Proceedings of DETC'05
2005 ASME Design Engineering Technical Conferences
Long Beach, CA, USA, September 24-28, 2005

DETC05/DAC-number

CONVERTING MOLECULAR MESHES INTO SPLINE SURFACES: INTERPOLATION AND STITCHING

Joel Daniels II
School of Computing
University of Utah
Salt Lake City, Utah 84102
Email: jdaniels@cs.utah.edu

Elaine Cohen
David Johnson
School of Computing
University of Utah
Salt Lake City, Utah 84102
Email: cohen@cs.utah.edu
Email: dejohnso@cs.utah.edu

ABSTRACT

INTRODUCTION

Biologists are enthusiastic about the ability to handle physical representations of the molecular data with which they work. These 3-dimensional visualization tools make it easy to explore and understand the models. This work addresses the challenges associated with the fabrication of their molecular data. The goal is to construct a colorful molding of a protein strand snaking through a translucent plastic representing its water inaccessible surface. Construction of a physical model is not a trivial task as the data is described by both spline and triangular mesh structures.

Many challenges exist when working within a heterogeneous environment. Operations to deform, combine, and work with the assorted objects are best performed when all representations are similar. In order to utilize modeling software packages, it is necessary to convert one representation into to the other. Conversion of spline models into triangular meshes results in the loss of data and injects a level of coarseness into the model. Instead, methods to perform the reverse can preserve the original data while hypothesizing the surface between known points.

In this paper, we define a system to convert a triangular mesh into a spline model, producing a smoother model that interpolates the original data points. The continuity of the splines is more adept at capturing the natural flow of biological data. The transformation also produces a homogenous environment, condusive to performing the necessary operations required for manufacturing the desired pieces.

Our system creates a near G^1 continuous model by compositing 10 bi-cubic spline patches to encompass the original mesh. The interior points of each tensor product surface are C^2 continuous by the properties of the splines. Meanwhile, the boundaries and corners require special considerations to ensure a level of smoothness. This work introduces an algorithm to minimize ridges along boundaries at the intersection of an odd number of spline patches. When coupled with complete spline interpolation, the system generates a smooth spline model that exactly interpolates the original data points.

System Overview

Figure 1 illustrates the pipeline that converts an input triangular mesh into a spline model. The following section briefly highlights the steps in the pipeline and the problems addressed by the system. Further implementation details, motivations, and challenges are provided in later sections.

The input to the pipeline is a triangular mesh with a connectivity equivalent to the n^{th} subdivision level of an icosahedron. This data structure is the bi-product of sampling techniques used to obtain some molecular models, including the water inaccessible surfaces that motivate this work. The input mesh can be reduced to its original 20 faces by recursively grouping children

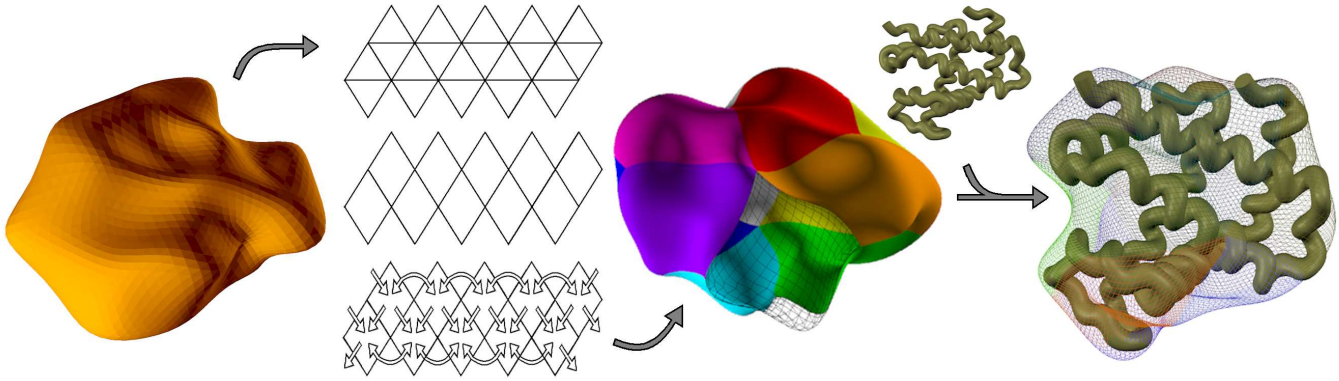

Figure 1. Illustration of the conversion pipeline of a molecular triangular mesh into a spline model. The input mesh has connectivity equivalent to the $n^t h$ subdivision level of an icosahedron. The data is segmented into 10 rectangular grids, tangents are fit across adjacent boundaries, and complete spline interpolation creates the spline surfaces. Boolean operations combine two spline models in the fabrication process.

with their parent triangles.

