Mesh Modelling with Curve Analogies

Steve Zelinka Michael Garland

University of Illinois at Urbana-Champaign
Thomas M. Siebel Center for Computer Science
201 N. Goodwin, Urbana, IL, USA
{zelinka, garland}@uiuc.edu

Abstract

Modelling by analogy has become a powerful paradigm for editing images. Using a pair of before- and after-example images of a transformation, a system that models by analogy produces analogous transformations on arbitrary new images. This paper brings the expressive power of modelling by analogy to meshes.

To avoid the difficulty of specifying fully 3D example meshes, we use Curve Analogies to produce changes in meshes. We apply analogies to families of curves on an object's surface, and use the filtered curves to drive a transformation of the object. We demonstrate a range of filters, from simple local feature elimination/addition, to more general frequency enhancement filters.

1. Introduction

In recent years, many researchers have investigated methods to ease the task of mesh modelling. Advances in scanning technology have made the acquisition of models from real world objects ubiquitous and fast. The editing of such acquired models, such as for animation or to achieve a particular look, however, remains difficult. Much research centers around the development of new surface representations, such as implicit surfaces, level sets and subdivision surfaces. Each such representation offers its own powerful suite of tools for modelling, and skilled artists can produce truly stunning results. However, these modelling systems still typically require skill and experience to use effectively. Our goal is to create useful tools for less artistically-skilled modellers.

Modelling by analogy has proved to be a powerful, particularly easy to use tool for image manipulation [5]. Under this paradigm, the user simply supplies an example of a desired transformation (using, for example, an unfiltered image and a filtered image), and the system allows the user to

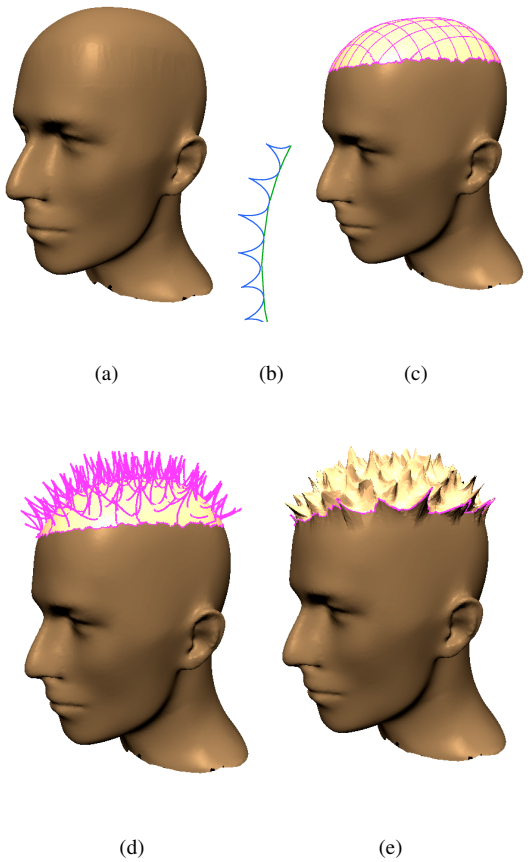


Figure 1. (a) Initial mesh. (b) Sample curves. Unfiltered in green, filtered in blue. (c) Network of unfiltered curves from mesh (magenta). Changes are limited to the brighter region. (d) Filtered curves, applying analogy in (b) to curves in (c). (e) Mesh transformed to match curves in (d).

apply a similar transformation to new images. Image Analogies essentially transcend traditional image processing techniques, allowing the use of all manner of image filters without requiring specific code for each and every particular filter. Given the recent proliferation of mesh filtering techniques, modelling by analogy would be an invaluable addition to any mesh modelling toolkit.

Hertzmann *et al.* [6] have recently demonstrated Curve Analogies, generalizing their image-based work to 2D curves and taking a plausible first step towards full 3D analogies. However, there is a hidden, increasing cost in moving from the image or curve domains into full 3D: that of demonstrating the desired transformation. In the 3D case, this requires finding and modifying an appropriate mesh- a task which, as noted above, often requires a skilled and dedicated artist. While *Mesh Analogies* generated in this way would certainly be a powerful tool, its accessibility, as compared to Image or Curve Analogies, is certainly limited.

