Evolutionary Morphing

David F. Wiley* Nina Amenta* Dan A. Alcantara* Deboshmita Ghosh* Yong J. Kil*
Eric Delson‡ Will Harcourt-Smith‡ F. James Rohlf§ Katherine St. John‡ Bernd Hamann*
* Institute for Data Analysis and Visualization (IDAV) and Department of Computer Science,
University of California, Davis
‡American Museum of Natural History
§Department of Ecology and Evolution, State University of New York, Stony Brook

ABSTRACT

We introduce a technique to visualize the gradual evolutionary change of the shapes of living things as a morph between known three-dimensional shapes. Given geometric computer models of anatomical shapes for some collection of specimens - here the skulls of the some of the extant members of a family of monkeys - an evolutionary tree for the group implies a hypothesis about the way in which the shape changed through time. We use a statistical method which expresses the value of some variable - in this case the shape - at an internal point in the tree as a weighted average of the values at the leaves. The framework of *geometric morphometrics* can then be used to define a shape-space, based on the correspondences of landmark points on the surfaces, within which these weighted averages can be realized as actual surfaces.

Our software provides tools for performing and visualizing such an analysis in three dimensions. Beginning with laser range surfaces scans of skulls, we use our *landmark editor* to interactively place *landmark points* on the surface. We use these to compute a *tree-morph* which smoothly interpolates the shapes across the tree. Each intermediate shape in the morph is a linear combination of all of the input surfaces. We create a surface model for an intermediate shape by warping all the input meshes towards the correct shape and then merging them together. Our merging procedure is novel. Given several similar surface meshes, we compute a weighted average between them by averaging their associated trivariate squared distance functions, and then extract the *extremal surface* which traces out the "valleys" along which the averaged function is nearly zero.

CR Categories: I.4.10 [Image Processing and Computer Vision]: Image Representation—Morphological; I.3.8 [Computer Graphics]: Applications

Keywords: morphometrics, morphing, surface blending, merging, warping, distance fields, extremal surface

*e-mail: wiley@cs.ucdavis.edu, amenta@cs.ucdavis.edu, kil@cs.ucdavis.edu, ghosh@cs.ucdavis.edu, dfalcantara@ucdavis.edu, jonesj@cs.ucdavis.edu, hamann@cs.ucdavis.edu

1 INTRODUCTION

Darwin's theory of evolution was originally applied using morphology - discrete qualitative features such as number of toes, and also quantitative shape differences, such as elongation of a limb - to place species within the tree of life. While genomic sequence data is now the basis of most phylogenies (evolutionary trees), morphology continues to be an essential part of evolutionary biology. One important reason is that the morphology of fossils, rather than comparisons between genomic data, provides our only direct evidence for extinct species.

For instance, the shape of the skull is very important in the study of human evolution, and that of our primate relations. Quantitative differences, such as shape of the cranium or brow ridge, are essential in defining the criteria for comparison between skulls. The idea that qualitative shape differences can be analyzed in terms of the transformations required to "morph" one shape into another goes back at least to D'Arcy Thompson's classic 1917 book *On Growth and Form*, from which Figure 1 is taken.

The statistical analysis of geometric shape transformations is the program of *geometric morphometrics*. In addition to evolutionary biology, morphometric techniques are also used widely in developmental biology, medical image analysis, and other areas.

We describe a three-dimensional *tree-morph* visualizing the evolutionary changes implied by a given evolutionary tree. Surface meshes captured by a laser range scanner from cranial specimens for the existing species appear at the leaves. The interior nodes and interior points of the edges of the evolutionary tree correspond to hypothetical ancestor species. These we visualize by computing synthetic surface meshes for the shapes at the internal nodes and at a dense set of points sampled along the edges. We can "play" the morph by displaying the precomputed meshes, either interactively, by sliding a cursor along branches of the tree, or as an animation.

This project goal was intended as an example application to drive the development of computer tools for three-dimensional morphometric analysis and visualization. A major contribution of this paper is that it introduces a challenging collection of visualization

problems to the scientific visualization community. Three aspects of the project deserve particular attention. The first is our interactive *landmark editor*, which makes it easy for a user to place dense sets of landmark points on input models. The editor facilitates an essential (and very tedious) step in traditional morphometric analysis, and it has already had additional impact beyond our application; it is currently being applied to additional problems by the paleontologists on our team. Second, the multiple-alignment and interpolation procedures we use, while standard in morphometrics, differ from those common in computer graphics in interesting ways; in particular, they are carefully designed to cause the differences measured or displayed to reflect, as well as possible, the intrinsic shape differences between the input objects and to avoid introducing artifacts into the transformation, even visually pleasing ones. The third

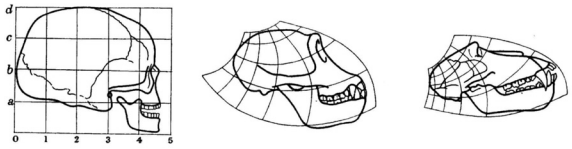

Figure 1: Spatial relationship between human, chimpanzee, and baboon skull, as envisioned by D'Arcy Thompson in 1917. The overall shape is recognizably similar, and the transformation between them describes the shape difference. In modern morphometric studies, the statistical analysis of the transformation is based on a matrix of selected landmark points from the surfaces, while warps of the embedding space, such as the one pictured here, are more often used for visualization of the results.

contribution is our novel method for merging several similar input meshes, with weights, which is designed to handle non-manifold surfaces with boundaries. Our merging procedure is based on a functional representation of the surfaces using the trivariate squared distance function rather than the more usual signed distance function, and extracts an *extremal surface* from the blended distance function. Extremal surfaces have previously been used in the context of point-set surface modeling. Our method does not require implicit, closed or oriented surfaces as input.

