Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization (2003)
G.-P. Bonneau, S. Hahmann, C. D. Hansen (Editors)

Contouring Curved Quadratic Elements

D. F. Wiley[1], H. R. Childs[2], B. F. Gregorski[1], B. Hamann[1] and K. I. Joy[1]

[1] Center for Image Processing and Integrated Computing (CIPIC), Department of Computer Science,
University of California,Davis, CA 95616-8562, U.S.A.;
e-mail: {wiley, gregorsk, hamann, joy}@cs.ucdavis.edu
[2] B Division, Lawrence Livermore National Laboratory, Mail Stop L-098,
7000 East Avenue, Livermore, CA 94550, U.S.A.;
e-mail: childs3@llnl.gov

Abstract
We show how to extract a contour line (or isosurface) from quadratic elements—specifically from quadratic triangles and tetrahedra. We also devise how to transform the resulting contour line (or surface) into a quartic curve (or surface) based on a curved-triangle (curved-tetrahedron) mapping. A contour in a bivariate quadratic function defined over a triangle in parameter space is a conic section and can be represented by a rational-quadratic function, while in physical space it is a rational quartic. An isosurface in the trivariate case is represented as a rational-quadratic patch in parameter space and a rational-quartic patch in physical space. The resulting contour surfaces can be rendered efficiently in hardware.

Categories and Subject Descriptors (according to ACM CCS): I.4.10 [Image Representation]: Volumetric I.3.5 [Computational Geometry and Object Modeling]: Curve, surface, solid, and object representations

1. Introduction

Higher-order elements have gained in importance since they can be used to represent complex data. Figure 1 shows the advantage of using a higher-order element to represent data in the univariate case. Higher-order elements can typically represent data better when compared to lower-order elements. This improvement in quality is true for two, three, and higher dimensions. In the 2D case, a *linear triangular* element represents a linear functional defined over the domain of the triangle. Most visualization and approximation techniques can use this type of element. A higher-order triangular element is the *quadratic triangle*, which has a quadratic functional defined over the same domain as the linear triangle. *Linear tetrahedra* can be extended to *quadratic tetrahedra* in a similar fashion in the 3D case.

Higher-order hexahedral elements are popular in finite element applications[3], and Wiley et al.[13] showed their potential for substantial reductions in the number of required elements when replacing linear elements with quadratic elements. Here, we not only consider *linear-edge higher-order* elements, we also consider elements that are also of higher order in domain space. We call these elements *curved higher-order* elements. We devise a method to find contour lines (and isosurfaces) in *curved quadratic* triangles (and tetrahedra). We define a curved quadratic triangle (and tetrahedron) as an element that has both a quadratically defined domain and quadratic functional defined over that domain, Figure 2 shows an example.

Higher-order elements are typically tessellated by several smaller linear elements for rendering purposes. Conventional visualization methods, such as contouring, ray casting, and cutting-planes, can be applied directly to these linear elements. Visualization of higher-order elements is not nearly as highly developed as visualization for linear elements. Methods for efficiently visualizing higher-order elements are needed.

We contour trivariate quadratic elements by extracting triangular rational-quadratic patches and show how to contour curved quadratic elements in 2D and 3D domain spaces. We first map a quadratic function $\mathbf{F}(\mathbf{u}) : U \longrightarrow R$ defined over a curved quadratic element into parameter space U and find the representation $\mathbf{Q}(\mathbf{u}) : U \longrightarrow U$ for a contour value c such that $\mathbf{F}(\mathbf{Q}(\mathbf{u})) = c$, and then transform $\mathbf{Q}(\mathbf{u})$ into physical space R, yielding the mapping $\mathbf{C}(\mathbf{u}) : U \longrightarrow R$—the repre-

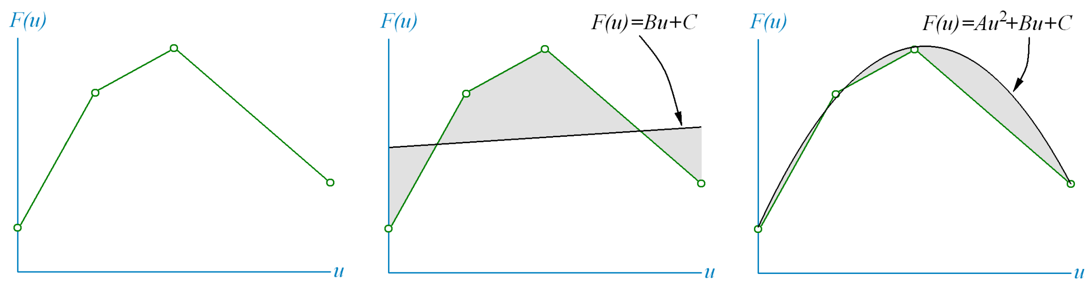

Figure 1: *Advantage of using a higher-order element representation. Left image shows original piecewise linear data. Middle image shows linear approximation using one linear element. Right image shows quadratic approximation using one quadratic element. Gray area represents approximation error.*

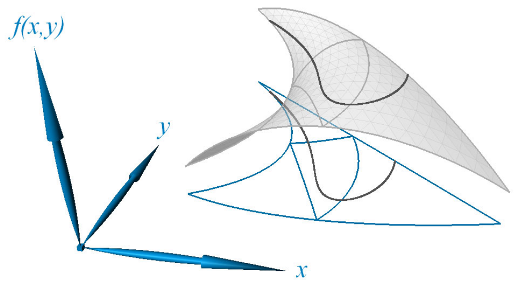

Figure 2: *Contour of a curved-quadratic triangle in physical space R^2. The dark curves show the contour in the xy-plane (domain space) and on the "graph" surface in 3D space.*

sentation of c in physical space. We show, for both the 2D and 3D case, how to transform rational-quadratic functions in U to rational-quartic functions in R. The resulting contour surfaces can be rendered efficiently in hardware. The ELSA Gladiac 920, nVidia GeForce 3 and GeForce 4, and ATI Radeon 8500 and Radeon 9700[12] video hardware all support varying levels of higher-order patch rendering suitable for quartic patches.

We first discuss how to find contours in 2D and 3D linear-edge quadratic elements (in our parameter space). We then continue with a description of transforming parameter-space contours into the physical space of the curved quadratic elements.

2. Previous work

Few higher-order element visualization techniques exist. Higher-order hexahedra visualization is described in [9]. Visualization of higher-order element isosurfaces in the form of *A-patches* is described in [1]. Elements with a higher-order domain and a linearly defined functional defined over that domain are volumetrically visualized by the method in [11]. Creation of hierarchical quadratic-tetrahedral approximations is discussed in [13].

