J-splines

Jarek Rossignac
School of Interactive Computing, College of Computing, Georgia Institute of Technology, Atlanta, GA
http://www.gvu.gatech.edu/~jarek

Scott Schaefer
Department of Computer Science, 3112 Texas A&M University, College Station, TX
http://faculty.cs.tamu.edu/schaefer

Abstract

Both the 4-point and the uniform cubic B-spline subdivisions double the number of vertices of a closed-loop polygon ${}^kP$ and produce sequences of vertices f_j and b_j respectively. We study the J-spline subdivision scheme J_s, introduced by Maillot and Stam, which blends these two methods to produce vertices of the form $v_j=(1-s)f_j+sb_j$. Iterative applications of J_s yield a family of limit curves, the shape of which is parameterized by s. They include four-point subdivision curves (J_0), uniform cubic B-spline curves (J_1), and uniform quintic B-spline curves ($J_{1.5}$). We show that the limit curve is at least C^1 when $-1.7 \le s \le 5.8$, C^2 when $0<s<4$, C^3 when $1<s \le 2.8$, and C^4 when $s=3/2$, even though 4-point yields only C^1 curves and cubic B-spline yields only C^2 curves. We generalize the J_s scheme to a two-parameter family $J_{a,b}$ and propose data-dependent and data-independent solutions for computing values of parameters a and b that minimize various objective functions (distance to the control vertices, deviation from the control polygon, change in surface area, and popping when switching levels of subdivision in multi-resolution rendering). We extend the J-spline subdivision to open curves and to a smooth surface subdivision scheme for quad-meshes with arbitrary connectivity.

1. Introduction

Because of its simplicity and expressive power, subdivision has become a standard tool for modeling curves and surfaces in CAD, Graphics, and Animation. Subdivision is a set of rules that refine polygonal curves (in any dimension) by increasing the number of their vertices. Applying a subdivision operator repeatedly to a control polygon produces a sequence of curves that, if the subdivision rules are chosen correctly, converge to a smooth limit curve.

There are many ways of choosing subdivision rules. We concentrate on *linear subdivision schemes,* where the new vertices are linear combinations of old ones, and restrict ourselves to *binary subdivisions*, where the number of vertices doubles at each refinement. One such scheme [Ros04] averages the results for cubic B-spline and 4-point subdivisions. Averaging these rules generates C^2 curves [PR08]. This was unexpected, since the 4-point rule produces C^1 curves. Inspired by this discovery, we investigate a generalized form based on an arbitrary affine combination of the cubic B-spline and 4-point subdivision rules. Specifically, we introduce what we named the J-spline scheme, $J_{a,b}$, whose rules are of the form

$$\begin{aligned} {}^{k+1}P_{2j} &= (a\,{}^kP_{j-1} + (8-2a)\,{}^kP_j + a\,{}^kP_{j+1})/8 \\ {}^{k+1}P_{2j+1} &= ((b-1)\,{}^kP_{j-1} + (9-b)\,{}^kP_j + (9-b)\,{}^kP_{j+1} + (b-1)\,{}^kP_{j+2})/16 \end{aligned}$$

where ${}^kP_j$ represents the jth control point at the kth level of subdivision.

Note that setting a=b yields the subclass of J-splines, $J_{s,s}$ which we denote J_s, originally proposed by Maillot and Stam [MS01]. Several known curve subdivision schemes fall in this subclass. Specifically, J_0 is the 4-point scheme [NDG87, DD89], $J_{1/2}$ is the scheme promoted in [Ros04], and J_1 is the uniform cubic B-spline scheme [Sab02, LR80] (Fig. 1).

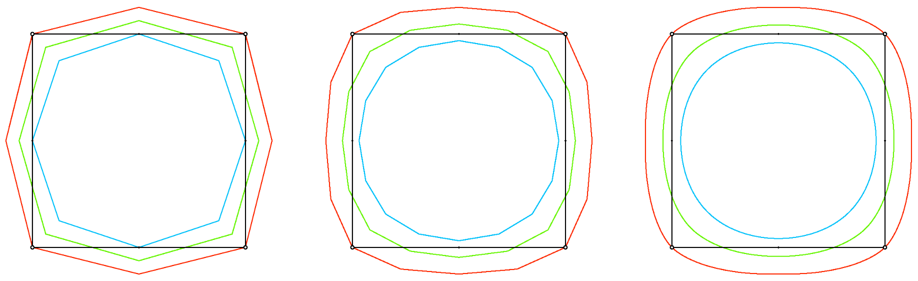

Fig. 1: Results of 1, 2, and 6 refinements of 4-point J_0 (outer), the compromise [Ros04] $J_{1/2}$ (intermediate), cubic B-spline J_1 (inner).

2. Previous Work

Several researchers have considered subdivision schemes with a tunable parameter. For example, *splines in tension* [Sch96, Cli74] are a generalization of polynomial splines with a tunable parameter that controls the degree of "tightness" of the curve. However, these subdivision rules are not linear in the tension parameter, which complicates the analysis of their smoothness. Barsky et al. [BB83] created a generalization of B-splines, called Beta-splines, which provides bias and tension controls. Later, Dyn et al. [NDG87] developed an interpolatory 4-point subdivision scheme with a tunable parameter that blends between local cubic Lagrange interpolation and linear interpolation. The smoothness of the curves generated with this subdivision scheme depends on the parameter, but is at most C^1. Maillot and Stam [MS01] proposed a one-parameter family of subdivision schemes that corresponds to the J_s subclass of our J-splines (they used $\alpha=1-s$ as the blending parameter). They also mention applications to subdivision surfaces and silhouette accuracy, but do not provide any smoothness analysis in the curve or surface case, nor any automatic parameter optimization. More recently Dyn et al. [DFH05] introduced a subdivision scheme based on local cubic Lagrange interpolation blended with Chaikin's subdivision scheme [Cha74] for C^1 B-spline curves. In the present paper, we prove strong and unanticipated continuity results for J_s curves, including the fact that J_s subdivision can generate C^3 and C^4 curves.

Despite the fact that several subdivisions schemes with tunable parameters exist, there has been little work on optimizing these parameters to achieve geometric properties, such as vertex or mid-edge point interpolation or area preservation. Dyn et al. [DFH05] show that their method generates C^2 curves for a large range of parameter values and optimize this parameter to create a subdivision scheme that is as close as possible to being interpolatory. Most subdivision optimization has focused on fitting subdivision surfaces to other data [LLS01, MK04]. Typically these methods are data dependent (their optimization is dependent on the input data) and either use parametric correspondence or attempt to find some geometric correspondence to match the given data. Halstead et al. [HKD93] also show how to set up an optimization problem to find the control vertices for a Catmull-Clark surface that minimizes various energy functionals such as thin-plate energy. Since these optimizations are model dependent, they must be recomputed whenever vertices are modified. To minimize this optimization cost, we aim to choose the optimal parameters for our subdivision scheme independently of any given data and ensure that our curves still maintain local control. To do so, we generalize the J_s scheme [MS01] to a two-parameter family $J_{a,b}$ and propose data-dependent and data-independent solutions for computing values of parameters a and b that minimize various objective functions (closeness to uniform B-splines, distance to the control vertices, deviation from the control polygon, change in surface area, and popping when switching levels of subdivision in multi-resolution rendering). We also extend the J-spline subdivision to open curves and to a smooth surface subdivision scheme for meshes with arbitrary connectivity.