1.1 Geometric Morphometrics

Geometric morphometrics is a branch of biostatistics dealing with the analysis of shape. Some good references are the "classic" text by Bookstein [4], a survey article by Adams, Rohlf and Slice [1], and an accessible recent textbook by Zelditch et al. [21].

Scientists need to be able to define and analyze statistically significant variables expressing object shape. This task is difficult because the choice of what to measure and analyze affects the results. Rather than measuring specific distances, angles, and so on, the approach used in geometric mophometrics is to choose a discrete set of homologous (or "corresponding") *landmark points* on all of the object surfaces. The shape is then represented by its set of landmark points; this representation subsumes measurements of specific distances between landmarks, angles produced by three landmarks, etc. A dense enough set of landmark points provides a sampled representation of the shape.

The *Procrustes distance* between pairs of landmark sets is defined as the square-root of the sum-squared distance between all pairs of corresponding landmark points, when the two landmark sets are scaled, rotated and translated so that this distance is minimized.

This distance imposes a geometry on the space of landmark configurations. Unfortunately, the interpolation we need for our tree-morph is complicated by the fact that the pairwise alignments do not produce a multiple mutual alignment of all the landmark sets; if we align A to B and B to C, we find that A is not aligned to C. The common practice is to iteratively compute an averaged *consensus configuration* of the landmark points which minimizes the total squared Procrustes distance from all of the input landmark sets, and then all of the landmark sets are aligned to this consensus configuration. This process is called *General Procrustes Alignment* (GPA).

When applying this process to two-dimensional images, the averaged shape represented by the consensus configuration is often visualized by warping one of the input images so that the input landmarks are brought into coincidence with the consensus landmarks. This is typically done using a thin-plate spline (TPS), which optimizes a particular smoothness criterion.

1.2 Application to Primate Evolution

We use this well-accepted morphometric framework to define the "correct" interpolation of the skull shapes for some species of Old World monkeys. The Old World monkeys evolved in the same time and place as early humans, making them a particularly interesting group to study. There are many extinct species of Old World monkeys, known from fossils, so that there is a lot of interesting data related to their evolutionary history. Yet this history, defined as the exact shape of the evolutionary tree, continues to be a matter of controversy.

We have used a laser range scanner to capture the three-dimensional shapes of the skulls of many species of Old World monkeys, both extant and fossil, as part of a larger effort to develop a database of three-dimensional primate morphology. We use some of this data to provide a method for visualizing some of the morphometric estimates of skull-shape variation which are relevant to the evolution of the group. We selected five extant species sampling both subfamilies of Old World monkeys, and a best-estimate of an evolutionary tree of the five, derived from DNA sequence data, which is available only for the extant, but not the extinct, species. We then visualize what the sequence-based tree implies about the morphology of ancient monkeys by interpolating the skull shapes across the tree. Figure 2 shows the tree.

Visualizations of the *intermediates* (the hypothetical species at interior points of the tree) are interesting in at least two ways. Scientists want to answer questions like, "Are the skull shapes predicted by this model biologically plausible?"', and "Where would this known fossil fit into the tree we hypothesize from genomic data?" The visualizations of the subset of the skull shape-space defined by the tree help to answer both kinds of questions.

This project is one specific example in which visualizing the three-dimensional morphs defined by morphometric theories can be used in biological applications. We believe that similar shape-space visualizations could be used for, among other examples, verifying models of developmental biology , characterizing degenerative disease processes such as Alzheimers, and the study of the change of shape as a function of size (allometry).

1.3 Technical Overview

Our goal was to produce a three-dimensional tree-morph visualizing the evolutionary hypothesis presented by a specific evolutionary tree using input models captured by a laser range scanner. The output of our procedure is a set of polygonal surface models, each one representing an intermediate shape corresponding to an interior point of the tree. Each of these intermediate models is a weighted