Extracting isosurfaces from linear-edge quadratic triangles has been studied in [2, 10, 14]. The Worsey-and-Farin method[14] uses a Bernstein-Bézier basis, which tends to work better than the monomial basis used in the Marlow-and-Powell method[10]. The Worsey-and-Farin method[14] and the method discussed by Bloomquist in his thesis[2] provide a foundation for finding contours in quadratic elements in their parameter spaces (linear-edge quadratic simplices). Bloomquist used the Worsey-and-Farin method for the 2D case and extended it to the 3D case to find contour surface intersections with the faces of a tetrahedron.

2.1. The 2D case

We implemented the Worsey-and-Farin method[14] to find rational-quadratic curves that represent the contour passing through a linear-edge quadratic triangle. The domain of the standard triangle—with vertices (0,0), (1,0), and (0,1)—$U \subseteq R$ defines our *parameter space* and R *physical space*. The contour in a quadratic triangle can be quite complex, and it is often desirable to represent it by several segments. We define a univariate *rational-quadratic curve* $\mathbf{Q}(u) : U^1 \longrightarrow U^2$ that represents a segment of the contour, in Bernstein-Bézier form, with three control points $\mathbf{p}_i \in U^2$ and three weights $w_i, 0 \le i \le 2$, $w_i \ge 0$, defined as

$$\mathbf{Q}(u) = \frac{\sum_{0\le i\le 2} w_i \mathbf{p}_i B_i^2(u)}{\sum_{0\le i\le 2} w_i B_i^2(u)}, \tag{1}$$

where the univariate n^{th}-degree Bernstein polynomial $B_i^n(u)$ is

$$B_i^n(u) = \frac{n!}{(n-i)!i!}(1-u)^{n-i}u^i. \tag{2}$$

3. The 3D case

Our method is an extension of Bloomquist's trivariate contouring method using a method similar to [8]. Bloomquist's method is extended by forming triangular rational-quadratic patches that represent the contour surface in a linear-edge quadratic tetrahedron. We compute our representation by applying the contour line method to each of the tetrahedron's faces to find the contour intersections. Then, we form

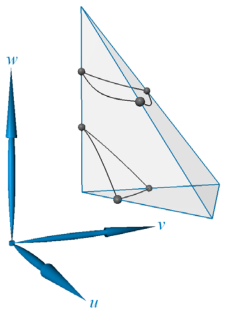

Figure 3: *Two contour surfaces inside a quadratic tetrahedron. Dark dots are the contour intersections with the edges. Dark curves are the contour intersections with the faces. There are two groups of three curves that bound two independent surfaces of the contour.*

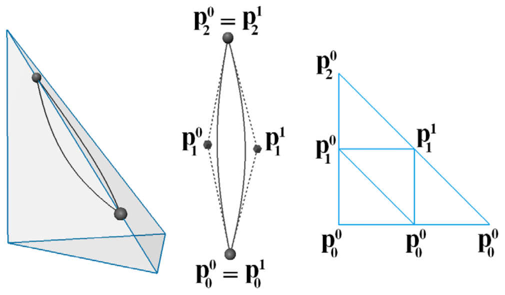

Figure 4: *Constructing a triangular patch from two curves. We collapse one edge of the patch by using the point $\mathbf{p}_0^0$ three times along an edge. Left image shows contour intersecting the faces of tetrahedron. Middle image shows labelled points of two-curve boundary polygon. Right image shows patch indexing.*

rational-quadratic patches that approximate the contour surface from the contour lines on the faces. We define a triangular *rational-quadratic patch* $\mathbf{Q}(u,v) : U^2 \longrightarrow U^3$ that represents a region of the isosurface, in Bernstein-Bézier form, with six control points $\mathbf{p}_{ij} \in U^3$ and six weights $w_{ij}, i, j \geq 0$, $i + j \leq 2$, $w_{ij} \geq 0$, defined as

$$\mathbf{Q}(u,v) = \frac{\sum_{\substack{i,j \geq 0,\\ i+j \leq 2}} w_{ij}\mathbf{p}_{ij}B_{ij}^2(u,v)}{\sum_{\substack{i,j \geq 0,\\ i+j \leq 2}} w_{ij}B_{ij}^2(u,v)}, \tag{3}$$

where the bivariate n^{th}-degree Bernstein polynomial $B_{ij}^n(u,v)$ is

$$B_{ij}^n(u,v) = \frac{n!}{(n-i-j)!i!j!}(1-u-v)^{n-i-j}u^i v^j. \tag{4}$$

3.1. Constructing contour surfaces

We apply the 2D algorithm to each face of a tetrahedron and find the intersections of the contour surface with each face; we call these intersections *face-intersection curves*. Since there can be more than one surface passing through a quadratic tetrahedron, we connect the face-intersection curves end-to-end to form groups of curves that bound various portions of the contour surface, see Figure 3, similar to Hamann's method [7]. We classify each group according to the number of curves it contains:

- **Zero curves**. Either the contour surface is not present or the surface is "pill-shaped" and lies completely inside the tetrahedron.
- **One curve**. This is the case when one edge of the tetrahedron is equal to the contour value. We do not treat the curve in this case.
- **Two curves**. This is the case when, along one edge, the contour surface intersects two neighboring faces and looks similar to the "peel-of-an-orange slice," see Figure 4 (left).
- **Three curves**. The surface intersects three neighboring faces.
- **More than three curves**. The surface is bounded by several curves.

Simple cases occur when there are two or three face-intersection curves bounding the surface. An approximation to the contour surface is found by representing the surface with one triangular quadratic patch.

When there are two curves, we form one triangular patch by collapsing one side of the patch to the same point, see Figure 4. A "crack" along the curves would be introduced if we were to cut the surface across the middle to form two patches since the curves found in neighboring elements would not necessarily be split. Later in the rendering process—when the patch is tessellated either in software or hardware—the degenerate patch edge produces zero-area triangles (where two vertices have the same location). In terms of visualization, no significant problems are introduced since normal vectors for the vertices are computed analytically from the patch.

Three curves are trivially converted into one triangular patch by using the control points from the three boundary curves as patch control points.

More than three curves bounding the surface is non-trivial. Figure 5 shows an example of the type of complicated surface we must represent. We first form a polygon from the control nets of the face-intersection curves that bound the surface; this polygon is always closed but not necessarily convex. We follow these three steps to represent the surface with rational-quadratic patches:

1. Choose the shortest diagonal in the polygon to "split across." Here, a diagonal splits the polygon into two halves. We only consider diagonals that connect end-

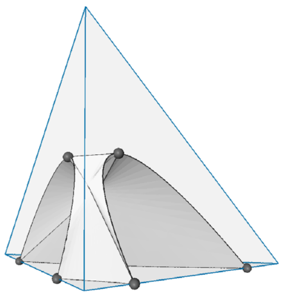

Figure 5: *Contour surface bounded by six face-intersection curves. Dark dots are the endpoints of the curves.*

points of the face-intersection curves. If n is the number of boundary curves, then the only valid diagonals to choose from are those that partition the polygon into two sets of $\frac{n}{2}$ curves (or additionally $\frac{n+1}{2}$ when n is odd).
2. Choose a control point and weight for the center of the diagonal.
3. Recurse on each half until the simple case of three boundary curves is reached.