The system first groups the 20 hierarchical triangle trees into 10 pairs that will become the base for each bi-cubic patch. Rectangular grids of data points are extracted by walking the paired trees. Next, cross-boundary tangents are computed to fit the shared boundaries. After, the system computes cross-boundary tangents and twists for each of the corners, the intersection of multiple patches. Estimates of all cross-boundary tangents make considerations to avoid undulations by analyzing surfaces fit to the regions.

The final phase of the system involves complete spline interpolation, using the earlier collected rectangular grids and computed tangents and corner twists. The 10 bi-cubic tensor product spline patches then describe surfaces that completely encompass the original data. The resultant model is C^2 continuous within each spline surface and G^1 across the shared boundaries.

The configuration of the 10 patches produces two intersection scenarios. On the end model, there are corners where either 3 or 5 spline surfaces intersect at a common point. It is impossible to specify cross-boundary tangents at these corners to make a C^1 intersection without producing a singularity for one patch. Consequently, this algorithm is only capable of achieving a G^1 stitching at the corner.

The remainder of the paper is organized as follows. The previous work section summarizes similar efforts in surface subdivision and spline fitting. The implementation section further describes the details, methods, and motivations of the pipeline highlighted by this section. The case studies analyze the results of the conversion process and illustrates the constraints of the algorithms. Finally, the last section provides concluding remarks, reporting the results and success of the conversion process, as well as stressing the importance of the issues addressed by this work.

PREVIOUS WORK

Many research topics exert efforts in solving mesh smoothing to better define a model. The most popular techniques are subdivision surfaces. Catmull-Clark [1], Doo-Sabin [2], and Loop [3] schemes each develop refinement methods to recursively define new smoother meshes. Stencils weight existing information in order to computed the locations of new vertices, edges and faces. Subdivision schemes produce smooth surfaces, which , in their limit are equivalent to a spline surface of a given degree. These refined meshes soften features; however, they fail to interpolate the original data as each recursive step shrinks the model.

Algorithms designed to fit surfaces to point cloud data sets address similar smoothing and data interpolation considerations. Xie et al. [4] create C^1 models by growing a defined surface along a frontier. The prioritized expansion fits quadrics to the local data points in an area at the edge of the defined surface. The recent efforts of Cheng et al. [5] present an iterative method to fit a Loop subdivision surface to an unorganized point cloud. The defined surface converges toward the original data by optimizing a defined square distance minimization method. Similarly Hoppe et al. [6] fit Loop subdivision surfaces to scattered data, focusing on constructing smooth piecewise surfaces. This work also present modifications to Loop's subdivision rules in order to model sharp features. Quadrics and triangular meshes closely approximate the input data without shrinking the model as with subdivision surfaces. However, they do not produce the homogenous environment needed to model our molecular data.

Instead, multiple non-uniform b-spline surfaces (NURBS) may be composited to define complex models. The patches are stitched together in order to match the differential properties across shared boundaries. Geometric modeling tests, [7], describe interpolation techniques capable of considering tangential

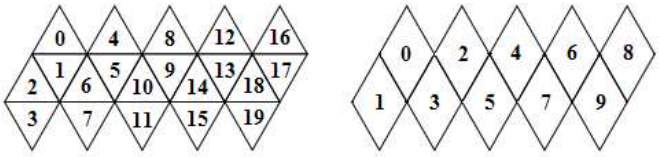

Figure 2. An unwound icosahedron and the triangle pairing scheme.

information. NURBS may be computed such that they exactly interpolate a set of data and produce seamless models.

Some efforts recognize these advantages and focus on defining splines to model input data. Grimm et al. [8] produce manifold surfaces of medical data by leveraging a user produced generator polyhedron. The model aids in the calculation of spline patches that are fit to the original data. Stitching between adjacent patches is avoided by overlapping boundaries. Loop [9] fits a G^1 continuous representation of an irregular mesh by utilizing quad-nets to generate smooth spline surfaces. Krishnamurthy and Levoy [10] describe an interactive algorithm, allowing a user to paint the boundaries of spline patches on the input mesh. They fit the spline patches to the user-partitioned data. These approaches develop methods to smoothly interpolate data with spline representations; however, our system removes user input by leveraging additional assumptions based on the inherent nature of our input data.