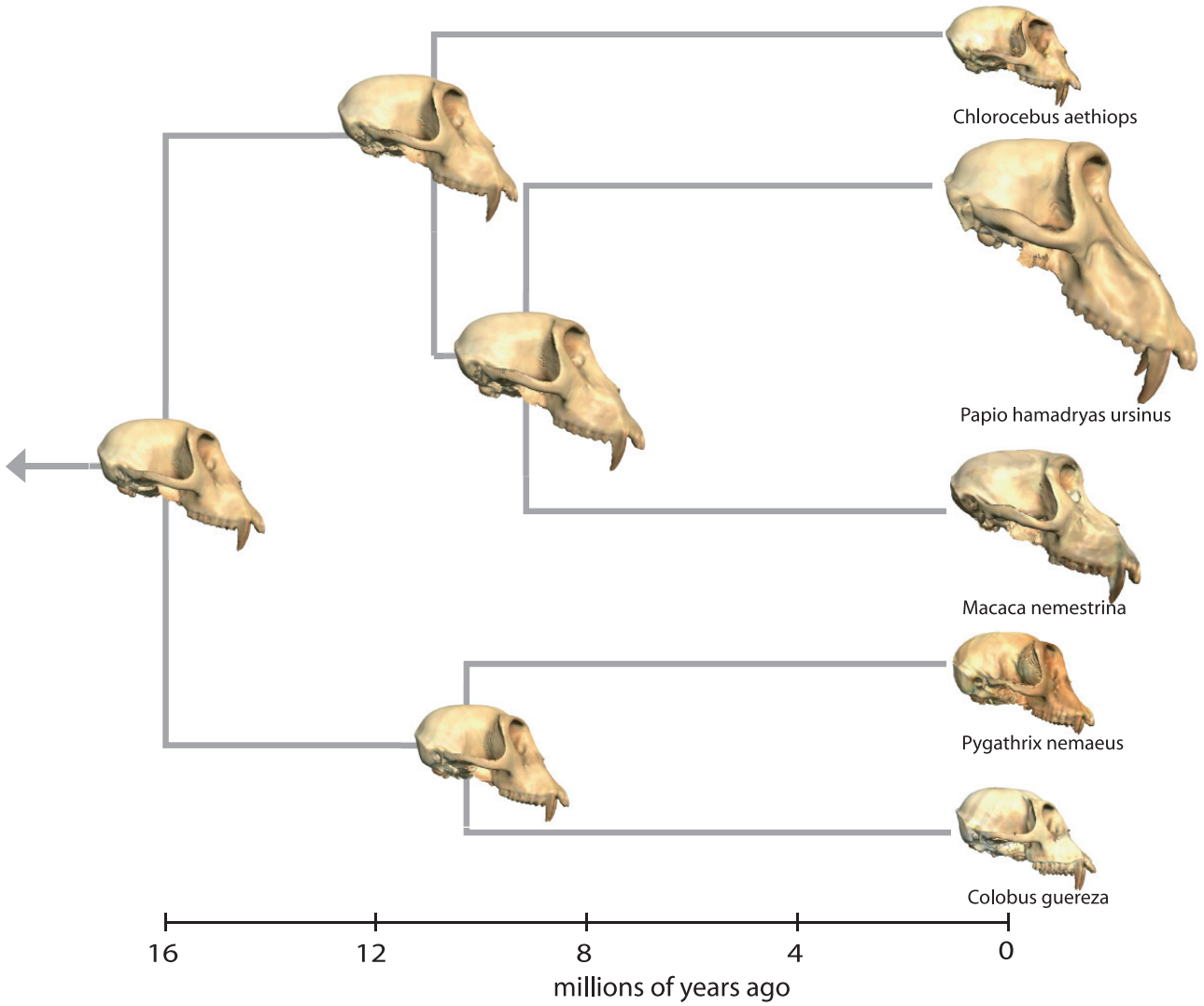

Figure 2: The input surface meshes, from laser range scans of the crania of existing monkey species, are shown on the right at the leaves of this tree. The surface meshes at the internal nodes, placed at the estimated dates at which the species are hypothesized to have diverged, represent the skull shapes of the hypothetical ancestors. They were produced by our morph, as weighted averages of the input surfaces. There is an interpolated shape corresponding to each point within an edge as well.

average of the input models; they differ only in the choice of the weights, which are computed as described in Subsection 2.2.

We faced two main challenges. First, we needed to carefully incorporate all of the theoretical assumptions commonly used in morphometric analysis into our tree-morph, so that the results would accurately reflect the statistical analysis, and be credible and interesting to primatologists. Second, we needed to handle the captured skull models, which are not manifold, have holes and occasional self-intersections, have color information, and are reasonably large (over a half-million triangles each). We developed the following procedure.

1. **Landmark entry:** The user interactively places landmark points at biologically meaningful locations providing homologous points on each of the input specimens.

2. **Alignment and target computation:** For each set of weights, we simultaneously align the landmark sets to each other and produce a weighted average target configuration of the landmark points, using a weighted version of the iterative Generalized Procrustes Alignment procedure mentioned above.

3. **Warp:** We compute a TPS warp from each input surface so that its landmarks coincide with the target configuration, and warp all surface vertices into the target space.

4. **Blend:** We compute a narrow-banded distance field around each surface and produce a weighted average squared distance field. Extracting the *extremal surface*, which lies along the "valley" of this function, produces the output surface.

Each of these steps is described in more detail in the following sections.

Our method is applicable to rigid objects; it is not intended to handle morphs in which the conformation varies as well as the shape, such as a moving arm or leg.

1.4 Related Work

Existing geometric morphometric software has mainly focused on the alignment and multivariate statistical analysis of specimens, with less emphasis on either landmark placement interfaces or visualization. *Morpheus* [18], *morphologika* [14], and the *TPS* suite of programs [15] are the packages most widely-used by morphologists.

Placing landmark points on 3D specimens for morphometric analysis is generally done using 3D contact digitizers on the actual objects, with the collected points automatically stored into a spreadsheet. This is extremely tedious, so that landmark sets consisting of tens of points are typical. For virtual 3D images of specimens, such as laser range surface or CT scans, there are generic software packages that allow the user to collect points and save them into an output file. These programs are not specialized to the landmark placement task, so the process remains quite cumbersome even for digital data.