There are several possibilities to choose the center control point location along the diagonal. Initially, we tried to choose this point by intersecting tangent planes of the contour surface. We chose not to compute the tangent planes exactly for performance reasons, instead, we estimated tangent planes using the control nets of the face-intersection curves. This method turned out to be inappropriate since the intersection quite often lay outside a tetrahedron.

We choose the more stable approach that considers various combinations of the center control points of the face-intersection curves (ensuring a point that lies inside a tetrahedron). For each diagonal, we only consider the center control points that are immediate neighbors to the endpoints of the diagonal. This approach always provides us with four control points, see Figure 6.

We consider all unique averages of each pair, group of three, and all four control points, in addition to each of the control points themselves. This method produces at most fifteen unique possibilities. For each point $\mathbf{b}_1^i$ in this set, we try to form a rational-quadratic curve $\mathbf{Q}_i(u)$ to represent the diagonal with endpoints $\mathbf{b}_0$ and $\mathbf{b}_2$. We compute the weight for the curve by intersecting the line connecting $\mathbf{b}_1^i$ and $\mathbf{m}$ with the contour surface, where $\mathbf{m} = \frac{\mathbf{b}_0+\mathbf{b}_2}{2}$, see [14] for how to compute the weight. We ignore a point $\mathbf{b}_1^i$ if there is not exactly one intersection with the contour surface. We choose the control point that produces the curve with least error. We estimate the error for curve $\mathbf{Q}_i(u)$ by evaluating it at parameter values $u = \frac{1}{6}, \frac{2}{6}, \frac{4}{6}$, and $\frac{5}{6}$ and then sampling the quadratic tetrahedron at these locations. We obtain an error estimate by summing the absolute difference between the sampled value and the contour value. If none of the control points can form a valid curve, then the diagonal is invalid and we mark the tetrahedron as containing a surface that is "too complex."

When a contour surface is too complex, we subdivide the tetrahedron to resolve the surface. These are the criteria that indicate when a surface is too complex:

1. There are no intersection curves with the faces but there exists a completely enclosed pill-shaped surface inside the tetrahedron. (Worsey and Farin[14] showed how to determine whether or not there exists such a surface.)
2. All the curves in a face-intersection group lie on the same face.
3. A surface bounded by more than three face-intersection curves cannot be split into patches.

4. Curved simplices

We find contours in curved quadratic elements by first finding the curve (or surface) in parameter space and then transforming the curve (or surface) to physical space. This section focuses on how to perform this transformation in the 2D and 3D cases to obtain quartic curves and surfaces that represents the contour through the curved quadratic elements.

4.1. The 2D case

We represent the contour in parameter space by a set of rational-quadratic curves. We consider each curve independently using a transformation from parameter space to physical space. For the rational-quadratic curve $\mathbf{Q}(u) : U^1 \longrightarrow U^2$, see equation (1), the weights w_0 and w_2 will always be one, since we require that $\mathbf{Q}(0) = \mathbf{p}_0$ and $\mathbf{Q}(1) = \mathbf{p}_2$. We define, in Bernstein-Bézier form, the bivariate quadratic mapping $\mathbf{T}(u,v) : U^2 \longrightarrow R^2$ of the standard triangle in parameter space, having corners (0,0), (1,0), and (0,1), to a curved triangle in physical space with six control points $\mathbf{b}_{ij} \in R^2, i,j \geq 0, i+j \leq 2$, as

$$\mathbf{T}(u,v) = \sum_{\substack{i,j \geq 0,\\ i+j \leq 2}} \mathbf{b}_{ij} B_{ij}^2(u,v). \tag{5}$$

Substituting (1), with weights w_0 and w_2 set to one, into (5) transforms $\mathbf{Q}(u)$ from parameter space to physical space, given by the mapping $\mathbf{T}(\mathbf{Q}(u)) : U^1 \longrightarrow U^2 \longrightarrow R^2$. We rearrange the terms and obtain

$$\mathbf{T}(\mathbf{Q}(u)) = \frac{\mathbf{c}_0 + \mathbf{c}_1 u + \mathbf{c}_2 u^2 + \mathbf{c}_3 u^3 + \mathbf{c}_4 u^4}{1 + g_1 u + g_2 u^2 + g_3 u^3 + g_4 u^4}. \tag{6}$$

The coefficients $\mathbf{c}_i$ and g_j are omitted since they are quite "involved." (However, one can easily compute these coefficients using a math package.)

We define the univariate rational-quartic curve $\mathbf{C}(u) : U^1 \longrightarrow R^2$ that we use to represent the contour curve in

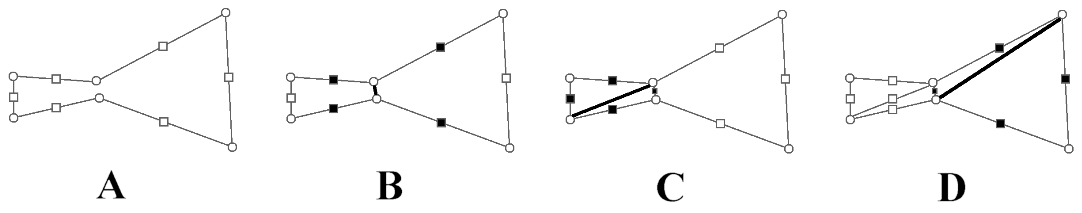

Figure 6: *Constructing four triangular patches from six face-intersection curves. Circles are endpoints and squares are center control points of face-intersection curves. Dark lines are chosen diagonal for splits. Dark squares are control points used to determine the center control point for each diagonal. An original polygon is shown in image A. Image B shows the first diagonal selection. Image C shows the diagonal selection for the left half. Image D shows the diagonal selection for the right half.*

physical space, in Bernstein-Bézier form, having five control points $\mathbf{d}_i \in R^2$ and five weights $m_i, 0 \le i \le 4$, $m_i \ge 0$, given by

$$\mathbf{C}(u) \;=\; \frac{\sum_{0 \le i \le 4} m_i \mathbf{d}_i B_i^4(u)}{\sum_{0 \le i \le 4} m_i B_i^4(u)}. \tag{7}$$

In order to represent (6) by a rational-quartic curve $\mathbf{C}(u)$, we must rewrite (7) to be in the form of (6). The weights m_0 and m_4 will always be one, since we require that $\mathbf{C}(0) = \mathbf{d}_0$ and $\mathbf{C}(1) = \mathbf{d}_4$. By substituting weights m_0 and m_4 set to one into (7) we obtain

$$\mathbf{C}(u) \;=\; \frac{\mathbf{d}_0 + \mathbf{h}_1 u + \mathbf{h}_2 u^2 + \mathbf{h}_3 u^3 + \mathbf{h}_4 u^4}{1 + r_1 u + r_2 u^2 + r_3 u^3 + r_4 u^4}, \tag{8}$$

which has the same form as (6). The coefficients $\mathbf{h}_i$ and r_j are omitted here.