3. Continuity of J_s curves

The J_s subdivision of a given control polygon converges to a limit curve whose shape and degree of smoothness depend on the value of the parameter s (Fig. 2). We know the smoothness of the curves generated by the two parent subdivision schemes (J_0 is C^1 and J_1 is C^2). Naively we might assume that, for other values of s, J_s will inherit the lower C^1 smoothness of the 4-point scheme. We show here that this is not the case.

Consider two loops, $P = \{P_0, P_1,\dots P_k\}$ and $Q = \{Q_0, Q_1,\dots Q_k\}$. Let the subdivision rule $L_s(P,Q)$ produce a new loop $R=\{R_0, R_1,\dots R_k\}$, where $R_i=(1-s)P_i+sQ_i$. Note that although $J_s(^0P)=L_s(J_0(^0P),J_1(^0P))$, in general, $^kJ_s(^0P)\neq L_s(^kJ_0(^0P), {}^kJ_1(^0P))$. Hence, the curves produced by iterations of J_s refinements are ***not linear combinations of the curves produced by iterations of 4-point and cubic B-spline schemes***. This observation explains why the limit curves produced by iterative J_s refinements may exhibit smoothness properties that are superior to those of J_0.

As the number k of refinements grows, ^kP converges to a limit curve $^*J_s(^0P)$, which we denote by $^*J_s$. We prove below that:

- for $-1.7 \le s < 0$ and $4 \le s \le 5.8$, $^*J_s$ is C^1,
- for $0 < s \le 1$ and $2.8 < s < 4$, $^*J_s$ is C^2,
- for $1 < s < 3/2$ and $3/2 < s \le 2.8$, $^*J_s$ is C^3, and
- for $s = 3/2$, $^*J_s$ is C^4.

To establish the continuity of $^*J_s$ for different values of s, we first consider the necessary conditions for continuity due to Reif [Rei95]. Given the subdivision matrix for J_s, if the subdivision scheme produces curves that are C^m, then the eigenvalues of its subdivision matrix are of the form 1, (1/2), (1/4), …, $(1/2)^m$, λ, … where $\lambda<(1/2)^m$. The eigenvalues of the subdivision matrix for the J_s subdivision are 1, (1/2), (1/4), (1/8), (2–s)/8, (s–1)/16, (s–1)/16, 0, 0. It is easy to verify that J_s subdivision satisfies the necessary conditions for C^1 continuity when $-2<s<6$, C^2 continuity when $0<s<4$, C^3 continuity when $1<s<3$, and C^4 continuity when s=3/2. Notice that these conditions are only necessary, they are not sufficient.

To determine <u>sufficient</u> conditions on the subdivision scheme, we use the Laurent polynomial of the subdivision scheme given by

$$S(z) = (s-1)/16 + s/8\, z + (9-s)/16\, z^2 + (1-s/4)\, z^3 + (9-s)/16\, z^4 + s/8\, z^5 + (s-1)/16\, z^6, \qquad (1)$$

which encodes the columns of the infinite subdivision matrix in a compact form.

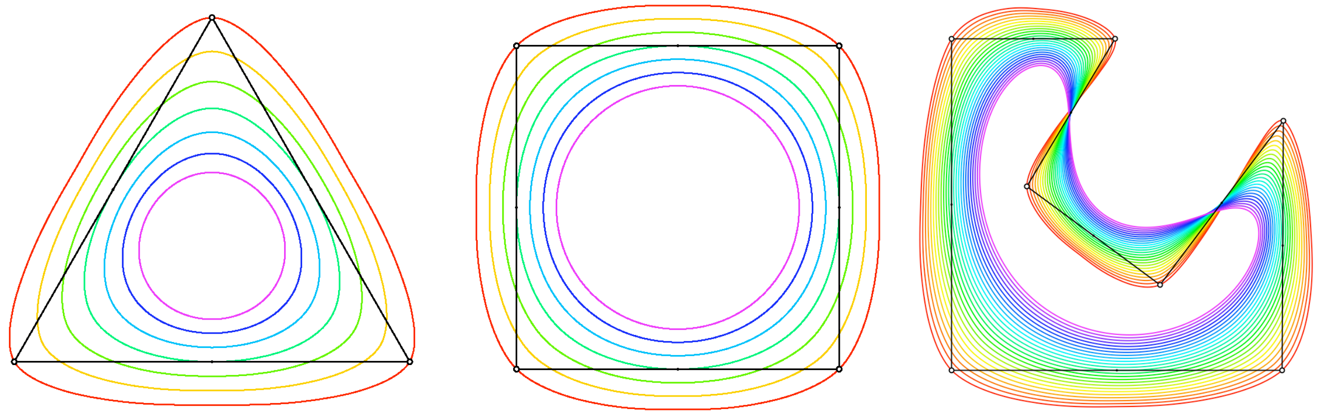
Fig. 2: ${}^5J_0$, ${}^5J_{2/8}$, ${}^5J_{4/8}$, ${}^5J_{6/8}$, ${}^5J_{8/8}$, ${}^5J_{10/8}$, ${}^5J_{12/8}$, from outer to inner (left & center). ${}^*J_0$ is the C^1 four-point curve. ${}^*J_{4/8}$ is the C^2 curve of [Ros04]. ${}^*J_{8/8}$ is the C^2 uniform cubic B-spline curve. ${}^*J_{12/8}$ is the C^4 quintic uniform B-spline curve. A denser sampling of ${}^*J_s$ curves is also shown (right).

The subdivision scheme generates C^m curves if the infinity norm of the kth power of the subdivision matrix for the mth divided differences is less than 1 for some k [WW02, p. 77]. The columns of this divided difference subdivision matrix are given by $(2^m/(1+z)^{m+1})S(z)$. We can numerically check what range of s satisfies these bounds for different continuity levels. We have verified that J_s subdivision produces curves that are at least C^1 for $-1.7 \le s \le 5.8$, C^2 for $0<s<4$, C^3 for $1<s\le 2.8$ and C^4 for $s=3/2$. In fact, $s=3/2$ corresponds to uniform quintic b-spline subdivision—indeed their Laurent polynomials are identical. Although the sufficient bounds that we were able to verify numerically are slightly more restrictive than the proven necessary bounds, we strongly suspect that the true sufficient bounds extend to match the necessary bounds for continuity in the limit. We were not able to verify this conjecture because the numerical verification is exponential in k and difficult to compute for large values of k.

4. Relation between J_s curves and uniform B-splines

Lane and Riesenfeld [LR80] showed that uniform B-spline curves B_d of degree d have a simple subdivision rule: *First double the vertices by inserting new vertices as mid-edge points. Then, take the dual (replace vertices by mid-edge points) d–1 times.* Stam points out [Sta01] that a pair of dual operations may be combined into a single smoothing step where each vertex moves towards the average of its two neighboring vertices. These subdivision rules create curves that are C^{d-1}.

J_s exactly reproduces the odd degree B-splines B_3 and B_5 for s=1 and s=3/2. It does not reproduce even degree B-splines exactly though. However, we can compute the optimal values of the parameter s in a *data-independent* manner to best match the basis functions created by B_2 and B_4 subdivisions. To perform this optimization in a data-independent manner, we minimize the difference between the basis function values on a dense uniform grid (we use a grid of 1531 samples). The optimal parameter s will depend on what norm is used to measure the distance between the values. For example, we prove that the disparity between ${}^*J_{.689}$ and B_2 never exceeds a fifth of the maximum edge length of ${}^0P$, but also notice that in practice (Fig. 3) ${}^*J_{.689}$ is a much closer *approximation* of B_2 than suggested by this conservative upper bound.