In a related master's thesis, Livingston [11] explores the continuity stitching the intersection of three spline patches. His work allows the end stitching to modify the location of corner point, to produce smoother results. In order to preserve the input data, we require that the end model interpolates the orginal mesh's vertices. The additional constraint inhibits the ability to modify the location of the corner control point.

IMPLEMENTATION

The following section further describes the system pipeline illustrated in Figure 1. Molecular models are smooth by nature without ridge lines or creases on their surface. The main challenge when converting the triangular mesh into a spline model arises when smoothing the boundaries of adjacent patches, more specifically when dealing with corners. The following subsections describe an algorithm to handle the stitching process between adjacent patches, focusing on minimizing the trouble areas surrounding corners shared by an odd number of spline patches.

The system requires the input triangular mesh to have a known connectivity, in order to remove the need for user input and avoid unnecessary complications. The molecular data is sampled and represented as a mesh with connectivity similar to the n^{th} subdivision level of an icosahedron. These triangles are recursively grouped under parent triangles to find the 20 hierarchical trees who's roots are the original faces of the icosahedron. The extracted trees are paired together as shown in Figure 2 such that no T-junctions occur between boundary edges and every triangle is paired once and only once. The coupling of the trees provides the data segmentation scheme that will be used to describe each spline patch. After pairing the triangles, the lowest level of children triangles are marched to extract the rows and columns of the rectangular data grids.

Complete spline interpolation is a frequently used technique to define surfaces through a grid of data by using non-uniform open cubic B-splines. Interpolation techniques require that each data point is assigned a parameter value $\{(u_i, p_i)\}_{i=0}^{s}$. The parameter values specify when the surface will interpolate the point. Spline surfaces ensure continuity conditions through their degree. A C^2 continuous surface requires the interpolant curve's domain be the interval $[u_0, u_s]$, all interior knots have a multiplicity of one, u_i, $i = 1, ..., s-1$, and the degree is 3. This forms a unique knot vector $t = \{t_j\}$ where $t_j = u_0$, $j = 0, 1, 2, 3$; $t_j = u_{j-3}$, $j = 4, ..., s+2$; and $t_j = u_s$; $j = s+3, ..., s+6$. The cubic B-spline defined by this knot set has $s+3$ degrees of freedom, and only $s+1$ data points exist to be interpolated.

Complete spline interpolation defines the final two constraints by specifying the tangents at the endpoints, p'_0 and p'_s. This is an attractive method to composite surfaces describing a model because it exactly interpolates original data values and is able to ensure that adjacent spline patches are continuous across their shared edge.

Cross-Boundary Tangents

Defining cross-boundary tangents matches the differential properties of the two interpolant surfaces. In order to perform the interpolation with stitching, first each point is associated with a parameter value. In the molecular model, each data point is sampled at a regular interval, making the parameter assignment simply its row and column within the grid. In a more general case, these parameter values may be computed based on the distances separating each data point. Next, complete spline interpolation requires the computation of cross-boundary tangents.

The stitching tangents require an educated guess at a reasonable values to avoid possible undulations. Point interpolation methods may create a surface that needs to fluctuate, sometimes wildly, in order to pass through each data point. Because the molecular model is a smooth surface and sampled regularly, then reasonable tangents exist along these edges. These tangents will closely represent the molecular surface without introducing waves along the patches' boundaries.

Quadratic B-splines, fit through 3 data points at every shared vertex, $\{p_0, p_1, p_2\}$, compute sensible tangents. The points are selected to align rows and columns across patches, with the points, p_0 and p_2, belonging to the separate adjacent patches, and the point, p_1, located on their shared edge. Parameter val-

ues are assigned to the 3 points in the same manner that they had been assigned to the rectangular grids previously described. The curve's tangent,

$$c\prime(t) = 2a_2t + a_1t, \tag{1}$$

where

$$a_2 = \frac{\frac{p_2-p_0}{u_2-u_0} - \frac{p_1-p_0}{u_1-u_0}}{u_2-u_1}, \tag{2}$$

$$a_1 = \frac{p_1-p_0}{u_1-u_0} - a_2(u_1+u_0), \tag{3}$$

is evaluated at $t = u_1$.