We develop an alternate approach to Mesh Analogies, using Curve Analogies to directly alter 3D surfaces. Given a sample and transformed curve, we generate analogies with curves taken from the object's surface. As discussed in Section 4.1, simple planar slicing of an object with rotating or parallel planes is often sufficient to generate a set of curves that will yield good results, although more complex schemes are certainly possible. After transforming the surface's curves, the surface is transformed to match the changes in the curves. Our surface modification algorithm is described in Section 4.2.

As shown in Section 5, our approach makes it possible to retain the accessibility and ease of use of Curve Analogies, while losing little in the way of *expressiveness*, the space of filters which may be successfully reproduced. We demonstrate a range of filters, from simple detail addition, as in Figure 1, to low frequency enhancements (Figure 3).

2. Related Work

There is a vast body of research in the computer graphics literature regarding 3D object modelling techniques. We focus in this section on only the most relevant prior work: those systems whose methods or goals are most similar to ours. As discussed in the introduction, our work is heavily influenced by both Image Analogies [5] and Curve Analogies [6]; indeed, the practical extension of these groundbreaking ideas to 3D is a prime motivation for this work.

Using 2D curves as a basis for modelling 3D objects has a long history. Early approaches included various sweeps and surfaces of revolution [3]. Generative Modelling [12] generalized many of these concepts, allowing the designer to programmatically combine profiles in complex ways. While very well-suited to applications requiring precise surface description, such programmatic approaches are often difficult for non-technical users to exploit.

More recently, Singh and Fiume [11] used space-curves to simulate armatures commonly used for sculpting. Once a user has placed Wires about an object, changes to these curves result in local deformations. Whereas Wires are suited to performing fine-scale object deformations, such as for detailed facial animation, our technique is more applicable to larger-scale, homogenous changes to an object, and requires less user involvement.

Sketching curves to create 3D objects was demonstrated in the particularly user-friendly Teddy system [7]. Usable even by young children, Teddy first demonstrated the how intuitive 2D sketching can be for inducing changes in 3D objects. However, Teddy makes a number of assumptions about how sketches are to be interpreted in 3D, generally limiting it to creating cartoon-like models.

Numerous digital filtering systems have been developed in the past few years. Guskov *et al.* [4] perform interesting filtering and multiresolution editing operations on meshes using progressive meshes and subdivision techniques. Praun *et al.* [10] develop consistent parameterizations for meshes, allowing detail transfer among (sufficiently similar) meshes, as exploited using a database of human body models by Allen *et al.* [1]. These systems generally require new code for supporting new kinds of filters (i.e., expressiveness is limited), and suffer from relatively unintuitive controls (minor changes to a control may produce unexpectedly major changes in the mesh).

A somewhat different approach to mesh modelling may be found in texture (and more general signal) synthesis systems [14, 13]. These systems substitute image-based details for geometry, covering a surface with the appearance of fine details. While computationally similar to our approach, these methods cannot change the large-scale geometry of the object, and thus are inherently limited in their expressiveness. Further, since only appearance is changed, problems may arise in applications such as collision detection. Geometry may be generated with cellular textures [2, 9] in order to introduce relatively fine-scale details, but these require cell programs to be coded for each kind of change desired.

3. Curve Analogies

First, we briefly review Curve Analogies and their implications for our work.

Curve Analogies [6] generalize image-based neighbourhood analysis techniques to 2D curves. Given a set of sample curves, A and A', and a new target curve, B, we generate B' such that the analogy is fulfilled: A: A':: B: B'. For simplicity, it is assumed that each pair of curves has the same parameterization. Thus, to generate B', it suffices

to filter each vertex in B in turn. Each vertex $B(t_j)$ is filtered by finding the best parameter value t such that $A(t)$ matches $B(t_j)$ and $A'(t)$ matches $B'(t_j)$. Matching is done by comparing arclength-sampled neighbourhoods around vertices under optimally-aligning rotations and translations. The next vertex in A' is then appropriately rotated and translated, and copied over to B'.

Hertzmann *et al.* acknowledge an excessive number of parameters are required to be set correctly in order to achieve good results. However, we have not found it necessary to tweak parameters for any of our results; indeed, all were generated under the same Curve Analogy parameter settings, which were simply those suggested by Hertzmann *et al.* in their paper.

4. Slice Analogies

In the next sections, we describe in detail our new approach to mesh modelling with Curve Analogies. The next section discusses selection of curve families on a surface, and their implications to expressiveness. We then detail our surface modification procedure.