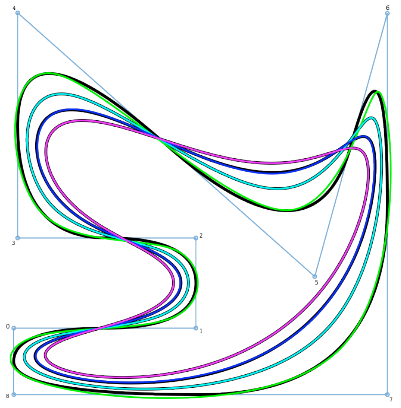

Fig. 3: From outer to inner: ${}^*J_{.689}$ approximates B_2, ${}^*J_1$ is B_3, ${}^*J_{1.27}$ approximates B_4, and ${}^*J_{1.5}$ is B_5. To facilitate comparison, the J curves are drawn on top of their thicker B counterparts.

L^2 is a popular norm because the resulting optimization problem is polynomial and easy to solve. However, the L^2 norm gives more weight to areas with large discrepancy and hence may not correspond to how we perceive average closeness of shapes.

Since we perform optimization in a data-independent manner, we only need to optimize the parameter s once and can afford to solve a more expensive minimization problem using other norms. Hence, we have considered the L^1 and L^∞ norms. The L^1 norm measures total variation or the integral of the absolute value of the difference between the two shapes. We believe that it matches our intuitive perception of similarity between shapes as it attempts to minimize the *average discrepancy* between the curves. The L_∞ norm provides a strict error bound in that this norm measures the *largest deviation* of the curve but provides no measure of how similar the curves are away from the point where they are furthest apart. For quartic B-splines, we obtain an optimal value of s=1.27 for both norms. When optimizing the J_s subdivision to match the quadratic B-spline subdivision, the L^1 and L^∞ norms produce very different values s=.689 and s=.639 respectively. We advocate the L^1 norm, since it produces what we perceive as a closer match.

Notice that a quadratic B-spline curve is C^1, whereas ${}^*J_{.689}$, which approximates it, is C^2. ${}^*J_1$ is a cubic B-spline curve B_3. ${}^*J_{1.27}$ is a C^3 curve that closely approximates the quartic B-spline curve B_4 (the curves appear identical in Fig. 3). Finally, ${}^*J_{1.5}$ is a C^4 quintic B-spline curve (Fig. 3).

5. Retrofitting J_s curves

As the value of s increases towards 1.5, the smoothness of our curve increases, but the limit curve drifts farther away from the vertices of the *original control polygon* C. To remedy this problem, we can perform a simple optimization to obtain a polygon ${}^0P$ for which the limit curve ${}^*P$ exactly interpolates the vertices of C. The limit mask for a subdivision scheme is given by the dominant left eigenvector of the subdivision matrix [HKD93]. For J_s subdivision, the limit mask has the closed-form {(s–1)s, 2s(8–s), 72+2(s–9)s, 2s(8–s), (s–1)s}/(12(6+s)) for arbitrary parameter values s.

We can solve a global system of equations using a matrix whose rows contain shifts of the limit mask to find control points such that the limit curve exactly interpolates the vertices of the control polygon [WW02, p. 182], but this solution may be expensive to calculate for large numbers of control vertices.

As an alternative, we have developed a simple and effective iterative retrofitting scheme (Fig. 4), which converges rapidly to the solution of these equations. We initialize ${}^0P$ with the vertices of C. Then, for each vertex ${}^0P_j$, we compute its limit position ${}^*P_j$ using the limit mask provided above. We then adjust each vertex ${}^0P_j$ to ${}^0P_j + (C_j - {}^*P_j)$. We iterate this process until the difference between ${}^*P_j$ and C_j for all j falls below a desired threshold.

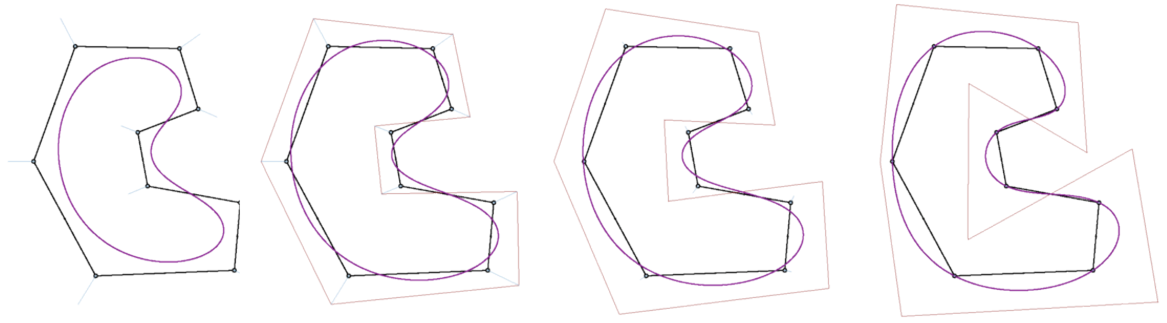

Fig. 4: We compute the vectors $C_j – {}^*P_j$ (left) and apply them (center-left) to adjust ${}^0P_j$. We repeat this process (center-right), quickly converging to a new control polygon (orange), which yields an interpolating curve (right).

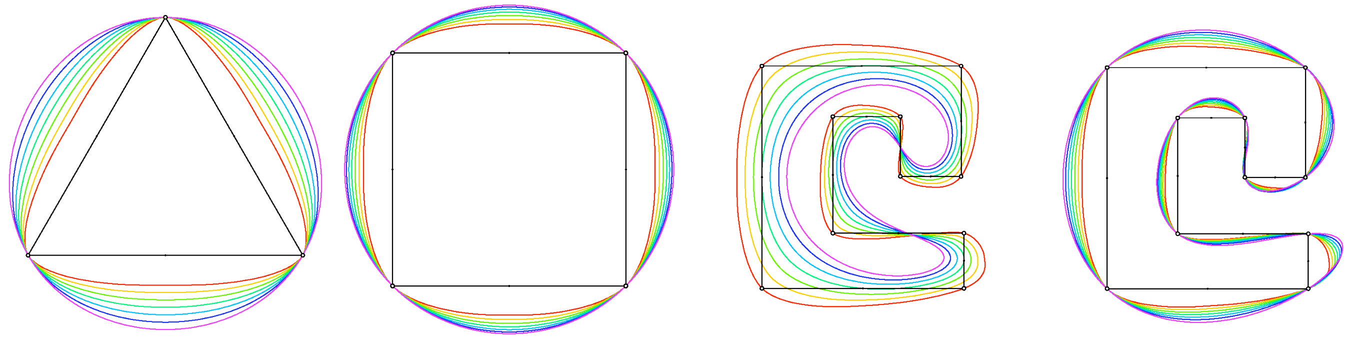

Fig. 5: Retrofitted versions of the refined triangle and square (left) of Fig. 3. The original (center-right) and retrofitted (right) J_s curves for another shape.