The interesting visualization problem of morphing primate skull shapes across an evolutionary tree was first approached by Delson et al. [7] using the three-dimensional analog of the practice from morphometrics mentioned above, in which the transformation from one shape to another determined by the landmark points is visualized by warping one of the input surface models. This approach has the drawback that the visualization of the intermediate produced by warping one input surface is not the same as the visualization produced by warping another instead. Our work improves on this first approach in that our novel surface merging technique allows us to produce a single surface mesh which accurately represents the desired proportions of the input shapes. Also, since we can generate many more landmarks, we achieve a better representation of the shape and its variation.

When the shapes are captured by computed tomography rather than laser range scans, the trivariate density functions for the different specimens can be blended, after warping to align significant features. This idea has been applied to visualizing the evolution of toads by Hodges et al. [9]. The problem of merging similar surfaces is replaced, in this case, with the problem of isosurfacing as the function is averaged across time, which is also non-trivial. We believe that producing more landmarks is one way this process could be improved.

We also draw on methods known in computer graphics and visualization. We were inspired by one particularly relevant project [2], in which a collection of full-body scans of humans was aligned to a closed synthetic "base mesh". The base mesh could then be warped to resemble any of the inputs, or a linear combinations of the inputs. This method produces a "space of human body shapes" useful in computer graphics, for instance for generating crowds of digital extras. In morphometrics, there is a strong emphasis on producing results which are derived from the data rather than introduced for computational convenience, so we wanted to avoid the synthetic base mesh; also, we had no appropriate mesh to use.

Instead, we use a warp-and-merge paradigm to produce the intermediate surface models. Our method is closely related to earlier work on morphing using implicit functions [6] (and see also [19]). Coinciding with the common practice in morphometrics, this method uses the TPS to warp models so that they resemble each other closely. Surfaces are then merged by converting each input surface into a signed distance function defined over a finite three-dimensional domain, taking a weighted average of the functions, and extracting the zero-set of the resulting function. An implicit method, in which each surface is represented as a trivariate function, is appealing for this application since we can take a weighted average of functions in a straightforward way. The existing method works well for closed input surfaces. Our inputs, however, do not have well-defined interiors, which makes it more difficult to define a signed distance function. Therefore we have developed an alternative method based on the squared distance function.

2 METHOD

2.1 Landmark Placement

An essential part of the project was developing the landmark editor, an interactive tool with several specific user interface features to help users generate and place landmarks. A basic but important feature is that the homology between the landmarks on two input surfaces is shown explicitly; with conventional methods, the user had to imply the homology by carefully selecting landmarks in a specific order. In the landmark editor, two surface meshes are shown at the same time in the main window. Figure 3 shows a screenshot. A second text window shows the correspondences between pairs of landmark points; the windows are linked, so that selecting a correspondence in the text window highlights the selected points on the surface meshes, and the user can see that they are indeed homologous. Points can of course be added, deleted and adjusted in any order.

Conventional tools use only single points to denote landmark features, which are typically placed at biologically distinctive homologous points on each specimen. We show the surface normal as well as the point itself as the user adjusts the landmark, which helps to place it exactly, especially on high-curvature features.

Not all shape differences can be captured in this way. To capture curvature or the area in smooth regions we also want to distribute points on these surfaces which can be placed in correspondence; such points are called *semi-landmarks*. We use curve and patch primitives to place large numbers of semi-landmarks easily. These primitives are three-dimensional Bezier curves or patches, with control points placed by the user on the surface. After the user has placed a curve or patch, the system generates a user-controlled number of semi-landmark points to quickly and accurately cover the primitive. The semi-landmarks are distributed as a sequence or grid, and then projected onto the surface. The orientation of curves and patches is shown with arrows, since it is easy to get these swapped; and the user can re-orient a patch or curve to correct a mis-match without having to move the points.

Consider, for example, the task of placing points around an eye socket. The conventional method is to manually place dozens of points, in a particular order and spacing, around the socket. With the landmark editor, the user only needs to place a few points to define curve primitives around the socket.

Bookstein [5] introduced a method for optimizing the positions of semi-landmarks on a curve or surface, minimizing the energy measure of the induced TPS warps. We have implemented this method, and it seems to have minimal effect on our semi-landmarks, which are generally well-spaced to begin with. We have not yet integrated this optimization step into the system.

We have also implemented a method that transfers landmark primitives from one surface to another semi-automatically. The user places enough landmarks to produce a crude warp, which is then used to transfer the rest of the landmarks; the user then has to adjust their positions, but just transferring the overall configuration simplifies the experience and reduces errors. We also export the

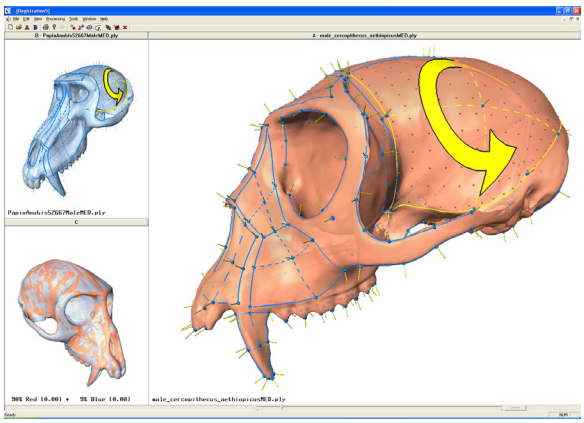

Figure 3: Screen capture of our landmark editor. Two input meshes are shown in the large pane and the upper left, while the two warped models are overlayed in the lower left. The yellow arrow indicates the patch orientation.

points generated by our process, which allows existing morphometric packages to use them. Figure 4 shows an entire set of exported landmark points for one of our crania.