Thus, the parametrization of the control net of $\mathbf{C}(u)$ in physical space in terms of (1) and (5) is defined by the values

$$\begin{aligned} m_1 &= w_1, \\ m_2 &= \tfrac{1}{3}(1 + 2{w_1}^2), \\ m_3 &= w_1, \end{aligned} \tag{9}$$

and

$$\begin{aligned} \mathbf{d}_0 &= \mathbf{c}_0, \\ \mathbf{d}_1 &= \tfrac{1}{4}\,\tfrac{4\mathbf{c}_0+\mathbf{c}_1}{w_1}, \\ \mathbf{d}_2 &= \tfrac{1}{2}\,\tfrac{2\mathbf{c}_0+\mathbf{c}_1+\frac{1}{3}\mathbf{c}_2}{\frac{1}{3}(1+2{w_1}^2)}, \\ \mathbf{d}_3 &= \tfrac{1}{4}\,\tfrac{4\mathbf{c}_0+3\mathbf{c}_1+2\mathbf{c}_2+\mathbf{c}_3}{w_1}, \text{ and} \\ \mathbf{d}_4 &= \mathbf{c}_0 + \mathbf{c}_1 + \mathbf{c}_2 + \mathbf{c}_3 + \mathbf{c}_4. \end{aligned} \tag{10}$$

Examining the transformation of the control net of $\mathbf{Q}(u)$—defined by the three points $\mathbf{p}_0$, $\mathbf{p}_1$, and $\mathbf{p}_2$—reveals some similarities between the control net of $\mathbf{Q}(u)$ and that of $\mathbf{C}(u)$. We find the similarities by transforming the two tangent lines $\mathbf{T}_L$ and $\mathbf{T}_R$ from parameter space to physical space, where $\mathbf{T}_L$ is the line segment connecting $\mathbf{p}_1$ and $\mathbf{p}_0$ and $\mathbf{T}_R$ is the line segment connecting $\mathbf{p}_1$ and $\mathbf{p}_2$. Two quadratic curves in physical space represent these tangent lines. We find the curves by fitting two quadratic curves—$\mathbf{l}(u)$ and $\mathbf{r}(u)$—to the transformed points $L = \{\mathbf{p}_0, \frac{\mathbf{p}_0+\mathbf{p}_1}{2}, \mathbf{p}_1\}$ and $R = \{\mathbf{p}_2, \frac{\mathbf{p}_2+\mathbf{p}_1}{2}, \mathbf{p}_1\}$, respectively. We determine the two curves by solving for the center control point for $\mathbf{T}(u) : U^1 \longrightarrow R^2$ when $u = \frac{1}{2}$, where

$$\mathbf{T}(u) \;=\; \sum_{0 \le i \le 2} \mathbf{b}_i B_i^2(u), \tag{11}$$

using three control points $\mathbf{b}_i \in R^2, 0 \le i \le 2$. The center control points for $\mathbf{l}(u)$ and $\mathbf{r}(u)$ turn out to be $\mathbf{d}_1$ and $\mathbf{d}_3$, respectively, and $\mathbf{d}_0$ and $\mathbf{d}_4$ turn out to be $\mathbf{p}_0$ and $\mathbf{p}_2$ transformed to physical space, respectively, (control net for $\mathbf{C}(u)$, see Figure 7). We prove this property in the Appendix.

Using this observation, we obtain four of the five required points that define the control net of $\mathbf{C}(\mathbf{u})$, given by

$$\begin{aligned} \mathbf{d}_0 &= \mathbf{T}(\mathbf{p}_0), \\ \mathbf{d}_1 &= \mathbf{l}_1, \\ \mathbf{d}_3 &= \mathbf{r}_1, \text{and} \\ \mathbf{d}_4 &= \mathbf{T}(\mathbf{p}_2), \end{aligned} \tag{12}$$

where $\mathbf{l}_1$ and $\mathbf{r}_1$ are obtained as described in the Appendix, using equations (21) and (22), respectively.

4.2. The 3D case

We represent the isosurface in parameter space by a set of rational-quadratic patches. We consider each patch independently using a transformation from parameter space to physical space. For the rational-quadratic patch $\mathbf{Q}(u,v) : U^2 \longrightarrow U^3$, see equation (3), the weights w_{00}, w_{20}, and w_{02} are one, since we require that $\mathbf{Q}(0,0) = \mathbf{p}_{00}$, $\mathbf{Q}(1,0) = \mathbf{p}_{20}$, and $\mathbf{Q}(0,1) = \mathbf{p}_{02}$. We define, in Bernstein-Bézier form, the trivariate quadratic mapping $\mathbf{T}(u,v,w) : U^3 \longrightarrow R^3$ of the standard tetrahedron in parameter space, having corners (0,0,0), (1,0,0), (0,1,0), and (0,0,1), to a curved tetrahedron having ten control points $\mathbf{b}_{ijk} \in R^3, i,j,k \ge 0$, $i+j+k \le 2$, as

$$\mathbf{T}(u,v,w) \;=\; \sum_{\substack{i,j,k \ge 0, \\ i+j+k \le 2}} \mathbf{b}_{ijk} B_{ijk}^2(u,v,w), \tag{13}$$

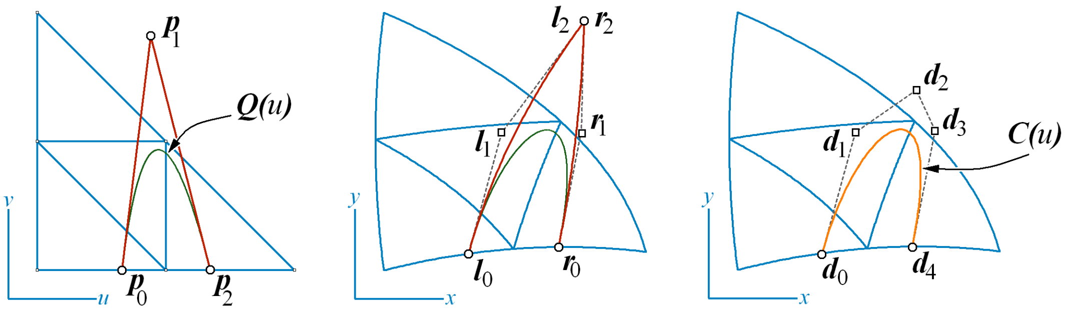

Figure 7: *Relationship between control net of $\mathbf{Q}(u)$ and control net of $\mathbf{C}(u)$. Left image shows rational-quadratic curve $\mathbf{Q}(u)$ in parameter space. Middle image shows control net of $\mathbf{Q}(u)$ transformed into physical space. Right image shows rational-quartic curve $\mathbf{C}(u)$ resulting from transforming $\mathbf{Q}(u)$ into physical space. It turns out that $\mathbf{l}_1 = \mathbf{d}_1$, $\mathbf{r}_1 = \mathbf{d}_3$, $\mathbf{T}(\mathbf{p}_0) = \mathbf{d}_0$, and $\mathbf{T}(\mathbf{p}_2) = \mathbf{d}_4$.*

where the trivariate n^{th}-degree Bernstein polynomial $B^n_{ijk}(u,v,w)$ is

$$B^n_{ijk}(u,v,w) = \frac{n!}{(n-i-j-k)!i!j!k!} (1-u-v-w)^{n-i-j-k} u^i v^j w^k. \quad (14)$$