The tangents computed across shared boundaries must be reasonable in order to reduce undulations in the surfaces. The closeness and regularity of the sampling, as well as the smoothness of the molecular surface causes tangents calculated by fitting a quadratic spline over an edge to closely resemble tangents that would be produced by fitting cubic surfaces over the same region. Quadratic fit splines require as little as 3 data points, giving each involved patch equal influence over the direction of the tangent. The computed educated guess is a reasonable estimate that will not produce any undulations within the surface that do not exist, based on the knowledge that the original model is a smooth molecular surface.

Corner Tangents and Twists

The intersection of an odd number of patches at a common point requires further attention and is the heart of this research. The following method describes a technique to compute a combination of tangents and twists for each spline patch surrounding the corner. The twist is defined as $\frac{\partial^2\sigma}{\partial u \partial v}$ where u and v are defined along the boundaries for the patch.

It is impossible to specify cross-boundary tangents, for an odd number or intersecting patches, without introducing a singularity on one of the patches. Instead, an optimal configuration must be computed such that the normals belonging to two patches at a shared point closely align. By striving for G^1 continuity along boundaries it is not required that cross-boundary tangents be co-linear at the corner. The remainder of this subsection describes the algorithm implemented by our system to deal with smoothing corners. While only the 3 patch intersection is discussed, it is scalable to any number of odd patches, and, in fact, must be scaled to the 5 patch intersection to solve the conversion process.

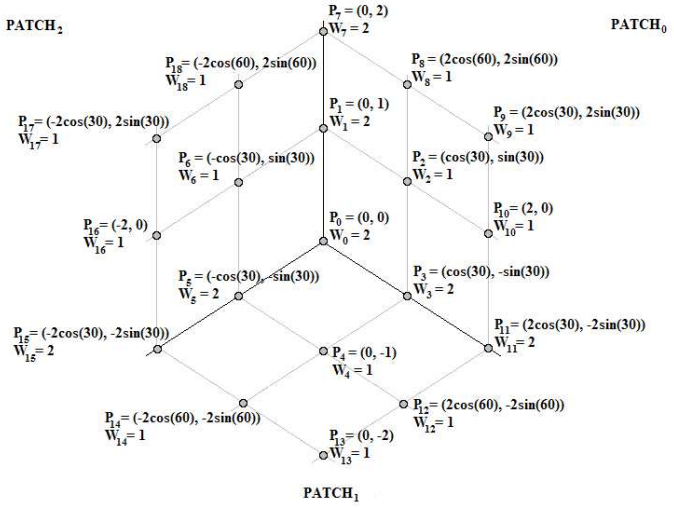

Figure 3. The point neighborhood of the 3 patch intersection with the parameter values and weights associated with each point.

The system analyzes the properties of a cubic surface fit to the corner region in order to estimate tangent and twist values that will smoothly stitch the intersections of the different patches. First the neighborhood fo the corner is extracted from the patch. As pictured in Figure 3 the static parameter values are assigned to this double ring of neighbor points. The parameter values treat the points as equally separated, and while this is not the case, an analysis in later sections further discusses the effects of these parameter values. Weighted least square algorithm then fits a cubic surface to the intersection area. The surface's equation is a product of the (u,v) parameter space,

$$\begin{aligned}\sigma = a_{00} + a_{10}u + a_{01}v + a_{20}u^2 + a_{11}uv + a_{02}v^2 \\ + a_{30}u^3 + a_{21}u^2v + a_{12}uv^2 + a_{03}v^3.\end{aligned} \tag{4}$$

The partial derivatives, $\frac{\partial\sigma}{\partial u}$ and $\frac{\partial\sigma}{\partial v}$ evaluated at $\sigma(0,0)$, span the tangent plan located at the origin. The tangent values are computed down each of the boundary edges by multiplying the parameter values for the boundary points with the computed partials.

$$tangent(u_i, v_i) = \frac{\partial\sigma}{\partial u}u_i + \frac{\partial\sigma}{\partial v}v_i \tag{5}$$

The system evaluates the control points along each boundary equal to $\frac{1}{3}^{rd}$ the length of the corresponding tangent value. The variables a,b,c,d, and e are computed corresponding to the control points locations on the tangent plane spanned by the previ-