This retrofitting process may be applied to any J_s scheme (Fig. 5). Our experiment show that the iterative solver converges in realtime for a range of values of s that supports the needs of practical design situations, making the proposed retrofitting suitable for interactive editing. Unfortunately, we have no proof of convergence. Furthermore, this method fails to converge for values of s outside of this range. To investigate when these failure might occur, we compute the spectral radius (largest absolute eigenvalue) of the infinite matrix (I–L) where I is the identity matrix and L is a matrix whose rows contains shifts of the limit mask. Despite the fact that this matrix is infinite, we can use techniques from block-circulant matrices to write down the infinite set of eigenvalues and bound their norm. If the spectral radius of the matrix (I–L) is greater than or equal to 1, then this iterative method for interpolating the vertices of the control polygon will fail. The J_s subdivision violates this convergence criterion when $s \leq -0.86$ and when $2 \leq s$. Therefore, the iterative retrofitting will diverge for these values of s. The iterative technique converged in realtime when $-0.75 < s < 1.5$ and the convergence speed deteriorates as s approaches 2.

6. Vertex-interpolation through a model-independent mixed schemes

In the retrofitting schemes discussed above, the effect of tweaking a control vertex of C is usually most significant in the nearby portion of ${}^*P$, but may affect the whole curve because the global nature of the retrofitting solution leads to a loss of local control, possibly making this approach impractical for local shape editing in some applications. Therefore, we propose an alternative that makes it possible to *retain local control* and obtain refined curves that nearly interpolate the vertices of C. Following [Lev03], we use our J-spline generalization of the J_s refinement and precede the series of J_s steps by a single *anticipation step* $J_{a,b}$ (as defined in Section 1) with optimized value of parameters a and b. The value of parameter a defines the position of each even vertex of ${}^1P$ as a linear combination of the corresponding vertex of ${}^0P$ and the cubic B-spline subdivision tucked version of that original vertex. The value of parameter b similarly defines the position of each odd vertex of ${}^1P$ as a linear combination of the corresponding mid-edge point of ${}^0P$ and the four-point subdivision tucked version of that point. We use the subdivision sequence $\{J_{a,b}, {}^*J_s\}$. To ensure local control, we solve for the optimal values of a and b in a *model-independent* manner. We compute a limit mask for the original vertices produced by the subdivision sequence $\{J_{a,b}, {}^*J_s\}$ and optimize the parameters a, b, and s to minimize the worst-case distance between the original and limit positions of these control vertices.

The coefficients of the limit mask can be extracted easily by multiplying the corresponding Laurent polynomials. For example, assume that we are building a C^4 curve that is as close as possible to the vertices of the control polygon. Then, combining equation 1 with the Laurent polynomial representing the limit mask for s=3/2, ($L(z) = 1/120\, z + 13/60\, z^2 + 11/20\, z^3 + 13/60\, z^4 + 1/120\, z^5$), yields a polynomial whose even degree coefficients encode the corresponding limit point. Minimizing the difference between this mask and the identity mask in the infinity norm provides a solution independent of a particular control polygon. For example, this optimization yields a single anticipation step J_r, with $r = -33/26$, which, when followed by a series of $J_{1.5}$ steps, produces to a C^4 quintic B-spline curve that nearly interpolates the original vertices (Fig. 6). A different anticipation J-spline step $J_{a,b}$ with $a= -7/4$ and $b= 59/52$, yields exact interpolation of the control vertices, but the final shape is slightly flattened along the edges (Fig. 7).

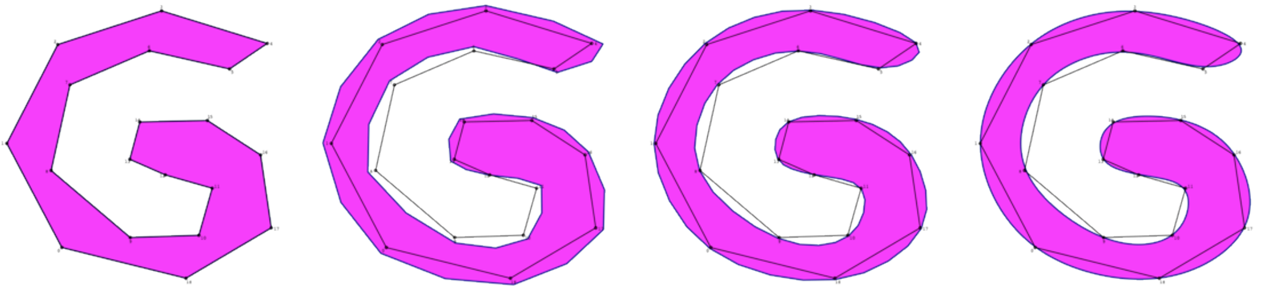

Fig. 6: From left to right: Original. After an "anticipation" step of J_r with $r = -33/26$. After a subsequent step $J_{1.5}$. Subsequent iterations of $J_{1.5}$ converge to a C^4 curve close to the original vertices (right).

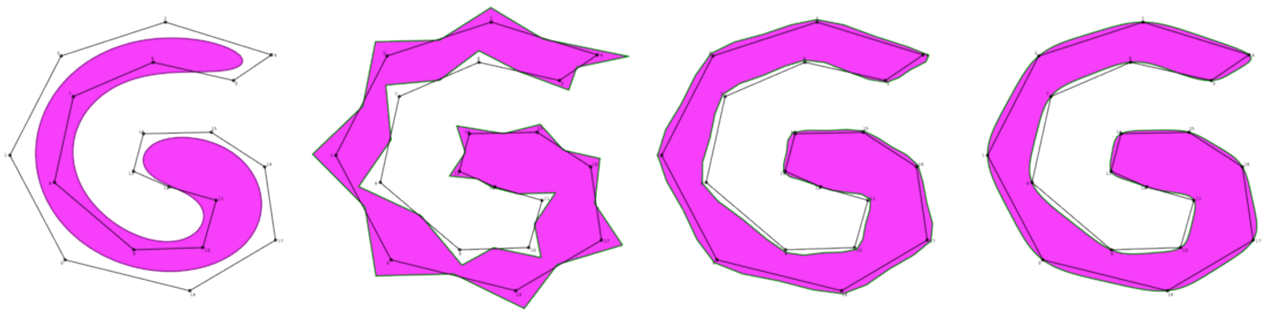

Fig. 7: From left: ${}^*J_{12/8}$, $J_{a,b}$ with $a= -7/4$ and $b= 59/52$, followed by $J_{1.5}$, followed by several additional $J_{1.5}$.

7. Mid-edge interpolation

While the previous section concentrated on interpolating the original control vertices, we interpolating mid-edge points is easier and tends to produces a more pleasant shape. This solution makes sense when the control polygon is designed *around* the desired curve and when its vertices are not computed or measured samples *on* the desired shape. The optimization problem is exactly the same as in Section 6. However, instead of applying the limit mask to the vertices corresponding to control vertices, we apply the limit mask to the vertices inserted on the edges during our anticipation subdivision step. To make the limit curve interpolate the mid-edge points of the control polygon, we minimize the difference between this limit mask and the mid-edge point mask (½, ½) in the infinity norm.

We have computed optimized values of parameters for several approaches to mid-edge interpolation (Fig. 8): the C^1 quadratic B-spline B_2 curve; the C^2 curve produced using a J_r step with r=2/3 followed by a series of J_1 steps, the C^2 $^*J_s$ with s=0.751, and the C^4 curve produced using a J_r step, with r = 29/59, followed by a series of $J_{1.5}$ steps. Note that, even though all four limit curves are similar, the first two schemes interpolate the mid-edge points *exactly*, while the other two only pass very close to them.