Substituting (3) into (13) transforms $\mathbf{Q}(u,v)$ from parameter space to physical space, $\mathbf{T}(\mathbf{Q}(u,v)) : U^2 \longrightarrow U^3 \longrightarrow R^3$. This mapping is defined as

$$\mathbf{T}(\mathbf{Q}(u,v)) = \frac{\left[\begin{bmatrix} \mathbf{c}_0 & 0 & 0 & 0 \\ \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 & \mathbf{c}_4 \\ \mathbf{c}_5 & \mathbf{c}_6 & \mathbf{c}_7 & 0 \\ \mathbf{c}_8 & \mathbf{c}_9 & 0 & 0 \\ \mathbf{c}_{10} & 0 & 0 & 0 \\ \mathbf{c}_{11} & 0 & 0 & 0 \\ \mathbf{c}_{12} & 0 & 0 & 0 \\ \mathbf{c}_{13} & 0 & 0 & 0 \\ \mathbf{c}_{14} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}\right]^T \begin{bmatrix} 1 \\ u \\ u^2 \\ u^3 \\ u^4 \\ v \\ v^2 \\ v^3 \\ v^4 \end{bmatrix}}{\left[\begin{bmatrix} 1 & 0 & 0 & 0 \\ g_1 & g_2 & g_3 & g_4 \\ g_5 & g_6 & g_7 & 0 \\ g_8 & g_9 & 0 & 0 \\ g_{10} & 0 & 0 & 0 \\ g_{11} & 0 & 0 & 0 \\ g_{12} & 0 & 0 & 0 \\ g_{13} & 0 & 0 & 0 \\ g_{14} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}\right]^T \begin{bmatrix} 1 \\ u \\ u^2 \\ u^3 \\ u^4 \\ v \\ v^2 \\ v^3 \\ v^4 \end{bmatrix}}. \quad (15)$$

Here, we omit the coefficients $\mathbf{c}_i$ and g_j since they are quite complicated.

We define the bivariate rational-quartic surface $\mathbf{C}(u,v) : U^2 \longrightarrow R^3$ used to represent the contour surface in physical space, in Bernstein-Bézier form, having fifteen control points $\mathbf{d}_{ij} \in R^3$ and fifteen weights $m_{ij}, i,j \geq 0$, $i+j \leq 4$, $m_{ij} \geq 0$, given by

$$\mathbf{C}(u,v) = \frac{\sum_{\substack{i,j \geq 0, \\ i+j \leq 4}} m_{ij}\mathbf{d}_{ij}B^4_{ij}(u,v)}{\sum_{\substack{i,j \geq 0, \\ i+j \leq 4}} m_{ij}B^4_{ij}(u,v)}. \quad (16)$$

In order to represent (15) by a rational-quartic patch $\mathbf{C}(u,v)$, we must rewrite (16) in the form of (15). The weights m_{00}, m_{40}, and m_{04} are all one, since we require that $\mathbf{C}(0,0) = \mathbf{d}_{00}$, $\mathbf{C}(1,0) = \mathbf{d}_{40}$, and $\mathbf{C}(0,1) = \mathbf{d}_{04}$. Substituting weights m_{00}, m_{40}, and m_{04} set to one into (16) allows us to rearrange the terms so that it takes on the same form as equation (15).

Thus, the parametrization of the control net of $\mathbf{C}(u,v)$ in physical space in terms of (3) and (13) is given by the values

$$\begin{aligned} m_{10} &= w_{10}, \\ m_{20} &= \tfrac{1}{3}(1+2{w_{10}}^2), \\ m_{30} &= w_{10}, \\ m_{01} &= w_{01}, \\ m_{11} &= \tfrac{1}{3}(w_{11}+2w_{10}w_{01}), \\ m_{21} &= \tfrac{1}{3}(w_{01}+2w_{10}w_{11}), \\ m_{31} &= w_{11}, \\ m_{02} &= \tfrac{1}{3}(1+2{w_{01}}^2), \\ m_{12} &= \tfrac{1}{3}(w_{10}+2w_{01}w_{11}), \\ m_{22} &= \tfrac{1}{3}(1+2{w_{11}}^2), \\ m_{03} &= w_{01}, \\ m_{13} &= w_{11}, \end{aligned} \quad (17)$$