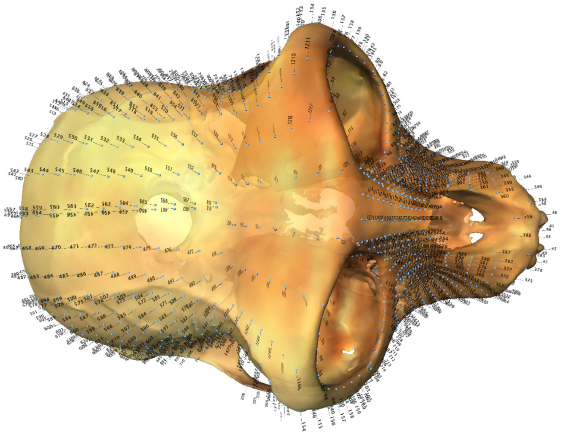

Figure 4: A full set of 853 landmarks placed on one of the scanned crania. These were created using 45 are single points, 32 curves and 9 surface patches.

Using this interface, it is easy to create large sets of correctly corresponding landmark and semi-landmark points. On each cranium in our example application, we use 853 landmarks. While placing these was indeed tedious (it took our novice users about three hours per skull), it would have been completely infeasible using previous methods.

Collecting landmarks and semi-landmarks on object surfaces is the first step in virtually every three-dimensional morphometric analysis, so our landmark editor will be useful in many projects beyond our visualization application. It is currently being used and tested by the primatologists on our team for a separate research project investigating congruence between joint surfaces in the primate skeleton, in which a grid of points are placed on the opposing joint surfaces. So far landmarks have been collected on laser range scan surfaces of over 80 primate lower limb-bone specimens, and results are being analyzed. The software is greatly facilitating an otherwise lengthy and complex process.

2.2 Weights

Each internal point of the tree corresponds to an intermediate skull shape, which is a weighted average of the skull shapes at the leaves. The weights are determined using a principle known in evolutionary biology as *squared-change parsimony*: the integral of the squared change of the variable over the tree should be minimized, within the constraints imposed by the values of the variable at the leaves. This principle is sometimes used to estimate the structure of the evolutionary tree from the values at the leaves [20]. Here we are concerned with the easier problem in which the structure of the tree, including the lengths of the branches, is given, and we just want to compute the values of the variables at the internal nodes.

Minimizing the squared change [can be implemented using] a generalized least-squares computation, in which the influence of each leaf on an internal node is weighted using a covariance matrix derived from the structure of the tree. Assuming a simple *Brownian motion* model of evolution,] the the variance of the difference between two points on an evolutionary tree is expected to be proportional to the amount of evolutionary time separating the two points.

The formula derived from this principle is given in a paper by Rohlf [16] (Equations 16, 17). We use this formula to assign, for a given internal point in the tree, a set of weights $w_1, \ldots, w_k$ for each of the k surfaces.

2.3 Alignment and Target Configuration

Given k sets of homologous landmark points, and weights $w_1, \ldots, w_k$, we now want to compute the surface corresponding to an internal point of the tree.

The first step in this process is Generalized Procrustes Alignment, an iterative procedure that simultaneously determines the target configuration for the landmark points (the positions of the landmarks on the output surface) and aligns the input landmark sets to the target configuration. This procedure, due to Gower [8], is fundamental to morphometrics; it is used to separate the difference in the landmark configurations due to actual shape differences from that due to differences in scale, translation and orientation.

We begin by scaling each input set of landmark points so that the sum of the squared distances between all of the points and the center of gravity is one. The scale variation can be re-introduced into the visualization at the end, as in Figure 2.

We pick one of the input configurations of landmark points, arbitrarily, as the first iteration of the target configuration. Then in each iteration, we align all of the input landmark sets to the proposed target configuration. We use Horn's algorithm [10], which gives an efficient, closed form solution to the problem of finding the rotation and translation of an input landmark set minimizing the sum of squared distances between each landmark point and the homologous point in the target configuration. After aligning all input landmark sets, we compute the new target configuration by taking the weighted average of all of the homologous copies of each landmark point. We terminate the iterative process when the target configuration is "stable". Fewer than ten iterations are typically needed.

2.4 Warp

The next step in the process is to warp each input model so that its landmark set coincides with the target landmark configuration. Following the common practice in morpometrics, suggested by Bookstein [4], we use the TPS defined by two sets of landmark points to

warp each input surface mesh. Since we use a dense set of landmarks, the warp brings the input models very close together.

The TPS defines a deformation of three-space which takes one set of landmarks into another. Of all such deformations, the TPS is the one that minimizes an energy functional based on a notion of "bending energy", related to the curvature of three-space induced by the warp. Conveniently, it is possible to get a closed-form solution for the TPS warp, since it can be expressed as a weighted sum of radial basis functions centered at the landmark points. The weights are determined by solving a linear system, which in this application can be done in a straightforward manner.

2.5 Merging

All of the warped input surfaces are close to each other. The gross differences between input models have been eliminated, and the remaining task is to produce a single output surface which averages the remaining differences, again using the weights $w_1, \ldots w_k$. Using the weights in the merging step means that near a leaf node of the evolutionary tree, the data at the leaf dominates not only the shape to which the surfaces are warped but also the proportions in which they are blended. As a result, the space of shapes produced in the morph exactly interpolates the input shapes.