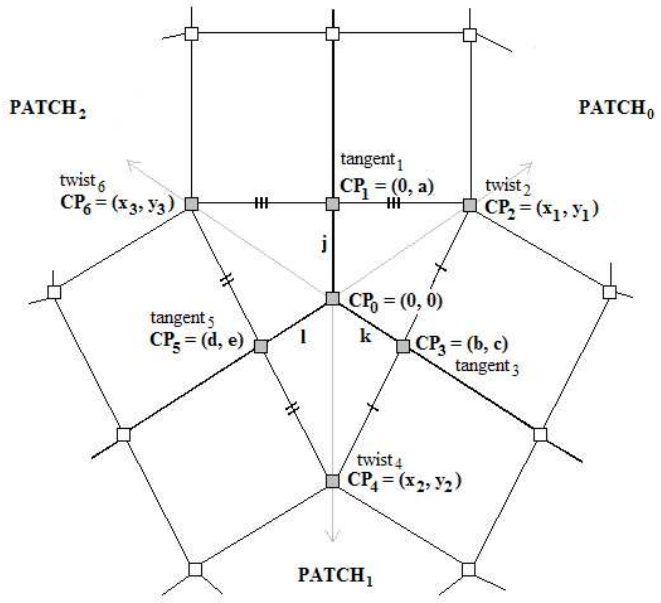

Figure 4. The computed control points surrounding the 3 patch intersection, depicting the variables and constraints enforced on the corner.

ously computed partials. Figure 4 illustrates the control points with their corresponding (u,v) position coordinates.

After the control points for the tangent values have been computed, the control points responsible for the corner's twists, $\frac{\partial^2 \sigma}{\partial u \partial v}$, are computed for each spline patch. Figure 4 depicts the constraints made on the twists' locations. The midpoints between each pair of twist control points is the tangent control point for the boundary between them. Additionally each twist point lies on the line defined by the tangent vector opposite it. These constraints continue the co-linear cross-boundary tangents and all points are co-planar around the corner thus restriciting the normals at this point to be equivalent. The constraints restrict the variables illustrated in Figure 4 to the following equations dependant upon j's value:

$$k = \frac{jad}{be - cd}, \tag{6}$$

$$l = \frac{jab}{cd - be}, \tag{7}$$

$$(x_1, y_1) = (\frac{2jabd}{be - cd}, \frac{2jabe}{be - cd}), \tag{8}$$

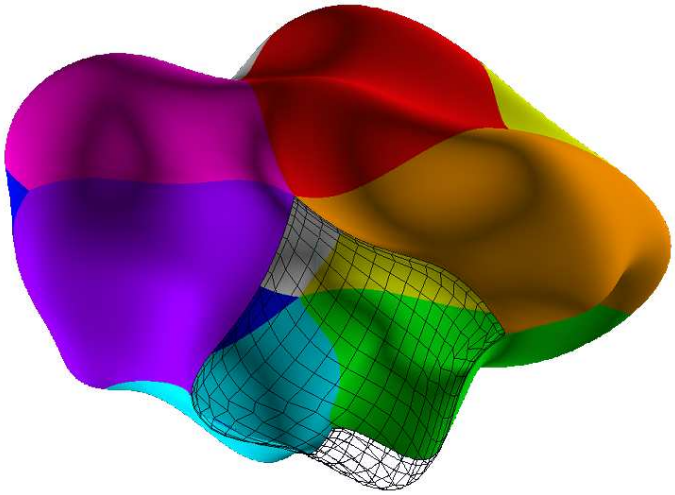

Figure 5. The converted molecular mesh as a smooth spline model.

$$(x_2, y_2) = (0, -2ja), \tag{9}$$

$$(x_3, y_3) = (\frac{2jabd}{cd - be}, \frac{2jacd}{cd - be}), \tag{10}$$

where, the j variable is set such that $j,k,l <= 1.0$, and $max(j,k,l) = 1.0$.

After computing the desired locations of the co-planar twist and tangent control points surrounding the corner, the system reverts these values back into the tangent and twist values for each patch. Referring to Figure 4, tangent and twist values for each patch are computed as follows.

$$tangent_i = 3(cp_i - cp_0), \tag{11}$$

$$twist_i = 9(cp_{i-1} + cp_{i+1} - cp_i - cp_0). \tag{12}$$

The algorithm produces the required components to create the spline patches. Complete spline interpolation converts the rectangular grids of data points, tangents, and twists into these spline surfaces. By compositing each patch the original molecule is reconstructed as a smoother model, as illustrated in Figure 5. The following section quantifies the smoothness results and analyzes the end surface.