and

$$
\begin{aligned}
\mathbf{d}_{00} &= \mathbf{c}_0, \\
\mathbf{d}_{10} &= \frac{1}{4}\frac{4\mathbf{c}_0+\mathbf{c}_1}{w_{10}}, \\
\mathbf{d}_{20} &= \frac{1}{2}\frac{2\mathbf{c}_0+\mathbf{c}_1+\frac{1}{3}\mathbf{c}_5}{\frac{1}{3}(1+2{w_{10}}^2)}, \\
\mathbf{d}_{30} &= \frac{1}{4}\frac{4\mathbf{c}_0+3\mathbf{c}_1+2\mathbf{c}_5+\mathbf{c}_8}{w_{10}}, \\
\mathbf{d}_{40} &= \mathbf{c}_0+\mathbf{c}_1+\mathbf{c}_5+\mathbf{c}_8+\mathbf{c}_{10}, \\
\mathbf{d}_{01} &= \frac{1}{4}\frac{4\mathbf{c}_0+\mathbf{c}_{11}}{w_{01}}, \\
\mathbf{d}_{11} &= \frac{1}{4}\frac{4\mathbf{c}_0+\mathbf{c}_1+\mathbf{c}_{11}+\frac{1}{3}\mathbf{c}_2}{\frac{1}{3}(2w_{10}w_{01}+w_{11})}, \\
\mathbf{d}_{21} &= \frac{1}{4}\frac{4\mathbf{c}_0+2\mathbf{c}_1+\mathbf{c}_{11}+\frac{2}{3}(\mathbf{c}_2+\mathbf{c}_5)+\frac{1}{3}\mathbf{c}_6}{\frac{1}{3}(2w_{10}w_{11}+w_{01})}, \\
\mathbf{d}_{31} &= \frac{1}{4}\frac{4\mathbf{c}_0+3\mathbf{c}_1+2\mathbf{c}_5+\mathbf{c}_6+\mathbf{c}_2+\mathbf{c}_{11}+\mathbf{c}_8+\mathbf{c}_9}{w_{11}}, \\
\mathbf{d}_{02} &= \frac{1}{2}\frac{2\mathbf{c}_0+\mathbf{c}_{11}+\frac{1}{3}\mathbf{c}_{12}}{\frac{1}{3}(1+2{w_{01}}^2)}, \\
\mathbf{d}_{12} &= \frac{1}{4}\frac{4\mathbf{c}_0+2\mathbf{c}_{11}+\mathbf{c}_1+\frac{2}{3}(\mathbf{c}_2+\mathbf{c}_{12})+\frac{1}{3}\mathbf{c}_3}{\frac{1}{3}(2w_{01}w_{11}+w_{10})}, \\
\mathbf{d}_{22} &= \frac{1}{2}\frac{2\mathbf{c}_0+\mathbf{c}_1+\mathbf{c}_{11}+\frac{2}{3}\mathbf{c}_2+\frac{1}{3}(\mathbf{c}_3+\mathbf{c}_5+\mathbf{c}_6+\mathbf{c}_7+\mathbf{c}_{12})}{\frac{1}{3}(1+2{w_{11}}^2)}, \\
\mathbf{d}_{03} &= \frac{1}{4}\frac{3\mathbf{c}_{11}+4\mathbf{c}_0+\mathbf{c}_{13}+2\mathbf{c}_{12}}{w_{01}}, \\
\mathbf{d}_{13} &= \frac{1}{4}\frac{\mathbf{c}_4+\mathbf{c}_{13}+\mathbf{c}_2+\mathbf{c}_3+2\mathbf{c}_{12}+\mathbf{c}_1+3\mathbf{c}_{11}+4\mathbf{c}_0}{w_{11}}, \text{ and} \\
\mathbf{d}_{04} &= \mathbf{c}_0+\mathbf{c}_{11}+\mathbf{c}_{12}+\mathbf{c}_{14}+\mathbf{c}_{13}.
\end{aligned}
\tag{18}
$$

5. Results

We show examples of isosurfaces for complex data sets in Figures 8, 9, and 10. In these figures, an isosurface of a curved-quadratic tetrahedral representation of a "spherical" data set ($x^2+y^2+z^2=c$) is shown. This data set consists of 320 curved-quadratic tetrahedra. The extracted isosurface consists of 308 triangular rational-quartic patches.

Figures 11 and 12 show the isosurface of a data set consisting of 15918 quadratic tetrahedra representing "eight spheres." The curved quadratic-tetrahedral mesh uses the same 90° twist as the one shown in Figure 8. The resulting contour surfaces consist of 6112 patches.

6. Conclusions

In the bivariate case, a rational quadratic can represent a contour curve exactly since it is a conic section. In the trivariate case, we can represent the intersection of the contour surface with each face exactly. However, the contour surface inside a tetrahedron cannot be represented exactly with a rational-quadratic patch[4]. Some degree of error is inherent in the surface representations we produce because of the

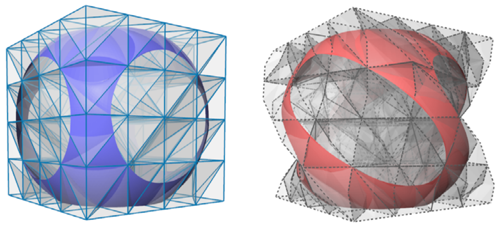

Figure 8: *Left image shows "un-twisted" mesh containing only linear-edge quadratic tetrahedra. Right image shows twisted mesh containing curved-quadratic tetrahedra. The mesh is twisted by* 90° *comparing top and bottom faces of entire mesh configuration.*

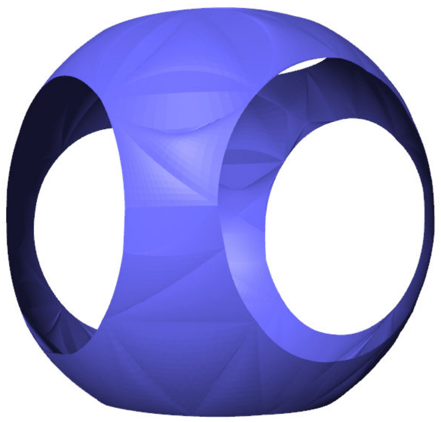

Figure 9: *Enlargement of rational-quadratic contour surface extracted from un-twisted mesh shown in Figure 8 (left); 320 quadratic tetrahedra.*

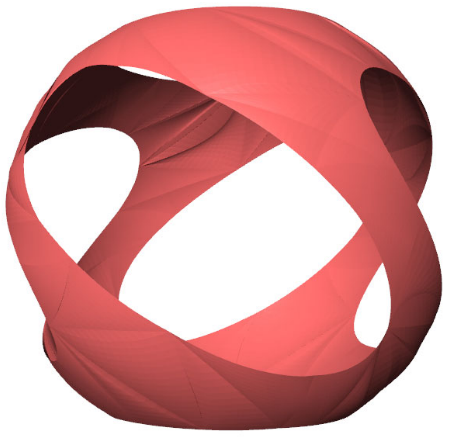

Figure 10: *Enlargement of rational-quartic contour surface extracted from twisted mesh shown in Figure 8 (right); 320 curved-quadratic tetrahedra.*

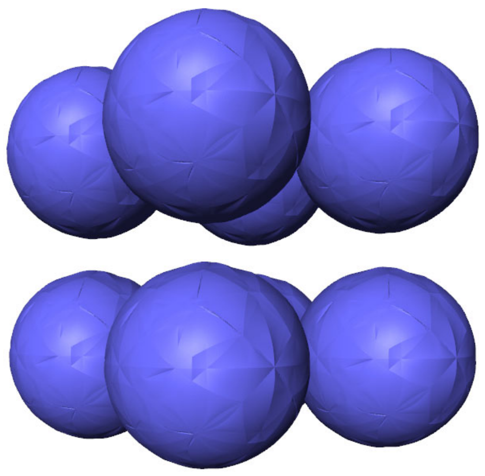

Figure 11: *Rational-quadratic contour surface extracted from un-twisted mesh consisting of 15918 quadratic tetrahedra.*

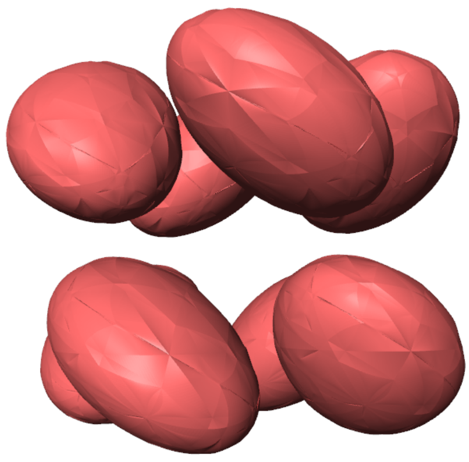

Figure 12: *Rational-quartic contour surface extracted from 15918 curved-quadratic tetrahedra.*

patches we chose to use. An alternative is to tessellate (approximate to) the quadratic tetrahedron with linear tetrahedra and then to extract the isosurface from these linear elements. To obtain an isosurface with less approximation error one would need to use several linear tetrahedra per quadratic tetrahedron, which is undesirable for two reasons: first, the performance penalty for the tessellation is too high; and second, the amount of data sent to the video hardware would increase. We can either send a few curved patches or several linear triangles.

