4. RESULTS

Figure 7. Similar points are matched on the two faces. The green points are placed manually on the template (cut out of an artist-created mesh). The deformation drags the red points to similar points on the target surfaces, showing that the deformation does a good job of preserving features such as the eyes, the groove between the lips, the bottom of the nose, and the philtrum (the groove on the upper lip).

We have applied our method to a variety of data. Left of Figure 1 shows a template mesh, obtained as an average of the target skulls.[19] Using a set of 27 user specified sparse correspondences (shown in green), and 30 iterations of our ICP based matching, we warp the template to each of the target meshes. We then transfer the dense landmarks (red) to the target shapes and generate a multi-way correspondence between the set of topologically different targets. non-manifold edges, holes, and

Figure 7 shows the landmark transfer from an artist-created model of the Nefertiti head to a couple of model heads; this time the target meshes are "nice" but their shapes have missing or extraneous parts. Here we used 6 sparse landmarks and 30 iterations to warp the template towards each of the target shapes.
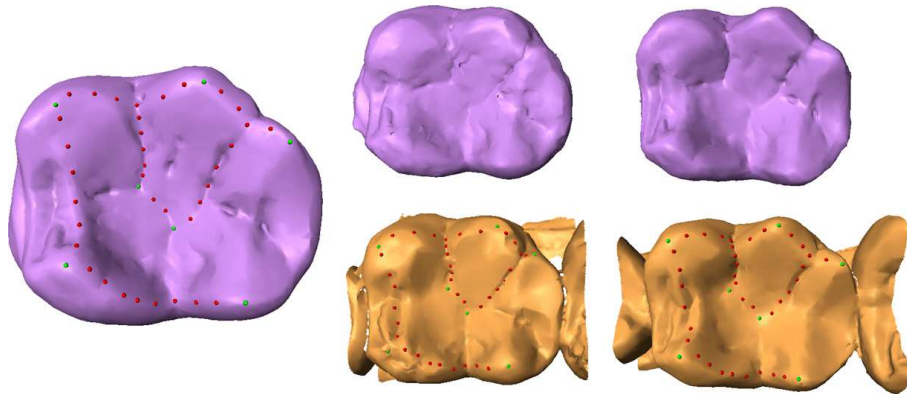
Figure 8. Scans of two molars, with a template made from the scan, by closing it off and filling holes in the data. The template gets matched nicely to the data, even without trimming extraneous pieces of surface, and again the red points are dragged onto similar features.

Figure 8 shows a pair of human molars on the right, and a template, created from another one of the scans, on the left. We use 7 sparse correspondences (green) for initial registration and warp. Again using 30 iterations, we are able to map the template to each of the targets. The template gets matched nicely to the data, even without trimming extraneous pieces of surface, and again the dense set of landmarks (red) are dragged onto similar features.
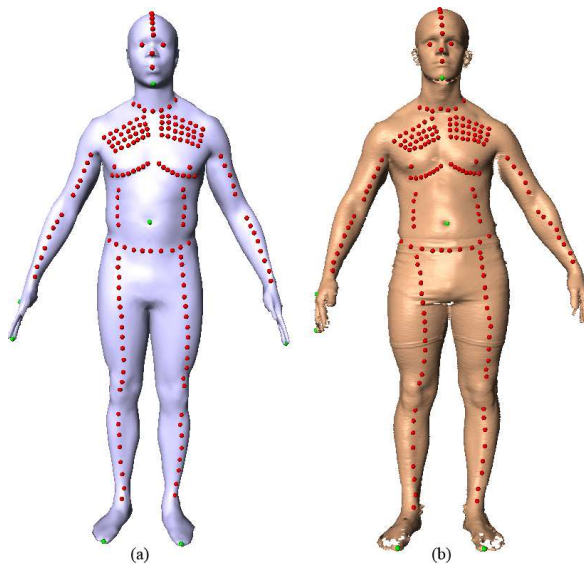

Figure 9. (a) Template with 11 sparse landmarks (green) and a set of dense landmarks (b) Target mesh with transferred landmarks (red). We are able to identify distinctive points without user-supplied landmarks to guide the deformation near it

We have also compared our result with Allen et al. on one of the models from the CEASAR data set. Figure 9 shows that using only 11 landmarks in contrast to the 76 supplied with the CEASAR data and used by Allen et al., we are able to deform the template and correctly identify and transfer distinctive landmarks (facial features, nipples), without any user-supplied landmarks. The corresponding deformations are shown in Figure 10.