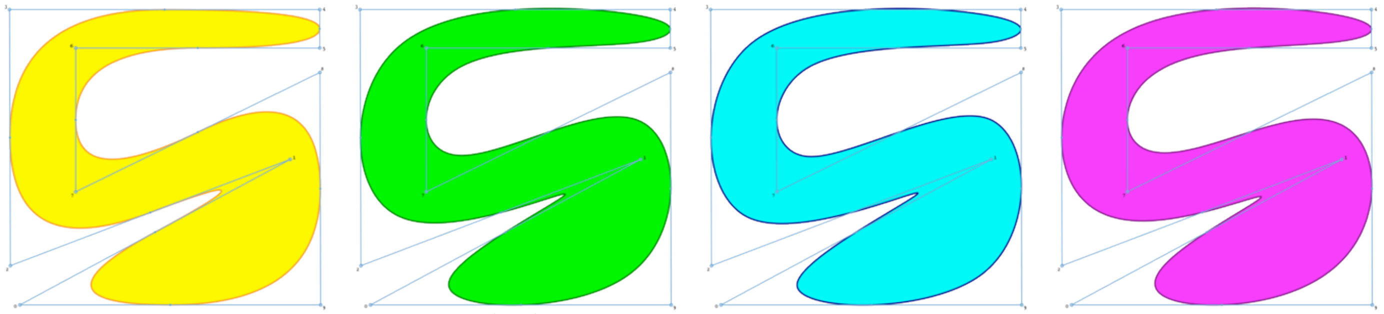

Fig. 8: From left-to-right: B_2, $J_{2/3}$ followed by $^*J_1$, $^*J_{0.751}$, $J_{29/59}$ followed by $^*J_{1.5}$.

8. Area preservation in 2D

For each control polygon, the a, b, and s parameters of our $\{J_{a,b}$, $^*J_s\}$ combination or the r and s parameters of our $\{J_r$, $^*J_s\}$ combination may be adjusted in a shape-dependent manner through numerical iteration to ensure that the refined curve has the same area as ^0P. Adjusting s in $*J_s$ will typically produce a C^2 curve though this level of smoothness is not guaranteed (Fig. 9a). Adjusting r, or a and b, while keeping s=1.5 will produce a C^4 curve (Fig. 9b). However, in all cases, when a control vertex is moved a new optimization must be performed to recomputed s and change the whole curve. Hence, local control is lost.

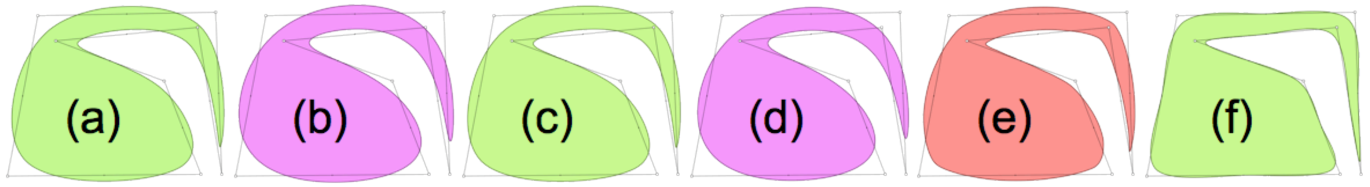

Fig. 9: Model-dependent optimizations, $^*J_{.476}$ (a) and $\{J_{.050}$, $^*J_{1.5}\}$ (b), both yield E=0, where E is the relative area error. Model-independent optimizations produce small errors: $^*J_{.46}$ yields E=0.79% (c), $\{J_{-0.03}$, $^*J_{1.5}\}$ yields E=3.4% (d), $\{J_{0.84}$, $^*J_{-0.53}\}$ yields E=0.035% (e). $\{J_{-0.0053,1.03}$, $^*J_{0.47}\}$ preserves the area exactly (f).

To avoid this shape-dependent optimization and *preserve local control*, we perform an optimization over the free parameters in our subdivision scheme in a *model-independent* fashion to preserve the area of the control polygon. [WW02, p. 165] showed how to compute exact inner products of non-polynomial subdivision schemes and how to use it to compute the exact area enclosed by the limit curve of a subdivision scheme. To minimize the difference in the area of the limit curve from the control polygon, we compute the area mask corresponding to linear subdivision and our parameterized subdivision scheme. The optimal parameters are given by minimizing the difference between these two masks in the infinity norm.

To reduce area change without an anticipation step, our optimization suggest to use $^*J_s$ with s = 0.46 (Fig. 9c). If a C^4 curve is desired, we recommend { J_r , $^*J_{1.5}$ } with an optimized parameter value r = –0.030 (Fig. 9d). If high levels of smoothness are not important, one may further reduce area change by using { $J_{0.84}$, $^*J_{-0.53}$ }, which yields a C^1 curve that may have visible kinks (Fig. 9e). Note that these solutions are independent of the particular control polygon, but do not guarantee that area will be preserved exactly. If exact area preservation is required with a model-independent solution, we suggest the mixed J-spline scheme { $J_{-0.0053,1.03}$, $^*J_{0.47}$ }, which produces a C^2 curve noticeably flattened along the edges of the control polygon (Fig. 9f).

9. Open curves

The extension of the J_s refinements to open curves is straightforward: we add two new control points $^0P_{-1}$, $^0P_{-2}$ and ^0P_n, $^0P_{n+1}$ to each end of the curve and proceed as described above, although we do not generate the *spans* that are produced through subdivision corresponding to the 5 edges of the control polyloop that are incident upon these added vertices. This way, we ensure two things: (1) the refinement rules apply only to vertices with a sufficient number of neighbor vertices to multiply by the non-zero coefficients in the subdivision rules and (2) the added vertices at one end of the control polygon do not influence the other end (Fig. 10 a, b).

Often, one may want the final curve to start at the first control vertex ${}^0P_0$ and to end at ${}^0P_{n-1}$. To ensure that the subdivided curve goes through ${}^0P_0$, we set ${}^0P_{-1} = 2{}^0P_0 - {}^0P_1$ and ${}^0P_{-2} = 2{}^0P_0 - {}^0P_2$. To ensure that it goes through ${}^0P_{n-1}$, we set ${}^0P_n = 2{}^0P_{n-1} - {}^0P_{n-2}$ and ${}^0P_{n+1} = 2{}^0P_{n-1} - {}^0P_{n-3}$.