We only guarantee $C0$-continuity between the rational-quadratic patches. We will investigate how to ensure $C1$-continuity.

When subdividing quadratic tetrahedra—in the case where the contour surface is too complex—we are now considering the use of a subdivision scheme that does not produce "hanging nodes" or "skinny tetrahedra." Longest-edge bisection, in this application, tends to produce skinny tetrahedra, which leads to poor patches. Ideally, a method should preserve the original shape of the initial tetrahedra. Examples of such methods are red-green subdivision [6] and diamond subdivision [5]. We are currently integrating quadratic tetrahedra and the contouring method described here into the view-dependant visualization method described in [5].

Acknowledgements

This work was supported by the National Science Foundation under contract ACI 9624034 (CAREER Award) and ACI 0222909, through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the National Institute of Mental Health and the National Science Foundation under contract NIMH 2 P20 MH60975-06A2; the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B503159; and the Lawrence Berkeley National Laboratory. We thank the members of the Visualization and Graphics Research Group at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis, and our colleagues in B-Division at the Lawrence Livermore National Laboratory.

References

1. C.L. Bajaj, *Free-form modeling with implicit surface patches*, Implicit Surfaces, J. Bloomenthal and B. Wyvill eds., Morgan Kaufman Publishers, San Francisco, CA, 1996
2. B.K. Bloomquist, *Contouring Trivariate Surfaces*, Masters Thesis, Arizona State University, Computer Science Department, Tempe, AZ, 1990
3. R.D. Cook, D.S. Malkus, and M.E. Plesha, *Concepts*

and Applications of Finite Element Analysis, John Wiley & Sons, New York, 1989

4. G. Farin, *Curves and Surfaces for CAGD*, Fifth edition, Morgan Kaufmann Publishers Inc., San Francisco, CA, 2001
5. B.F. Gregorski, M.A. Duchaineau, P. Lindstrom, V. Pascucci, and K.I. Joy *Interactive view-dependent rendering of large isosurfaces*, in Proceedings of the IEEE Visualization 2002, R. Moorhead, M. Gross, K.I. Joy, eds., , pp. 475-482, 2002
6. R. Grosso, C. Lürig, and T. Ertl, *The multilevel finite element method for adaptive mesh optimization and visualization of volume data*, in Proceedings Visualization 97, R. Yagel and H. Hagen, eds., 1997
7. B. Hamann, *Modeling contours of trivariate data*, Mathematical Modelling and Numerical Analysis (Modelisation Mathematique et Analysis Numerique) 26(1), Gauthier-Villars, France, pp. 51-75, 1992
8. B. Hamann, I.J. Trotts, and G. Farin, *On approximating contours of the piecwise trilinear interpolant using triangular rational-quadratic Bézier patches*, IEEE Transactions on Visualization and Computer Graphics, 3(3), 315– 337, 1997
9. R. Khardekar and D. Thompson, *Rendering higher order finite element surfaces in hardware*, in Proceedings of GRAPHITE 2003, M. Adcock, I. Gwilt, and L. Y. Tsui, eds., pp. 211-ff, ACM, Feb 11-14, Melbourne, Australia, 2003
10. S. Marlow and M.J.D. Powell, *A Fortran subroutine for plotting the part of a conic that is inside a given triangle*, Report no. R 8336, Atomic Energy Research Establishment, Harwell, United Kingdom, 1976
11. N. Max, P. Williams, and C. Silva, *Cell projection of meshes with non-planar faces*, in "Scientific Visualization, Dagstuhl 2000", 2002
12. A. Vlachos, J. Peters, C. Boyd and J.L. Mitchell, *Curved PN Triangles*, ACM Symposium on Interactive 3D Graphics 2001, pp. 159-166, 2001
13. D. F. Wiley, H. R. Childs, B. Hamann, K. I. Joy, and N. L. Max, *Best quadratic spline approximation for hierarchical visualization*, in Data Visualization 2002, Proceedings of VisSym 2002, D. Ebert, P. Brunet, and I. Navazo, eds., pp. 133-140, 2002
14. A.J. Worsey and G. Farin, *Contouring a bivariate quadratic polynomial over a triangle*, Computer Aided Geometric Design 7 (1–4), 337–352, 1990

Appendix

We prove the similarities between the control net of $\mathbf{Q}(\mathbf{u})$ and that of $\mathbf{C}(\mathbf{u})$, as illustrated in Figure 7. We first prove this property for the left tangent line formed by $\mathbf{p}_0$ and $\mathbf{p}_1$. We must show that $\mathbf{l}_0 = \mathbf{d}_0$ and $\mathbf{l}_1 = \mathbf{d}_1$. First, we find that $\mathbf{l}_0 = \mathbf{T}(\mathbf{p}_0)$ and $\mathbf{l}_2 = \mathbf{T}(\mathbf{p}_1)$. These variables are given as

$$\begin{aligned}\mathbf{l}_0 &= \left[\left[\begin{array}{ccc}\mathbf{b}_{20} & \mathbf{b}_{11} & \mathbf{b}_{10}\\ \mathbf{b}_{11} & \mathbf{b}_{02} & \mathbf{b}_{01}\\ \mathbf{b}_{10} & \mathbf{b}_{01} & \mathbf{b}_{00}\end{array}\right]\right.\\ &\qquad \left.\left[\begin{array}{c}u_0\\ v_0\\ 1-u_0-v_0\end{array}\right]\right]^T \qquad (19)\\ &\qquad \left[\begin{array}{c}u_0\\ v_0\\ 1-u_0-v_0\end{array}\right]\\ &= \mathbf{c}_0,\end{aligned}$$

thus, we find that $\mathbf{l}_0 = \mathbf{d}_0$, and

$$\begin{aligned}\mathbf{l}_2 = &\left[\left[\begin{array}{ccc}\mathbf{b}_{20} & \mathbf{b}_{11} & \mathbf{b}_{10}\\ \mathbf{b}_{11} & \mathbf{b}_{02} & \mathbf{b}_{01}\\ \mathbf{b}_{10} & \mathbf{b}_{01} & \mathbf{b}_{00}\end{array}\right]\right.\\ &\qquad \left.\left[\begin{array}{c}u_1\\ v_1\\ 1-u_1-v_1\end{array}\right]\right]^T \qquad (20)\\ &\qquad \left[\begin{array}{c}u_1\\ v_1\\ 1-u_1-v_1\end{array}\right],\end{aligned}$$