4.1. Curve Families

One approach to curve selection is to simply slice an object by a set of parallel planes. Beyond its extreme simplicity, this approach provides relatively even coverage of the surface, allowing changes to occur anywhere on an object's surface (assuming the density of slices is high enough). Computing slices is straightforward. We first assign to each mesh vertex a signed distance to a slicing plane, which allows us to identify edges crossing the slice as those with one positive and one negative vertex. We then compute actual intersections on each edge, and finally sort and connect the edges together into contours. Degenerate situations such as faces entirely within a slice are handled simply by including only their edges in the contours.

The simplicity of this approach is not without cost, however; we cannot introduce coherent features along the axis of slicing. For example, if slicing orthogonal to the z-axis, the user has little control to introduce features which span some interval of z.

This issue can be alleviated somewhat by using a rotating set of planes, rather than simply a translating set of planes. Rotating planes has its own set of disadvantages, however: areas near the axis of rotation receive higher coverage than areas far from the axis of rotation. Nonetheless, rotating planes can be highly effective for objects with rotational symmetries.

Better results can be obtained by using multiple sets of parallel planes, or multiple axes of rotation, simultaneously. We generally create a network of curves by slicing in more than one direction at a time. This allows for even coverage, and also supports feature creation in any direction. However, one issue is potential interactions among curves near their intersections. Unless some sort of special care is taken during the analogy construction, a point on the surface may be transformed into a spike on one curve, and a dip on an orthogonal curve. One way to avoid this would be to use multi-analogies, as described by Hertzmann *et al.* [6] in their multi-resolution curve analogies algorithm. Simultaneous analogies are drawn between a target and multiple sets of samples, and thus the choices made in the first curve (such as, to put a spike at a particular point) may be taken into account when generating the following curves. However, this would require relatively complicated establishment of correspondences between different surface curves. We have opted for a simpler approach: as we shall see in the next section, our surface transformation procedure is designed to handle these situations seamlessly.

4.2. Surface Transformation

We now detail our strategy for transforming the surface once its associated curves have been transformed.

Given an original surface S, a set of curves C embedded within it, and a set of transformed curves C', we construct the new surface S'.

Each curve c in C is assigned a geodesic radius of influence r_c. For a particular vertex v on S and a particular curve c in C let $\mathbf{u} = c(t_v)$ be the closest point on c to $\mathbf{v}$, and d_c be the geodesic distance between $\mathbf{v}$ and $\mathbf{u}$. We compute these values by extending shortest paths along the edges of S from each curve, recording each vertex encountered (note that this is only an approximation of geodesic distance). The set of curves C is then transformed to a new set of corresponding curves, C'. The closest point $\mathbf{u}$ is mapped to its corresponding position $\mathbf{u}'$ in the transformed curve. We transform the resulting displacement into a local frame on the surface, and evaluate it in a similar local frame at $\mathbf{v}$. Thus, the candidate position $\mathbf{v}_c$ of vertex $\mathbf{v}$ with respect to curve c is given by:

$$\mathbf{v}_c = \mathbf{v} + w_c \mathbf{M}_v^T \mathbf{M}_u (\mathbf{u}' - \mathbf{u}) \tag{1}$$

where $\mathbf{M}_p = [\mathbf{t}_p \mathbf{b}_p \mathbf{n}_p]^T$ is a matrix of the local frame vectors at point $\mathbf{p}$: $\mathbf{t}_p$ is the curve tangent at $\mathbf{p}$, $\mathbf{n}_p$ is the surface normal at $\mathbf{p}$, and $\mathbf{b}_p = \mathbf{n}_p \times \mathbf{t}_p$. Note that the curve tangents are transported across the surface from the curve to affected vertex positions using parallel transport [8] across the connecting edges. The weight is typically a Gaussian function of the distance of the vertex from the curve, e.g., $w_c = Gaussian(d_c/r_c)$. More complex filtering functions, including user-defined fall-offs (e.g., according to a Photoshop-like "curves" control), may of course be used.

There are potentially several curves influencing a particular vertex, so we take the weighted average of their candidate positions as the final position of the vertex:

$$\mathbf{v}' = \frac{\sum w_c \mathbf{v}_c}{\sum w_c} \tag{2}$$

Thus, supposing a vertex is influenced by only one curve, its transformation is similar to that curve's transformation at the closest point. Meanwhile, a vertex influenced by multiple curves is moved to the weighted centroid of the positions to which each influencing curve would move it. Thus, where curves covering the same portion of the surface conflict in their transformations, the resulting surface still smoothly transitions between the curves.