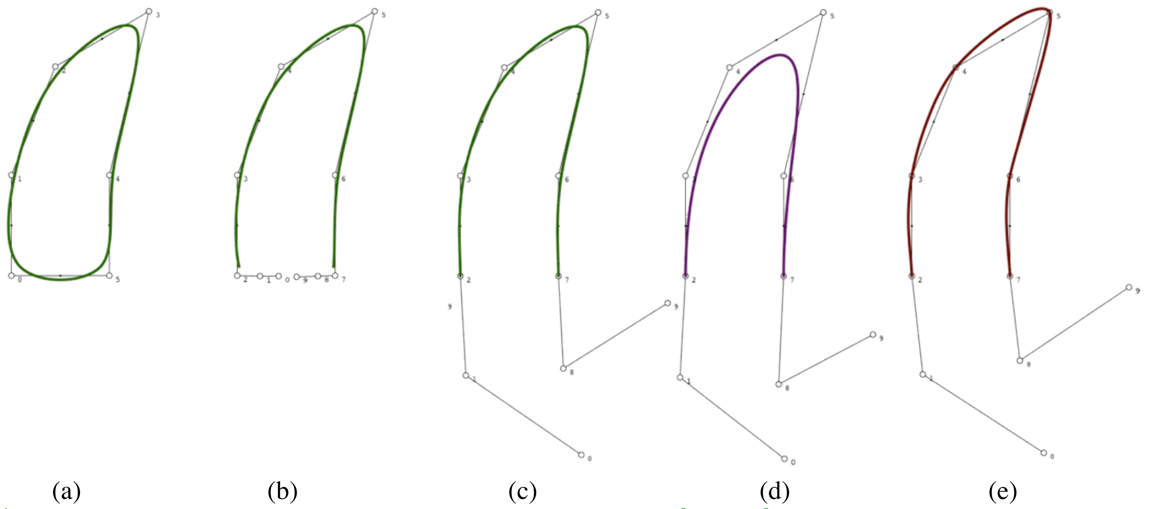

Fig. 10: (a) ${}^*J_{0.5}$ for a closed loop; (b) Same with 4 vertices added between ${}^0P_5$ and ${}^0P_0$ and 5 spans removed; (c) Added vertices adjusted to interpolate ${}^0P_5$ and ${}^0P_0$ in position and direction; (d) Same for ${}^*J_1$; (e) And for ${}^*J_0$. In each curve, the vertices are labeled with numbers starting at zero. For example, in (c), (d), and (e), the vertex labeled 0 is ${}^0P_{-2}$.

In addition to interpolating the ends of the control curve, one may want to force the limit curve to be tangent to the control polygon at these ends, that is to interpolate ${}^0P_0$ and ${}^0P_{n-1}$ with a tangents in the direction of ${}^0P_1 - {}^0P_0$ and ${}^0P_{n-2} - {}^0P_{n-1}$ respectively. Using the limit mask from Section 5 and the tangent mask $\{1-s, 2(s-4), 0, -2(s-4), -(1-s)\}/12$ derived from the left eigenvector of the subdivision matrix corresponding to ½, we solve a simple set of equations for these additional control points to enforce the specified properties. The solution adds end points ${}^0P_{-1} = (9-s)/4\ {}^0P_0 + (s-3)/2\ {}^0P_1 + (1-s)/4\ {}^0P_2$ and ${}^0P_{-2} = (12-s)/2\ {}^0P_0 + (s-8)\ {}^0P_1 + (6-s)/2\ {}^0P_2$ to the beginning of the curve. The masks for the opposite end of the curve are mirror images of these (Fig. 10 c, d, e).

10. Extensions to surfaces

Although, for clarity, we used 2D curves for illustration, subdivision can operate on control vertices in arbitrary dimension. These extra dimensions may be used to define space-curves in 3D or to add local properties to points on the curve such as color or thickness. This curve subdivision technique also trivially extends to quadrilateral, tensor product surfaces. For example, to tessellate such a surface, we subdivide edges of polygons along parallel *u*-parameter lines. Next, we use these control points to form curves in the orthogonal *v*-parameter direction and subdivide these curves to form one round of subdivision (Fig. 11). The limit surfaces produced by this process have the same smoothness properties of the original J_s subdivision scheme for curves.

To produce surfaces with borders, we add border vertices in each parametric direction, as described in Section 9 (Fig. 12).

For surfaces with extraordinary vertices (valence not equal to four), we need a different set of subdivision rules. Catmull-Clark subdivision [CC78] is a generalization of cubic b-spline subdivision (s=1) to quadrilateral surfaces with extraordinary vertices. Kobbelt also created an interpolatory quadrilateral subdivision scheme [Kob96] by generalizing the four-point curve subdivision rules (s=0) to surfaces. A natural generalization of our method would be to build a surface subdivision scheme that is a combination of these two methods $(1-s)S_c + sS_k$ where S_c is Catmull-Clark subdivision and S_k is Kobbelt's quad subdivision scheme. In ordinary regions of the surfaces (vertices with valence 4), these subdivision schemes reduce to the tensor-product of the corresponding curve subdivision schemes and our combination will have the same smoothness properties as our curve subdivision scheme for the particular value of s used. However, at extraordinary vertices, analyzing the smoothness of our combined subdivision method is more involved.

A surface subdivision scheme produces smooth surfaces at extraordinary vertices if several conditions hold, which are due to Reif [Rei95]. First, the subdivision matrix must have eigenvalues of the form 1, λ_1, λ_2, λ_i, … where $\lambda_2 = \lambda_1$ and $\lambda_i < \lambda_2$ for i>2 when sorted from highest to lowest magnitude. Furthermore, the characteristic map formed from the sub-dominant left eigenvectors of the subdivision matrix must be regular and injective for all valences. Unfortunately, it is impossible to symbolically extract the eigenvectors/values for this subdivision scheme because the result requires symbolic roots of high degree polynomials. We can, however, compute the eigenvalues and eigenvectors numerically for various valences and values of s. Our numerical experiments indicate that the surface is not always smooth at the extraordinary vertex especially at high valence vertices with values of s near -2 or 6. However, we conjecture that the surfaces are smooth for $-¼ < s < 3/2$ up to at least valence 30 (we computed all eigenvalues and eigenvectors for valences 3 through 30 for ¼ increments of s). Fig. 13 shows some examples of the characteristic maps generated for s = –1/4, ½, and 3/2 and for valences 3, 5, 6, 7, 8, and 20.

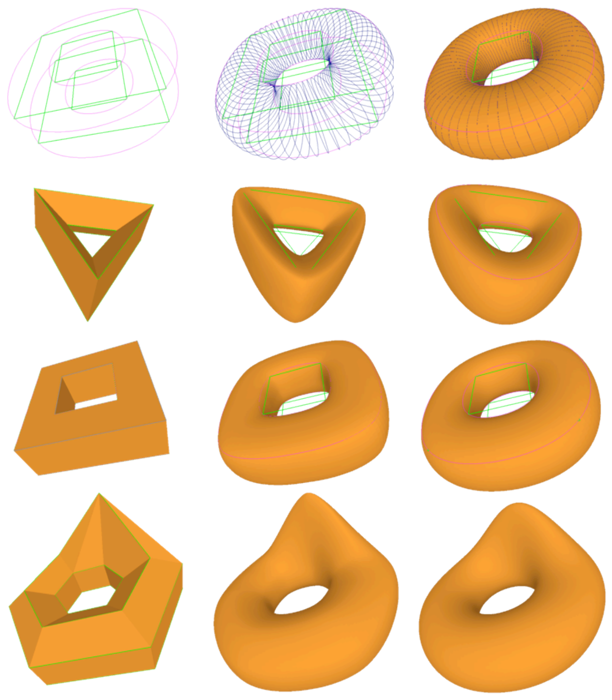

Fig. 11: Top: A torus-like surface defined by 4 control curves with 4 control vertices each. The refined curves are shown left. Each set of 4 vertices, one on each curve, controls a transversal curve (center). Triangle strips formed by pairs of consecutive transversal curves are shaded (right). Below: The control polyhedron (left), $*J_1$ (center), and $J_r\ {}^5J_{12/8}$ (right) are shown for three control meshes. The rendering was performed using a footprint of respectively 5, 6, and 8 rings of 5 points each.