CASE STUDY

The following section separately analyzes the continuity in two different cases. The first model is a flatter molecule, while

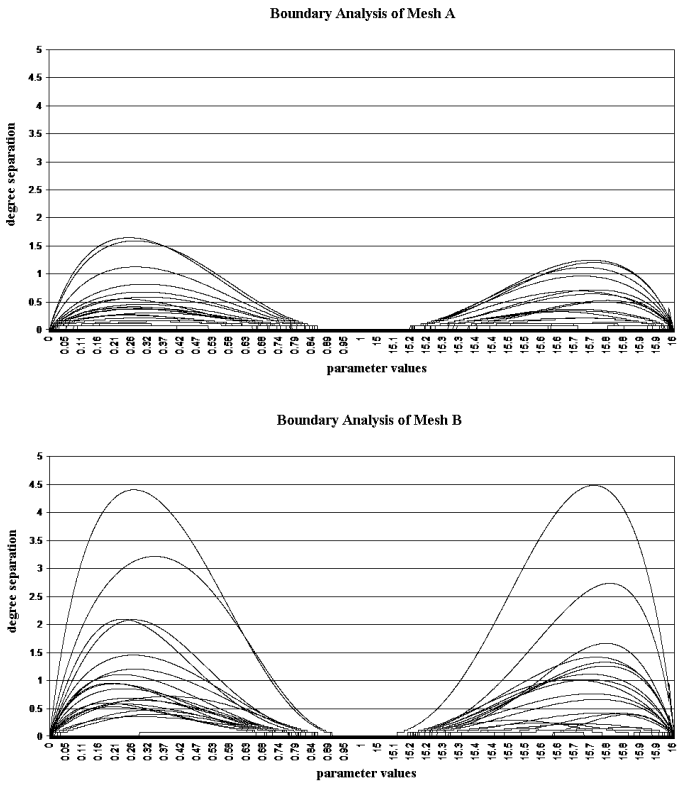

Figure 6. The angles between normals at shared points along each boundary on (a) the better model and (b) the high curvature model.

the second is dominated by areas of high curvature, particularly at the corner intersections. The first and last knot interval are the only areas of a boundary not gauranteed to be G^1 continuous by the configuration of the cross-boundary tangents.

Figure 6 plots the angle differences between normals of shared points along each boundary on the models. The graphs only depict the differences within the first and last knot intervals, focusing on the regions of potential trouble. The graphs indicate that the first model experiences better results than the second model.

The flatter molecule of graph (a) endures a worst case angle of 1.74°. This same boundary on the original mesh has a 7.6° angle between the triangle normals. Approximately 92% of the boundary points for this model are G^1 continuous, and 99% are within 1° of G^1 continuity, by the graphs. The conversion process produces a smoother model, that in its worst case still is better than the original representation. The ridges resulting between the composited patches add to a small fraction of the overall surface, and of those creases, a small portion may have an actual impact.

The second model with higher curvature experiences similar results, however, with a higher worst case. The 4.71° angle difference corresponds to a 12.4° angle difference between the triangles on the original mesh. The converted spline model is 91% G^1 continuous and 98% within 1° of G^1 continuity. While the worst case is larger than the first model, the conversion records similar percentages of smoothness. The second model converts well in most cases, and only 4 boundaries, as indicated by Figure 6, exhibit poor performance.

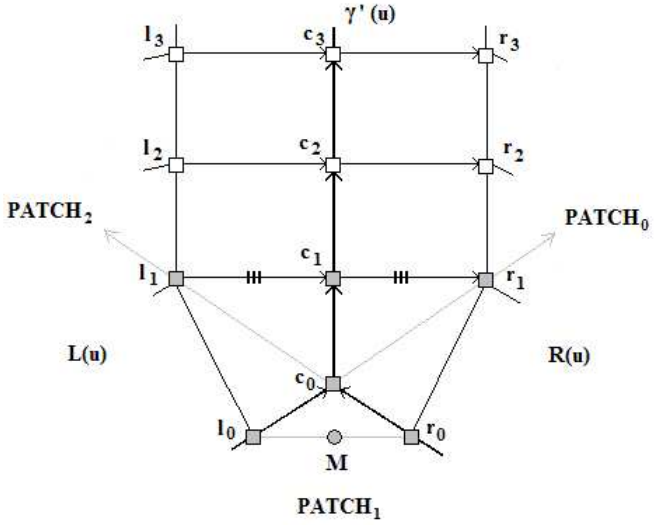

Figure 7. The naming scheme of $L(u)$, $R(u)$, and $\gamma'(u)$, used during the examination of the poor performance surrounding areas of high curvature.