Note that we do not change the connectivity or sampling of the mesh. We make the simplifying assumption that the original mesh's sample density is high enough to support the required sample density of the transformed mesh (under this assumption, our geodesic distance approximation is generally good as well). However, introducing large folds throughout a mesh requires that the mesh initially be highly over-sampled. Such sampling may easily be achieved through subdivision or re-tiling. Note also that we may introduce self-intersections if the changes to the mesh are greater than the local feature size.

Our approach is quite similar to that of Wires [11]. The primary difference is our use of approximate geodesic distances for determining the influence of a curve. Wires instead uses simple 3D distances, and thus a wire near a fold may influence parts of the surface arbitrarily far away from it, geodesically. While this makes sense for their goal of digital armatures, we have found geodesic distances to be generally more intuitive for our system. Euclidean distances would be useful in the case of strand-like objects, such as plants or trees, where it would make more sense to scatter space-curves throughout the volume of the object.

5. Results

We have found our approach to mesh modelling using Curve Analogies to work quite well in practice. Runnings times are generally on the order of a few minutes, depending on the number of slices and the curves' sample densities.

Figure 1 uses Slice Analogies to introduce spikes of hair on a mannequin's head. The changes were localized to the top region of the head, and slicing was performed with two orthogonal sets of parallel planes. Note how the network of curves allow spikes to form in all directions.

In constrast, we produce simple smoothing results by using a wavy unfiltered curve and a smooth filtered curve in Figure 2. Only one set of parallel horizontal slices were necessary to produce this result. This sample transformation also demonstrates a useful user-interface feature of our system: taking a slice from the object as the unfiltered sample curve. This allows the user to simply sketch over the portions they want to change.

We use rotating planes, instead, in Figure 3 in order to enhance the low-frequency components of a rotationally-symmetric vase. The analogy works well, inflating the middle of the vase while extending and thinning its neck, and retaining the high-frequency details of the flower relief (there is some smoothing, of course, inherent in our surface transformation method). Note that the resulting surface is influenced by the transformed curves, but does not necessarily interpolate them.

We derive a slice based on the silhouette of the dinosaur model in Figure 4. Changes are then limited to be along the back of the dinosaur, and we add some curved spikes to the model. Note that this kind of change requires geometry; the curving of the spikes cannot be represented with displacement maps or bidirectional texture functions.

6. Conclusions and Future Work

In conclusion, we have presented a new system which brings the expressive power and flexibility of modelling by analogy to the domain of meshes. We base our system on Curve Analogies in order to avoid the difficulty of specifying a fully 3D example transformation, instead allowing a user to simply sketch 2D curves in order to enact transformations.

There a numerous avenues for future work with our system. We are developing a new formulation of Curve Analogies more directly suited to our application goals. In particular, it seems that curve comparisons under rotation invariance is not always desired. In our system, rotation invariance can result in an incorrect (from the user's view) orientation for an analogy. For example, an analogy intended to add bumps to a surface may instead add dips. Such orientation flipping can even occur in the middle of a curve. A better formulation could prevent such things by taking orientation hints from the local frame on the surface, rejecting inconsistent alignments. We would also like to experiment with additional curve families on surfaces. Silhouettes, as shown in Figure 4, seem quite promising. Isoparameter lines, for regularly parameterized surfaces, would likely form useful target curves as well. In addition, use of object skeletons may produce interesting results. For example, one could produce a series of slices orthogonal to the skeleton of an object, to allow relatively principled transformation along its extent. Applying Curve Analogies directly to an object's skeleton may also produce interesting effects.