2.5.1 Averaged Squared Distance Function

We compute the squared distance field on a uniform voxel grid around each input mesh M_i, to within a small distance α (we use 1/8 the largest dimension of the model). Since the surfaces are very close together, many voxels will be farther than α from any surface and we will never write to them. We save memory by breaking the volume into $4 \times 4 \times 4$ blocks, and keeping a low-resolution version of the entire volume in which each location addresses one block and points to the actual storage for the block. Only when we first write to a block do we actually allocate its storage; for blocks to which we never write, the pointer in the low-resolution volume remains null.

We compute the squared distance from each voxel within distance α to the surface exactly using the closest point transform (CPT) code distributed by Mauch [12], which implements his robust computational-geometry-based method. This code finds for each voxel w the nearest *foot-point* on the polyhedral input surface: a foot-point is a surface point x such that a sphere centered at w is tangent to the surface at x. The nearest foot-point is also the closest surface point. We modified the code to find not only the closest foot-point but also the second-closest as well, for reasons described below.

We use the CPT code to find the trivariate squared distance function d_i for each M_i, and also the exact gradient ∇d_i of the squared distance. Note that the gradient is the same as the unsigned distance function, times two. Averaging the d_i produces a single scalar trivariate function d, and, because of the linearity of the derivative operator, averaging the ∇d_i produces its exact gradient ∇d.

Color is also averaged. Each triangle in the input mesh is assigned the average colors of its three vertices (we could have interpolated the vertex colors across the triangle but the differences are negligible at the grid resolution we are using). The color c_i of every voxel closest to this triangle then inherits this triangle color. The colors at the voxels are averaged to produce the average color c. After the surface is extracted, color is assigned to each of its vertices by interpolating the surrounding voxels.

2.5.2 Extremal Surface

The squared distance d_i from one input surface M_i is zero at the surface, and its gradient is the zero vector, but the average d is not exactly a squared distance function - it has small, but not zero values, near the input surfaces - and its gradient is exactly zero only a set of discrete minimum points. Our desired output surface lies along the two-dimensional valley in d near where the input surfaces were all zero. To extract a surface from the valley, our basic approach is to consider the directional derivative of d, in a direction v roughly perpendicular to the input surfaces. Finding this direction v at every point is the tricky part of the process, which we describe in following section. Once we have an appropriate vector field v, the directional derivative

$$g = \partial d / \partial v = \frac{\nabla d \cdot v}{2}$$

is a signed function, whose zero set is taken as the desired output surface. See Figure 5.

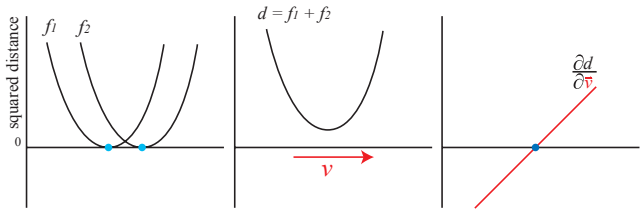

Figure 5: Averaging multiple squared distance functions produces a function which is similar to a squared distance function, but generally not zero anywhere. In one dimension the squared distance function is a parabola in two-space, and the average of several is also a parabola. In higher dimensions the situation is similar, although complicated by the fact that the input surface normals do not match exactly. Taking a directional derivative in a direction v roughly perpendicular to the desired surface produces a signed function, with its zeros defining the bottom of the parabolic "valley".

This approach is a variant of the *extremal surface* construction, which was developed for extracting surfaces from noisy tensor fields [13] and used recently for point-set surfaces [3]. Since the function g is locally nearly linear, extracting the output surface using marching cubes [11] works well and produces smooth surfaces free of "jaggy" artifacts (this method can fail near very thin features or surface self-intersections, where we do see some typical marching cubes artifacts).

2.5.3 The Vector Field

To produce the vector field v, we average the ∇d_i as *unoriented* rather than oriented vectors. This kind of averaging makes sense only when the vectors at a voxel are "nearly parallel" to each other. Specifically, we require that the smaller angle between the two lines supporting any two vectors d_i at the voxel is at most $\pi/4$; otherwise the voxel is "divergent". In the usual case in which the vectors are "nearly parallel", we re-orient the vectors so that every pair has positive inner product; we can then average the re-oriented vectors to produce v. The effects of this averaging are illustrated in Figure 6.

Divergent voxels are handled differently, as discussed in subsection 2.5.5.

2.5.4 Consistent Orientation

Observe that if, in the example in Figure 6, we orient all of the vectors in v to be pointing upwards, and then compute the signed

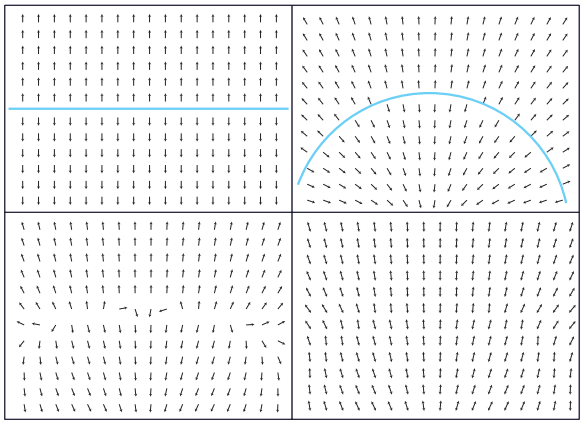

Figure 6: The gradient vector fields ∇d_i for two similar surfaces are shown at the top. The usual vector average of the ∇d_i produces the average gradient field ∇d on the lower left; notice that the vectors converge to a couple of minimum points. On the lower right, we show v, formed by averaging the g_i as unoriented vectors. Notice that it remains nearly perpendicular to both the input surfaces.

function $g = \nabla d \cdot v$, we find that g is negative at the bottom of the picture and positive at the top, and the zero crossing occurs in the region where ∇d is zero or nearly parallel to the input surfaces.