The size of the ridges produced by the corners corresponds with the curvature of the surfaces at their intersection. To produce a G^1 continuous boundary the twists and tangents must satisfy the equation,

$$\gamma'(u) \times (L(u) - R(u)) = 0. \tag{13}$$

Figure 7 illustrates these variables and the naming scheme of their associated points. Reducing the equation by defining $\gamma'(u)$ and $L(u) - R(u)$ explains poor performance recorded around some boundaries.

$$\gamma'(u) = (c_1 - c_0)\beta_0(u) + (c_2 - c_1)\beta_1(u) + (c_3 - c_2)\beta_2(u), \tag{14}$$

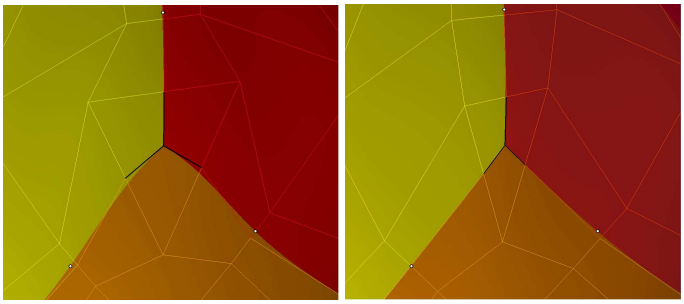

Figure 8. The swirl effect of a static parameterization versus the straightened results of the dynamic parameterization.

$$\begin{aligned} L(u) - R(u) &= [(l_3 - c_3) - (c_3 - r_3)]\Theta_3(u) \\ &\quad +[(l_2 - c_2) - (c_2 - r_2)]\Theta_2(u) \\ &\quad +[(l_1 - c_1) - (c_1 - r_1)]\Theta_1(u) \\ &\quad +[(l_0 - c_0) - (c_0 - r_0)]\Theta_0(u) \\ &= (l_3 + r_3 - 2c_3)\Theta_3(u) + (l_2 + r_2 - 2c_2)\Theta_2(u) \\ &\quad +(l_1 + c_1 - 2c_1)\Theta_1(u) + (l_0 + c_0 - 2c_0)\Theta_0(u) \\ &= 2(\tfrac{l_0+r_0}{2} - c_0)\Theta_0(u) \\ &= 2(M - c_0)\Theta_0(u), \end{aligned} \tag{15}$$

By substituting back into the original equation,

$$\begin{aligned} \Theta_0(u)[\quad & (M - c_0) \times (c_1 - c_0)\beta_0(u) \\ & + (M - c_0) \times (c_2 - c_1)\beta_1(u) \\ & + (M - c_0) \times (c_3 - c_2)\beta_2(u)] = 0 \end{aligned} \tag{16}$$

The algorithm gaurantees that $(M - c_0)$ aligns with $(c_1 - c_0)$ thus cancelling the error calculated by the first term. $\beta_0(u)$ is a major contributor over the region of potential trouble, thus such a configuration will yield good results. However, in areas where the region is not planar, $(c_2 - c_1)$ and $(c_3 - c_2)$ are not similar to the vector $(c_1 - c_0)$. Because $\beta_1(u)$ also has a strong influence over this region, the second error term will contribute largely. When the vectors $(c_3 - c_2)$, $(c_2 - c_1)$, and $(c_1 - c_0)$ do not align, it is impossible to create a perfectly G^1 surface across the boundary without modifying the original data. The best one can hope for is that the chosen tangents compute an M that minimizes the error.

Multiple methods to compute different tangent planes were implemented, including using dynamic parameter values for the cubic surface fit. This method computes parameter values based on the points location to a projected plane. This method is intended to remove the small swirls that appear around corners with the static parameterization. Dynamic parameter values evaluate tangents that emanate directly from the corner to the first data value. The new parameter values remove the swirl; however, it fails to realize better worst case results. As shown earlier, the tangent planes fit by the static parameterization consistantly compute better M values for each boundary.