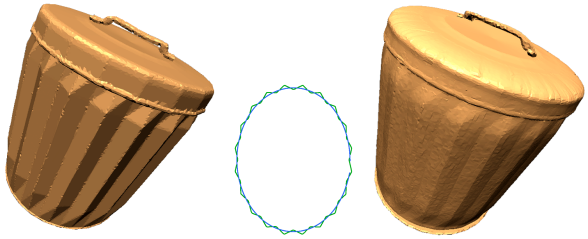
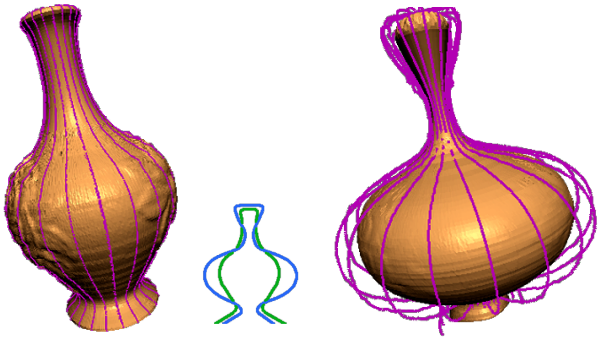


Figure 2. Smoothing example. Transforming a spiked curve into a smooth curve smooths out the sides of the trash can.



(a) (b) (c)

Figure 3. Low-frequency enhancement. Rotational slices of the object are transformed to have their low-frequencies emphasized. (a) Original, with unfiltered curves in magenta. (b) Sample curves. (c) Transformed surface, with filtered curves in magenta.

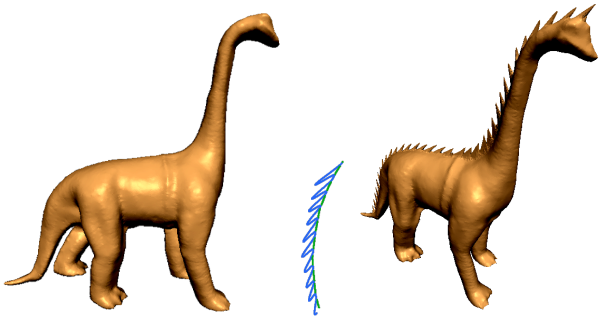


Figure 4. We transform part of the silhouette of the dinosaur with the given analogy to add spikes along its back.

Acknowledgements

We would like to thank the anonymous reviewers for all of their helpful comments. This research was supported under a grant from the NSF (CCR-0086084).

References

[1] B. Allen, B. Curless, and Z. Popović. The space of human body shapes. *ACM Transactions on Graphics*, 22(3):587–594, 2003.

[2] K. W. Fleischer, D. H. Laidlaw, B. L. Currin, and A. H. Barr. Cellular texture generation. In *Proceedings of SIGGRAPH 95*, pages 239–248, 1995.

[3] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice, 2nd ed.* Addison-Wesley, Reading MA, 1990.

[4] I. Guskov, W. Sweldens, and P. Schröeder. Multiresolution signal processing for meshes. In *Proceedings of SIGGRAPH 99*, pages 325–334, 1999.

[5] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *Proceedings of SIGGRAPH 2001*, pages 327–340, 2001.

[6] A. Hertzmann, N. Oliver, B. Curless, and S. M. Seitz. Curve analogies. In *Proceedings of the 13th Eurographics Workshop on Rendering*, pages 233–245, Pisa, Italy, June 2002.

[7] T. Igarashi, S. Matsuoka, and H. Tanaka. Teddy: A sketching interface for 3d freefrom design. In *Proceedings of SIGGRAPH 99*, pages 409–416, Aug. 1999.

[8] E. Kreyszig. *Differential Geometry*. Dover Publications, New York, 1991.

[9] J. Legakis, J. Dorsey, and S. J. Gortler. Feature-based cellular texturing for architectural models. In *Proceedings of SIGGRAPH 2001*, pages 309–316, 2001.

[10] E. Praun, W. Sweldens, and P. Schröeder. Consistent mesh parameterizations. In *Proceedings of SIGGRAPH 2001*, pages 179–184, Aug. 2001.

[11] K. Singh and E. Fiume. Wires: A geometric deformation technique. In *Proceedings of SIGGRAPH 98*, pages 405–414, July 1998.

[12] J. M. Snyder. *Generative Modeling for Computer Graphics and CAD*. Academic Press, 1992.

[13] X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo, and H.-Y. Shum. Synthesis of bidirectional texture functions on arbitrary surfaces. *ACM Transactions on Graphics*, 21(3):665–672, 2002.

[14] L. Ying, A. Hertzmann, H. Biermann, and D. Zorin. Texture and shape synthesis on surfaces. In *Proceedings of the Twelfth Eurographics Workshop on Rendering Techniques*, pages 301–312. Eurographics Assocation, 2001.