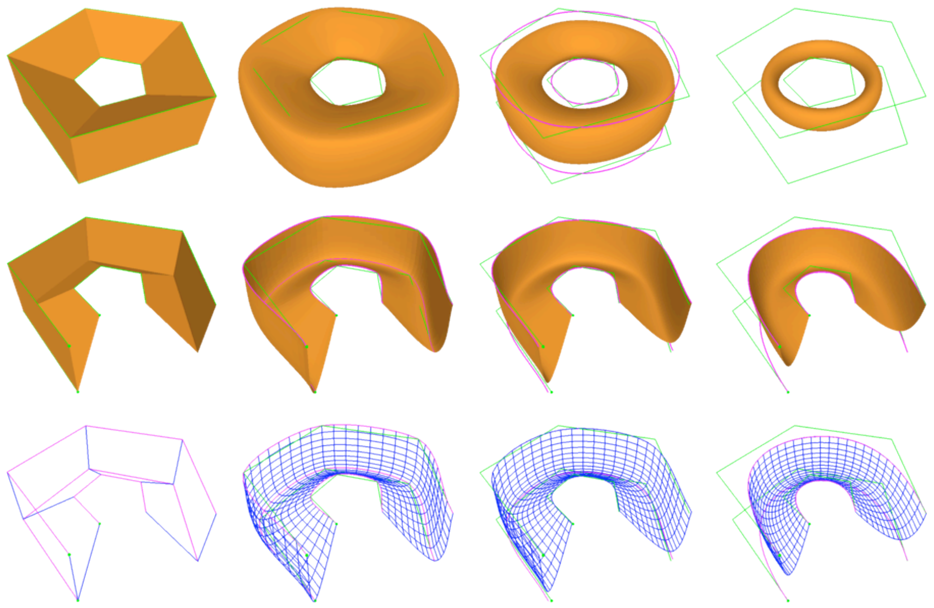

Fig. 12: A surface defined by 3 curves of 5 vertices each. From left to right: control polyhedron, four-point, compromise [Ros04], and the quintic B-spline. Closed surfaces (top row), open surfaces with one border (middle), and quads drawn (bottom).

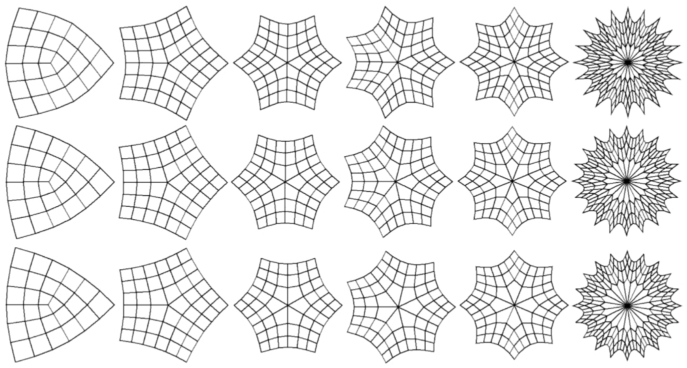

Fig. 13: The characteristic maps for our combined surface subdivision scheme at valences 3, 5, 6, 7, 8 and 20. The parameter s is -1/4 for the top row, ½ for the middle row and 3/2 for the last row.

Fig. 14 shows an example surface containing extraordinary vertices of many different valences subdivided for various parameter values of s. This control mesh was originally designed to model the desired shape using the Catmull-Clark subdivision and, therefore, surfaces produced by other schemes look somewhat unnatural. However, the example is useful for comparing the output of the different subdivision schemes. Notice that, in each case, the surface is smooth, even at extraordinary vertices.

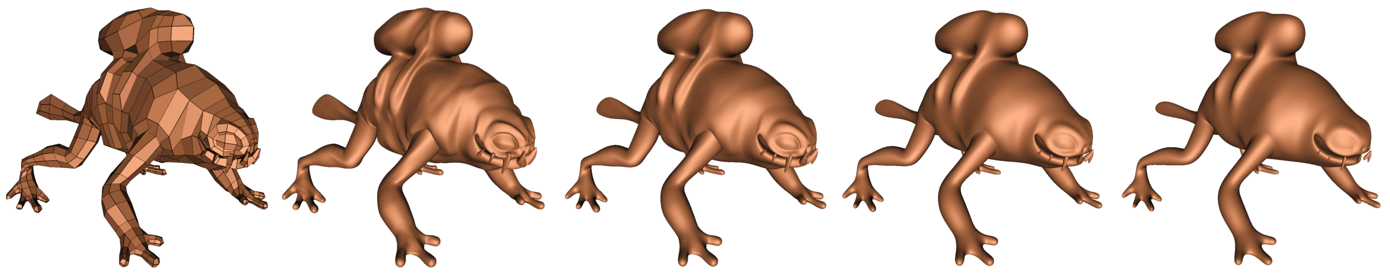

Fig. 14: From left to right: the control polygon for a complex shape with many extraordinary vertices, J_0 surface subdivision (Kobbelt's quad subdivision scheme), $J_{1/2}$ surface subdivision, J_1 surface subdivision (Catmull-Clark) and $J_{1.5}$ surface subdivision.

11. Popping reduction in multi-resolution rendering

When switching between consecutive levels of detail (LoDs) in multi-resolution rendering, a distracting popping artifact occurs. The effect may be diminished by using a *geomorph* that linearly blends over time between the results of applying $J_{0,1}$ and applying another J_s step to the previous level. As observed by Maillot and Stam [MS01], the need for a geomorph and the amount of shape disparity that it serves to mask may be considerably reduced by selecting a subdivision scheme that minimizes the discrepancy between consecutive levels of subdivision (Fig. 15 and 16). Maillot and Stam show this benefit using s=0.3, but did not attempt to find an optimal value of s. Through experimentation, we have found that $^*J_{0.375}$ works well. Beyond this empirical evaluation, we compute, in a model-independent manner, as explained below, the value of s that minimizes the difference between one level of subdivision and a linear subdivision $J_{0,1}$. Notice, however, that the optimal value will change with the level of refinement.

Let L(z) be the Laurent polynomial for linear subdivision. For the first level of subdivision, we minimize the magnitude of the coefficients of the polynomial L(z)–S(z) in the infinity norm where S(z) comes from Equation 1 producing an optimal value of s=0.2. However, if we subdivide again, the optimal value will change. Minimizing the magnitude coefficients of the coefficients of the polynomial $(L(z) - S(z))\, S(z^2)$ yields s=0.366. Repeating this process for further subdivision levels yields a non-stationary subdivision scheme where the rules change at each level of subdivision. The optimized values of s for the first nine levels of subdivision are 0.2, 0.366, 0.421, 0.449, 0.465, 0.475, 0.482, 0.487, and 0.491. Since we do not have a closed-form expression for the values of s as we subdivide, it is very difficult to analyze the smoothness of the curve generated by this non-stationary subdivision scheme. However, given that all values of s fall within the C^2 range of the stationary subdivision scheme, we strongly suspect that the smoothness of the curve is at least C^2.

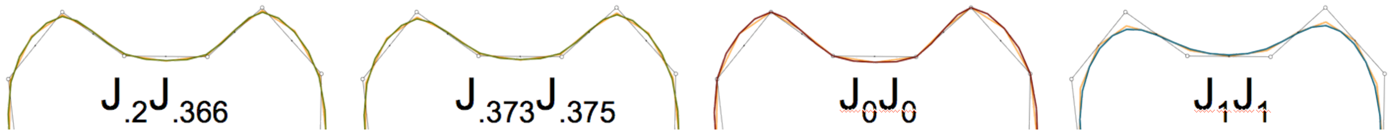

Fig. 15: The current (darker) and previous (lighter) levels of a polygon refinement are shown for various values of r and s. The disparity between consecutive levels is smaller (left) for the suggested parameter values.