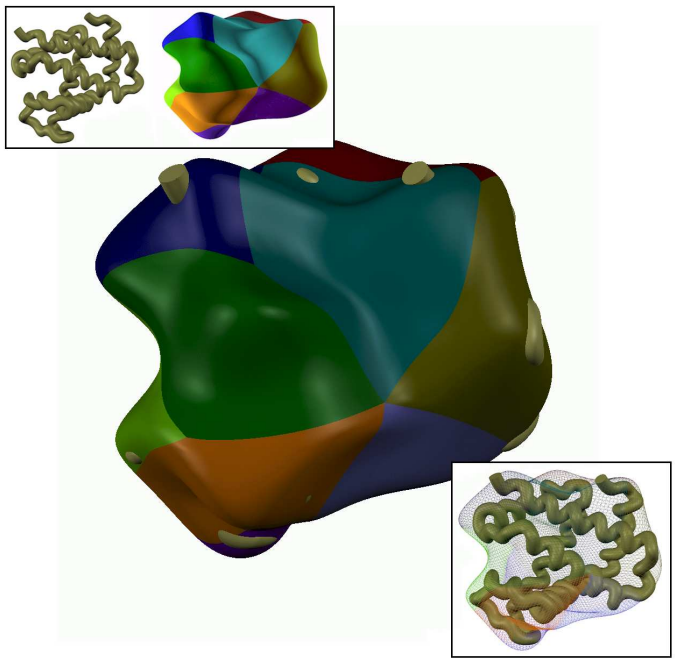

Figure 9. Within the homogenous environment, the protein spline model is combined with the converted molecular spline model.

CONCLUSION

The system successfully converts a molecular mesh into a spline model. The new surface preserves the original data by interpolating the vertices of the input mesh. 10 bi-cubic patches seamlessly encompass the original molecular model with a smoothness that more closely simulates a biological surface. 99% of the boundary space is within 1° of G^1 continuity, as regions of potential trouble are confined to the corners of each patch. In the worst case, the angles between normals of two patches at a common point are smaller than their corresponding boundaries on the original mesh. The resultant spline model is an accurate representation of the original data with an added level of smoothness.

Leveraging the homogenous environment, modeling software toolkits can combine the protein strand's spline model with its water inaccessible surface via boolean operations. A rapid prototype machine constructs the combined model, which in turn is used to create a flexible split away mold. Meanwhile the protein model is manufactured with opaque plastics colored based on the different properties of the strand. The fabricated protein is fit into the split away mold and a translucent plastic fills the remaining volume. Figure 9 shows the combined models and gives an idea of the manufactured piece.

The restriction on the connectivity of the input mesh constrains the generality of the conversion process. The pipeline makes assumptions of the input mesh, not addressing the prob-

lem of data segmentation. Instead the algorithms focus on smoothly stitching boundaries between adjacent patches, particularly at the intersections of an odd number of spline surfaces. While the system is unable to support arbitrary meshes, it does produce the algorithms to create the optimal continuity across boundaries.

REFERENCES

[1] Catmull, E., and Clark, J., 1978. "Recursively generated b-spline surfaces on arbitrary topological meshes". *Computer-Aided Design*, **10**(6), November, pp. 350–355.

[2] Doo, D., and Sabin, M., 1978. "Behavior of recursive division surfaces near extraordinary points". *Computer Aided-Design*.

[3] Loop, C., 1987. "Smooth subdivision surfaces based on triangles". *Master's Thesis, University of Utah, Department of Mathematics*.

[4] Xie, H., Wang, J., Hua, J., Qin, H., and Kaufman, A., 2003. "Piecewise c^1 continuous surface reconstruction of noisy point clouds via local implicit quadric regression". In IEEE Visualization '03 Conference Proceedings, pp. 91–98.

[5] Cheng, K., Wang, W., Qin, H., Wong, K., Yang, H., and Liu, Y., 2004. "Fitting subdivision surfaces to unorganized point data using sdm". In 12th Pacific Conference on Computer Graphics and Applications '04 Proceedings, pp. 16–24.

[6] Hoppe, H., DeRose, T., Duchamp, T., and Halstead, M., 1994. "Piecewise smooth surface reconstruction". *Proceeding of the 21st Annual Conference on Computer Graphics*.

[7] Cohen, E., Riesenfeld, R., and Elber, G., 2001. *Geometric Modeling with Splines: An Introduction*. AK Peters, Ltd.

[8] Grimm, C., Crisco, J., and Laidlaw, D., 2002. "Fitting manifold surface to 3d point clouds". *Journal of BioMechanical Engineering*, February.

[9] Loop, C., 1994. "Smooth spline surfaces over irregular meshes". In SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques, pp. 303–310.

[10] Krishnamurthy, and Levoy, M., 1996. "Fitting smooth surfaces to dense polygon meshes". In ACM Transaction on Computer Graphics (SIGGRAPH 1996 Proceedings).

[11] Livingston, J. B., 1990. "Intersurface continuity of solid models". *Master's Thesis, University of Utah, Department of Computer Science*.