where $\mathbf{c}_0$ is obtained from equation (6). We fit a quadratic curve to $\{\mathbf{l}_0, \mathbf{T}(\frac{\mathbf{p}_0+\mathbf{p}_1}{2}), \mathbf{l}_2\}$ and find $\mathbf{l}_1$ to be

$$\begin{aligned}\mathbf{l}_1 \quad = &\\ &\left[\left[\begin{array}{ccc}\mathbf{b}_{00} & 0 & 0\\ \mathbf{b}_{10}-\mathbf{b}_{00} & \mathbf{b}_{00}+\mathbf{b}_{20}-2\mathbf{b}_{10} & \mathbf{b}_{00}+\mathbf{b}_{11}-\mathbf{b}_{10}-\mathbf{b}_{01}\\ \mathbf{b}_{01}-\mathbf{b}_{00} & \mathbf{b}_{00}+\mathbf{b}_{11}-\mathbf{b}_{10}-\mathbf{b}_{01} & \mathbf{b}_{00}+\mathbf{b}_{02}-2\mathbf{b}_{01}\\ \mathbf{b}_{10}-\mathbf{b}_{00} & 0 & 0\\ \mathbf{b}_{01}-\mathbf{b}_{00} & 0 & 0\end{array}\right]\left[\begin{array}{c}1\\ u_1\\ v_1\end{array}\right]\right]^T\\ &\left[\begin{array}{c}1\\ u_0\\ v_0\\ u_1\\ v_1\end{array}\right]. \qquad (21)\end{aligned}$$

By substituting the solutions for $\mathbf{c}_0$ and $\mathbf{c}_1$, from equation (6), into the solution for $\mathbf{d}_1$ from (10) it follows that $\mathbf{d}_1 = \mathbf{l}_1$. A similar proof can be constructed to show that $\mathbf{r}_0 = \mathbf{T}(\mathbf{p}_2) = \mathbf{d}_4$ and $\mathbf{r}_1 = \mathbf{d}_3$, where $\mathbf{r}_1$ is given as

$$\mathbf{r}_1 = \left[\begin{bmatrix} \mathbf{b}_{00} & 0 & 0 \\ \mathbf{b}_{10}-\mathbf{b}_{00} & \mathbf{b}_{00}+\mathbf{b}_{20}-2\mathbf{b}_{10} & \mathbf{b}_{00}+\mathbf{b}_{11}-\mathbf{b}_{10}-\mathbf{b}_{01} \\ \mathbf{b}_{01}-\mathbf{b}_{00} & \mathbf{b}_{00}+\mathbf{b}_{11}-\mathbf{b}_{10}-\mathbf{b}_{01} & \mathbf{b}_{00}+\mathbf{b}_{02}-2\mathbf{b}_{01} \\ \mathbf{b}_{10}-\mathbf{b}_{00} & 0 & 0 \\ \mathbf{b}_{01}-\mathbf{b}_{00} & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ u_1 \\ v_1 \end{bmatrix}\right]^T \begin{bmatrix} 1 \\ u_2 \\ v_2 \\ u_1 \\ v_1 \end{bmatrix}. \tag{22}$$

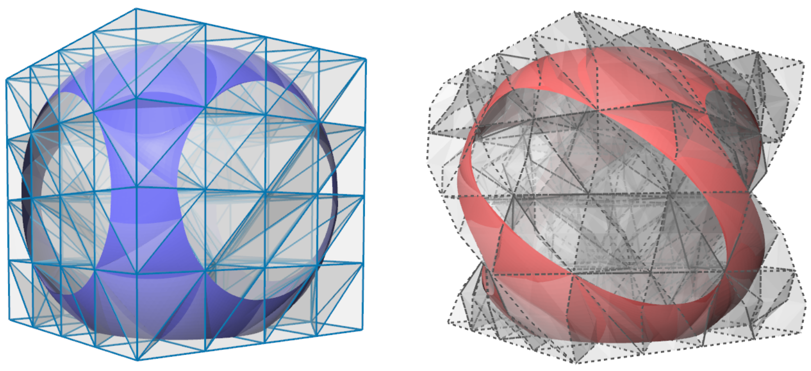

Figure 8: Left image shows "un-twisted" mesh containing only linear-edge quadratic tetrahedra. Right image shows twisted mesh containing curved-quadratic tetrahedra. The mesh is twisted by 90° comparing top and bottom faces of entire mesh configuration.

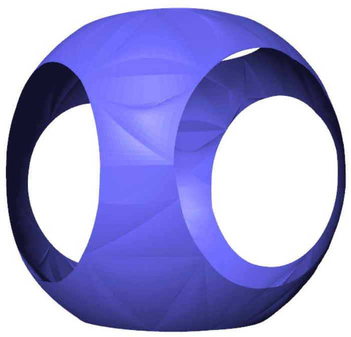

Figure 9: Enlargement of rational-quadratic contour surface extracted from un-twisted mesh shown in Figure 8 (left); 320 quadratic tetrahedra.

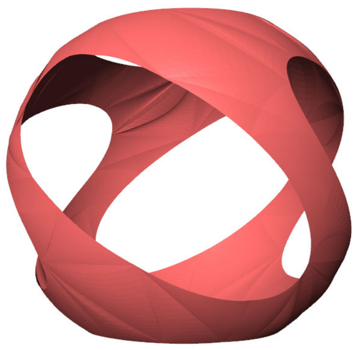

Figure 10: Enlargement of rational-quartic contour surface extracted from twisted mesh shown in Figure 8 (right); 320 curved-quadratic tetrahedra.

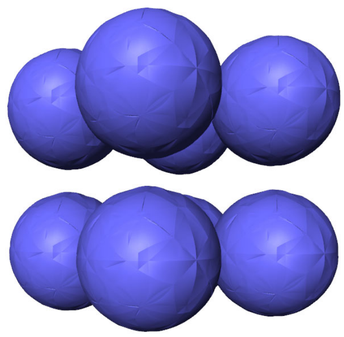

Figure 11: Rational-quadratic contour surface extracted from un-twisted mesh consisting of 15918 quadratic tetrahedra.

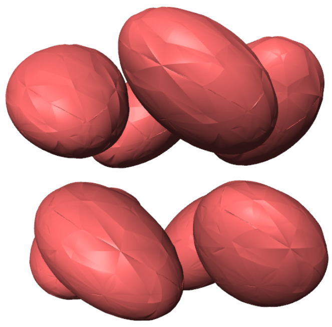

Figure 12: Rational-quartic contour surface extracted from 15918 curved-quadratic tetrahedra.