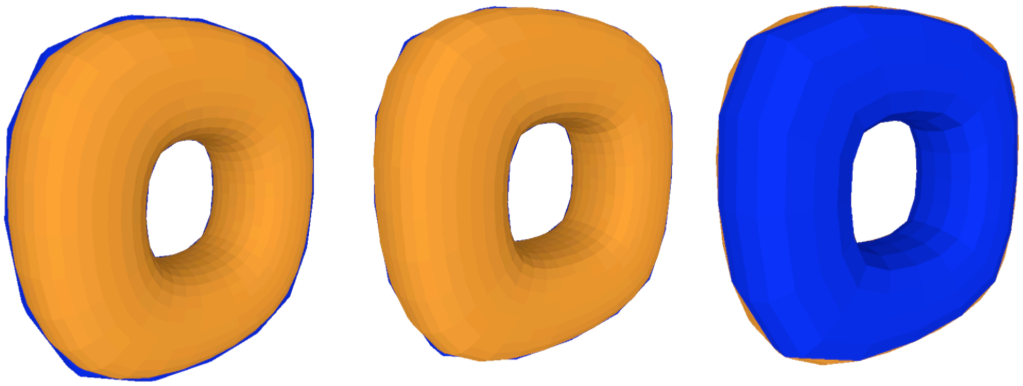

Fig. 16: The images of the current (lighter) and previous (darker) refinements of a surface are superimposed to show silhouette disparities for J_0 (left with the lighter image painted on top), $J_{3/8}$ (center with lighter on top), and J_1 (right with darker on top).

12. Conclusions

In this paper, we analyze the J_s family of polygon subdivision rules invented by Maillot and Stam that blend the 4-point J_0 and the uniform cubic B-spline J_1 schemes. We prove that the limit curve $^*J_s$ is at least C^1 when $-1.7 \le s \le 5.8$, C^2 when $0<s<4$, C^3 when $1<s\le 2.8$ and C^4 when $s=3/2$. We point out that $J_{1.5}$ is the quintic uniform B-spline subdivision. We propose $^*J_{.69}$ as a C^2 approximation of quadratic B-splines. We provide several solutions for producing limit curves that interpolate or closely approximate the control vertices. To provide a model-independent solution with local control, we use a mixed scheme $\{J_{a,b}, ^*J_s\}$, where the J_s iterations are preceded by an anticipation step $J_{a,b}$ of the J-spline scheme introduced here as a generalization of Maillot and Stam subdivision. Through model-independent optimization, we computed two subdivision schemes that generate C^4 curves: $\{J_{-7/4,\ 59/52}, ^*J_{1.5}\}$ which interpolates the vertices exactly and $\{J_{-33/26,\ -33/26}, ^*J_{1.5}\}$ which approximates them very closely and yields more rounded and hence aesthetically preferable results. We show that one can easily produce C^4 curves that have the same area as their control polygon, either by iteratively adjusting the parameter of the anticipation step, or by using $\{J_{-0.03}, ^*J_{1.5}\}$, which preserves local control. We explain how to subdivide open curves ensuring that the limit curve interpolates the ends of the control polygon in position and direction. By blending the Catmull-Clark and the Kobbelt schemes, we provide an extension of this approach to quad-mesh surfaces with arbitrary connectivity and analyze their smoothness. Finally, we propose $*J_{0.375}$ or a specific series of s-parameter values (a different one for each subdivision level) for minimizing the disparity between consecutive refinements. This result is useful when it is desired to produce a limit shape that is close to the control shape (in the Hausdorff distance sense) or to reduce the popping between consecutive levels of subdivision during multi-resolution rendering.

13. Acknowledgement

We thank Malcolm Sabin for suggesting that the compromise advocated in [Ros04] may be C^2 and inspiring these developments. We also thank the reviewers for suggesting several improvements to the paper.

14. Bibliography

[BB83] B. Barsky and J. Beatty, "Local control of bias and tension in beta-splines," In SIGGRAPH '83: Proceedings of the 10th annual conference on computer graphics and interactive techniques, ACM Press, 193-218, 1983.

[CC78] E. Catmull and J. Clark, "Recursively generated B-spline surfaces on arbitrary topological meshes," Computer Aided Design, 10:350-355, 1978.

[Cha74] G. Chaikin, "An algorithm for high speed curve generation," Computer Graphics and Image Processing, 3:346-349, 1974.

[Cli74] A. Cline, "Scalar- and planar-valued curve fitting using splines under tension," Commun. ACM, 17:218-220, 1974.

[DD89] G. Deslauriers and S. Dubuc, "Symmetric iterative interpolation processes," Constructive Approximation, 5:49–68, 1989.

[DFH] N. Dyn, M. Floater, K. Hormann, "A C^2 four-point subdivision scheme with fourth order accuracy and its extensions ," In Mathematical Methods for Curves and Surfaces: Tromso 2004, Modern Methods in Mathematics, pp. 145–156, 2005.

[HKD93] M. Halstead, M. Kass, T. DeRose, "Efficient, Fair Interpolation using Catmull-Clark Surfaces," Computer Graphics, 27:35-44, 1993.

[Kob96] L. Kobbelt, "Interpolatory subdivision on open quadrilateral nets with arbitrary topology". Comp. Graph. Forum 15(3): 409–420, 1996.

[Lev03] A. Levin, "Polynomial Generation and Quasi-Interpolation in stationary non-uniform subdivision," Computer-Aided Geometric Design, 20(1):41-60, 2003.

[LLS01] N. Litke, A. Levin, P. Schroder, "Fitting subdivision surfaces," In VIS '01, 319-324, 2001.

[LR80] J. Lane and R. Riesenfeld. "A theoretical development for the computer generation and display of piecewise polynomial surfaces," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2:25-46, 1980.

[MK04] M. Marinov and L. Kobbelt. "Optimization techniques for approximation with subdivision surfaces," In ACM Symp. Solid Modeling and Applications, 113-122, 2004.

[MS01] J. Maillot and J. Stam. "A unified subdivision scheme for polygonal modeling", Eurographics, vol 20, 2001.

[NDG87] N. Dyn, J. A. Gregory, D. Levin, "A four-point interpolatory subdivision scheme for curve design". CAD, 4:257–268, 1987.

[PR08] A. Powell and J. Rossignac, "ScrewBender: Smoothing Piecewise Helical Motions". IEEE Computer Graphics and Applications, 28(1):56-63 Jan/Feb 2008.

[Rei95] U. Reif, "A unified approach to subdivision algorithms near extraordinary vertices," Computer Aided Geometric Design, 12:153-174, 1995.

[Ros04] J. Rossignac, "Education-driven research in CAD". Computer Aided Design 36, 14 (2004), 1461–1469.

[Sab02] M. Sabin, "Subdivision surfaces." In *The Handbook of Computer-Aided Geometric Design*, G. Farin, J. H., Kim M. S., (Eds.), chapter 12:309–327, 2002.

[Sta01] J. Stam, "On Subdivision Schemes Generalizing Uniform B-Spline Surfaces of Arbitrary Degree". Computer Aided Geometric Design, 18(5):383-396 2001.

[WW02] J. Warren and H. Weimer: "Subdivision methods for geometric design," San Francisco: Morgan Kaufmann, 2002.

[ZSS96] D. Zorin, P. Schroeder, W. Sweldens: "Interpolating subdivision for meshes with arbitrary topology," Computer Graphics, 30:189–192, 1996.