The next step in our process is indeed to find a consistent global orientation for v. While the original work of Medioni et.al. on noisy data suggested that a consistent orientation is not strictly necessary (see [13], Appendix C), we found that in our application a consistent orientation is required to get reasonable results.

Our approach is to propagate an orientation through the voxel grid, beginning by assigning an arbitrary orientation at some voxel at which v is defined. We then propagate this orientation to its neighbors, then onto their neighbors, and so on. The sequence in which we propagate is essential to achieving a consistent global orientation. This order is determined by a putting the neighbors into a priority queue and orienting the highest priority neighbor first.

The priority function we use combines two sub-functions. The first is designed to avoid propagating the orientation across the medial axis, so that the orientation of v agrees near the surface and "flips" in orientation are located near the medial axis. So we define a priority sub-function $p_1 = d$, with lower values having higher priority; in other words, we propagate along the valleys first, and then outwards.

The other issue is that at self-intersecting parts of the surface, near very thin parts or near sharp edges, the orientation tends to flip while propagating. We therefore prefer to orient these regions last, after the "easy" regions have been handled. Detecting these regions is the reason we modified the CPT code to also find the *second* closest foot-point. Near thin regions, self intersections and sharp edges, the closest foot-point f_1 and the second closest foot-point f_2 are near each other, and in "easy" regions they are far apart. We define a second priority sub-function $p_2 = ||f_1 - f_2||$ for every input M_i, and average the values of p_2 when we average the surfaces.

It was not really clear to us how to combine these two priority sub-functions p_1 and p_2 to order the priority queue. After trying various methods and examining the results, we settled on scaling them both so that the values are comparable, and then averaging them. This means that the highest priority voxels are high priority according to both functions.

2.5.5 Clean-up

The resulting surfaces still has some extraneous parts. One issue is that the two vector fields v and ∇d are perpendicular to each other near ridges of d as well as valleys, that is, the zero-set S includes the medial axis of the desired surface as well as the surface itself. We handle this simply by taking the single largest connected component of S and deleting the rest; this removes the medial axis components and also some small noise artifacts near the surface.

Another issue is that the vector field v is always pointing inwards towards a boundary edge. The orientation of v has to flip somehow as it goes around the edge, which introduces a spurious zero-crossing in g. These we detect by considering $|g|$ as well as the signed value of g; if $|g|$ is large we are far from the surface and we do not include surface components from that voxel.

At divergent voxels v is undefined and we do not compute g directly. Instead, we compute g for the surrounding voxels and then interpolate the values, by averaging, to fill in the divergent voxels. We also tried a least-squares fitting, but found that it did not improve the results. Note that this would not be possible without a consistent orientation for v and hence a consistent sign for g.

3 Results

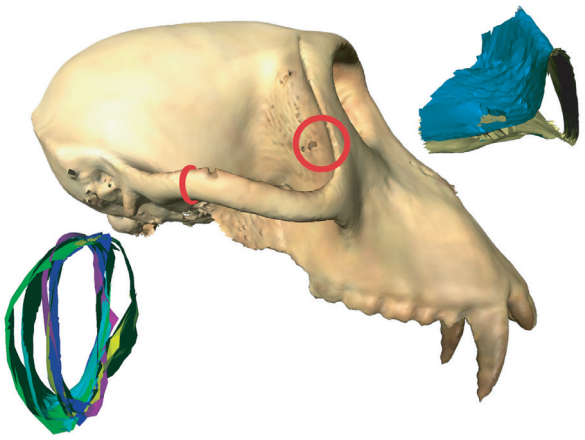

Figure 7: The extremal merging of five warped input meshes. The insert on the lower left shows a cross-section taken from the cheekbone, showing the variation between the five input meshes in that area (one of the more difficult for us). The "jaggy" artifacts behind the eye sockets occur because the thickness between two parallel surfaces is less than the resolution of the voxel grid; in this case, some of the input meshes are actually self-intersecting, as shown in the insert at upper right.

Figure 2 illustrates our results. The main point is that the synthetic skulls, created by averaging the input meshes, are virtually indistinguishable from the original models. A video, including an animation of the tree-morph and some examples of interaction with the landmark editor, accompanies this paper.

The input surface meshes varied in size, from 797K to 433K triangles, except for the *Papio* model, for which only a medium-resolution 75K triangle mesh was available. Computing the trivariate distance function from the input mesh is the most expensive part of the computation, and this is roughly linear in the size of the input mesh. For the animation, we simplified all of the meshes down to about 75K triangles, since more detail would not be resolved at

video resolution. Note that the trivariate distance function has to be computed for each frame, since the warp for each frame is different. We used the full resolution input meshes for the figures.

We did our processing on four Intel 3.2GHZ Hyperthreaded workstations, each with 2GB of memory. The distance field for the high-resolution meshes is computed in about 500 seconds per model on a voxel grid of size $300 \times 192 \times 147$, and for the lower resolution meshes at the same output resolution it requires about 150 seconds.