5. CONCLUSION AND FUTURE WORK

We present an iterative method for generating dense correspondences between assemblages of surfaces, by warping a template to each of the targets using a very small number of user-specified landmarks. Using spin images we
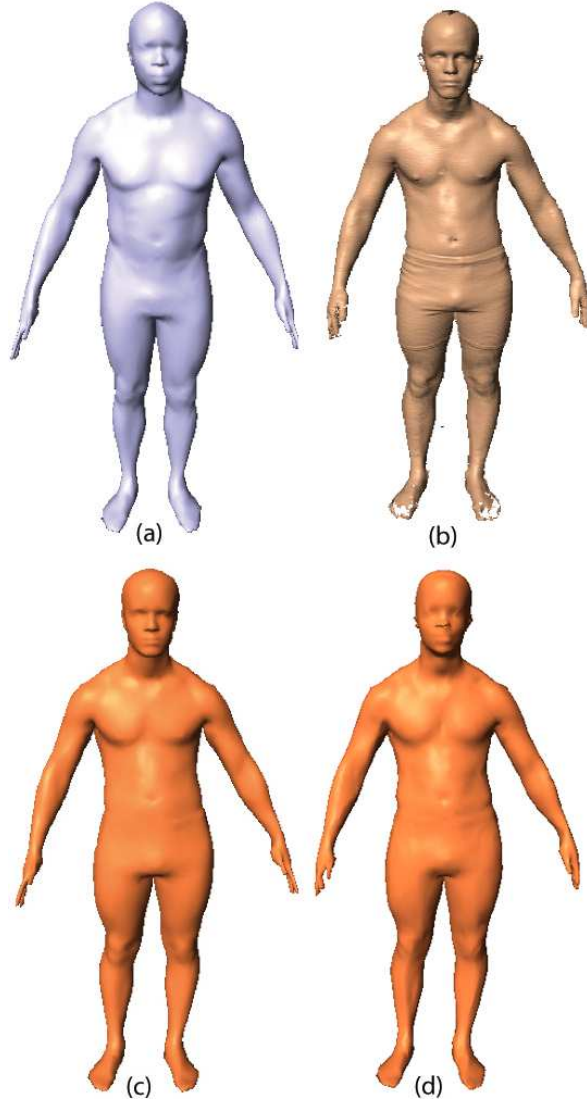
Figure 10. (a) Template (b) Target (c) Deformed template provided by Brett Allen, using 76 landmarks (d) Deformed template produced by our method, using 11 landmarks

automatically detect more correspondences on regions with distinct features, and further deform the template based on a ICP-like framework. While we do use the mesh connectivity of the target to find nearby points, a version of our algorithm that uses point data for the target seems straightforward.

A limitations of our deformation model is that it fails to maintain the rigidity of the mesh given a set contradictory deformation targets. We handle this problem by smoothing the deformation rotation and translation fields at each iteration. Another drawback is that the mesh-based representation does not contain any volumetric information. This might cause the deformation of the mesh to introduce self intersects in thin regions. However, there is nothing in the framework that requires the template to be a manifold. We have experiments with adding extra edges between the vertices of nearby surfaces to prevent these intersections.

REFERENCES

[1] Blanz, V. and Vetter, T., "A morphable model for the synthesis of 3d faces," in [*SIGGRAPH*], 187–194 (1999).

[2] Allen, B., Curless, B., and Popovic, Z., "The space of human body shapes," in [*SIGGRAPH*], 587 – 594 (2003).

[3] Smith, R. C., Pawliki, R., Kókai, I., Finger, J., and Vetter, T., "Navigating in a shape space of registered models," *IEEE Transactions on Visualization and Computer Graphics* , 1552–1559 (2007).

[4] Kilian, M., Mitra, N., and Pottman, H., "Geometric modeling in shape space," in [*SIGGRAPH*], 399 – 405 (2007).

[5] Pauly, M., Mitra, N. J., Giesen, J., Gross, M., and Guibas, L., "Example-based 3d scan completion," in [*Symposium on Geometry Processing*], 23–32 (2005).

[6] TF Cootes, CJ Taylor, D. C. J. G., "Active shape models: their training and application," *Computer Vision and Image Understanding* **61**(1), 38 – 59 (1995).

[7] Johnson, A. E. and Hebert, M., "Using spin images for efficient object recognition in cluttered 3d scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **21**(5), 433–449 (1999).

[8] Lipman, Y., Sorkine, O., Levin, D., and Cohen-Or, D., "Linear rotation-invariant coordinates for meshes," *ACM Trans. Graph.* **24**(3), 479–487 (2005).

[9] Maintz, J. and Viergever, M., "A survey of medical image registration," *Medical Image Analysis* **2**, 1–36 (1998).

[10] Wang, Y., Peterson, B., and Staib, L., "Shape-based 3D surface correspondence using geodesics and local geometry," in [*Proceedings of IEEE CVPR*], 644–651 (2000).

[11] Schreiner, J., Asirvatham, A., Praun, E., and Hoppe, H., "Inter-surface mapping," in [*SIGGRAPH*], 870 – 877 (2004).

[12] Sumner, R. W. and Popović, J., "Deformation transfer for triangle meshes," in [*SIGGRAPH*], 399 – 405 (2004).

[13] Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J., and Davis, J., "Scape: shape completion and animation of people," in [*SIGGRAPH*], 408 – 416 (2005).

[14] Anguelov, D., Srinivasan, P., Pang, H.-C., Koller, D., Thrun, S., and Davis, J., "The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces," in [*Proceedings of Advances in Neural Information Processing Systems (NIPS)*], 33–40 (2005).

[15] Zhang, H., Sheffer, A., Cohen-Or, D., Zhou, Q., van Kaick, O., and Tagliasacchi, A., "Deformation-driven shape correspondence," (2008).

[16] Huang, Q., Adams, B., Wicke, M., , and Guibas, L. J., "Non-rigid registration under isometric deformations," in [*Proc. of Eurographics Symposium on Geometry Processing 2008 (SGP)*], 1149–1458 (2008).

[17] Bookstein, F. L., [*Morphometric tools for landmark data: Geometry and Biology*], Cambridge Univ. Press, New York (1991).

[18] Horn, B., "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society A* **4**, 629–642 (April 1987).

[19] Wiley, D. F., Amenta, N., Alcantara, D. A., Ghosh, D., Kil, Y. J., Delson, E., Harcourt-Smith, W., Rohlf, F. J., John, K. S., and Hamann, B., "Evolutionary morphing," in [*Proceedings of IEEE Visualization*], 431–438 (2005).