Other processing, including the GPA, the TPS warp, and the extraction of the extremal surfaces, required about an hour altogether and was minor compared to the time required to generate the distance functions.

Figure 7 shows a larger image of one of our high-resolution models, with some artifacts, and the input geometry which makes these areas difficult for our technique to handle.

4 Discussion and future research

This application raises a number of research questions which we are interested in pursuing. With respect to primate evolution, we plan to compare the average ancestral shapes predicted by the statistical model and illustrated in this visualization with the shapes of know fossils, both visually and statistically. Integration of fossil evidence with trees such as ours, whose structure is inferred from DNA evidence from existing species, has to be based on morphological features. Visualizations such as these help paleontologists develop intuition about morphological change and encourage them to accept or reevaluate statistical models.

Generating landmarks automatically in a way which users would find sufficiently accurate and biologically meaningful is an important area for future research; as more data becomes available, the need for automation is becoming more pressing. For instance, it would be very helpful to be able to attract landmark points onto significant geometric features, especially ridges. More ambitiously, it would be useful to be able to develop a reliable surface correspondences using only a small number of landmarks, and hence transfer large sets of landmarks almost automatically. There has been some work on this problem in the graphics community [17] and extending these techniques to handle inputs which are not closed manifolds would be very interesting.

The problem of merging multiple similar surfaces, which might include holes and self-intersections, is challenging, and approaches other than the one we have taken here might also be successful. In general, it would be useful to study this and related problems such as parameterization in the context of "real world" scanned models and the difficulties they present. An alternative approach to the merging problem would have been to use the signed distance function rather than the squared distance; this would allow for an easier surface extraction process. We spent some time exploring this option, and even used it in a prototype, but we realized that making it robust would require investing more effort upfront on the problem of cleaning the input and producing consistent orientations before merging. The extremal surface technique we implemented instead comes down to first averaging the functions and then finding a consistent orientation and cleaning up the output function.

Acknowledgements

We thank James Jones, a former U.C. Davis undergraduate, for writing the thin-plate spline code, and Tom Brunet, currently a graduate student at the University of Wisconsin, for the code for Horn's algorithm. We thank Prof. Steve Frost for comments on the manuscript, Lissa Tallman for beta-testing the landmark editor and John Liechty for sound recording.

References

[1] D. C. Adams, F. J. Rohlf, and D. E. Slice. Geometric morphometrics: Ten years of progress following the 'revolution'. *Italian Journal of Zoology*, 71:5–16, 2004.

[2] B. Allen, B. Curless, and Z. Popovic. The space of human body shapes: reconstruction and parameterization from range scans. In *Proceedings of ACM SIGGRAPH*, pages 587–594, 2003.

[3] Nina Amenta and Yong Joo Kil. Defining point-set surfaces. In *Proceedings of ACM SIGGRAPH*, pages 264–270, 2004.

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

[5] F.L. Bookstein. Landmark methods for forms without landmarks: morphometrics of group differences in outline shape. *Medical Image Analysis*, 1(3):225–243, 1997.

[6] Daniel Cohen-Or, David Levin, and Amira Solomovici. Three-dimensional distance field metamorphosis. *ACM Transactions on Graphics*, 17:116–141, 1998.

[7] E. Delson, D. Reddy, S. Frost, F.J. Rohlf, M. Friess, K. McNulty, K. Baab, T. Capellini, and S. E. Hagell. 3d visualization of inferred intermediates on a phylogenetic tree–applications in paleoanthropology (abstract). *Amer. J. Phys. Anthropol. Suppl.*, 36:86–87, 2003.

[8] J. C. Gower. Generalized procrustes analysis. *Psychometrika*, pages 33–51, 1975.

[9] W.L. Hodges, R. Reyes, Jr. T. Garland, and T. Rowe. Visualizing horn evolution by morphing high-resolution x-ray ct images. In *Sketches*. SIGGRAPH 2003, 2003.

[10] B.K.P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4(4):629–642, 1987.

[11] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, volume 21, pages 163–169, July 1987.

[12] Sean Mauch. Closest point transform, 2004.

[13] G. Medioni, M. S. Lee, and C. K. Tang. *A Computational Framework for Segmentation and Grouping*. Elsevier Science B.V., 2000.

[14] Paul O'Higgins and Nicholas Jones. Morphologika: Tools for shape analysis.

[15] F. James Rohlf. tpsrelw: analysis of relative warps, 2004.

[16] F.J. Rohlf. Comparative methods for the analysis of continuous variables: geometric interpretations. *Evolution*, 55(11):2143–2160, 2001.

[17] J. Schreiner, A. Asirvatham, E. Praun, and H. Hoppe. Inter-surface mapping. In *Proceedings of ACM SIGGRAPH*, pages 870–877, 2004.

[18] Dennis E. Slice. Morpheus: Software for morphometric research.

[19] G. Turk and J. O'Brien. Shape transformation using variational implicit functions. In *Proceedings of ACM SIGGRAPH*, pages 335–342, 1999.

[20] J. J. Wiens, editor. *Phylogenetic analysis of morphological data*. Smithsonian Institution Press, 2000.

[21] Miriam Zelditch, Donald Swiderski, David H Sheets, and William Fink. *Geometric Morphometrics for Biologists*. Academic Press, 2004.