Reliable joining of surfaces for combined mesh-surface models

Di Jiang and Neil F. Stewart
Université de Montréal
{jiangdi, stewart}@iro.umontreal.ca

Keywords— **surface mesh, joining, graphical simulation, shape-interrogation models, normal-vector error**

Abstract— **Algorithms to join two mesh patches along an edge are of immediate practical interest in the context of higher-level operations on models of objects formed by such mesh patches. Such models are widely used in graphical visualization and simulation, shape interrogation, and other areas. Thus, there are now available methods to join two subdivision surfaces along a common edge curve, as well as methods to join mesh patches that approximate given trimmed-surface patches. The latter problem is studied in this paper.**

The auxiliary information available to the algorithm, in the context of surface joining, varies, depending upon circumstances. In particular, it may or may not be true that an explicit common edge curve, representing the boundary between the two patches to be joined, is available as part of the data. Even in the case, however, when maximal auxiliary information is available algorithms are not necessarily reliable. For example, methods that do not use normal-vector error criteria, to measure the discrepancy between the surface patch and the associated mesh patch, can produce poor results, due to large changes in the normal direction of a triangle near the mesh boundary. It is even possible to give examples where the triangles near the joined boundary are turned upside down by the joining process, so that computed meshes self-intersect. In this paper an algorithm is presented that uses a proxy for a normal-vector error criterion, and the Whitney extension theorem, to produce reliable algorithms. Examples are given, and an implementation is described.

I. Introduction

This paper is concerned with the problem of the reliable joining of surface meshes used in combined mesh-surface models. Such models are of interest for graphical visualization of solid objects, shape interrogation, computer-aided design, and vision [1], [2], [3], [4], [5], [6], [7]. The joining process is sometimes referred to as *sewing* [1]. The main novel aspect of the work is the use of normal-vector criteria, described below, to prevent folding of edges during the joining process.

A *mesh patch* is a surface made up of non-degenerate triangles lying in R^3. Algorithms to join two mesh patches along a common edge are of immediate practical interest in the context of higher-level operations on objects formed by such mesh patches. For example, methods have been given to join two subdivision patches along a common edge curve, specified in R^3. In particular, *combined subdivision surfaces* [8] were designed for this purpose, and *dynamic subdivision surfaces* [9] may be used to produce subdivision surfaces with hard edges along a given curve in space. Similarly, methods are available [1], [10, Sec.3.4] to join together mesh patches that approximate given trimmed-surface patches lying in R^3. It is the latter problem (surface-mesh joining) that is studied in this paper.

The auxiliary information available to the algorithm, in the context of surface-mesh joining, may vary. Mesh solids formed by a trimmed-surface model coupled with a triangular mesh are used in solid modeling [1], [3], [5] and in graphical simulation [1], [2], [4]. In the latter case, the mesh model may be carried along with the surface model, or computed adaptively during rendering, given the current camera position. The trimmed-surface model is illustrated in Figure 1.

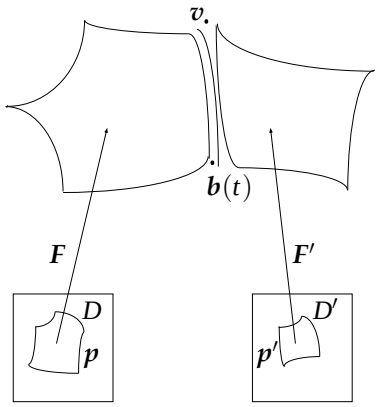

Fig. 1. Two adjoining trimmed patches in surface model, with boundary curve $\boldsymbol{b}(t), t \in [0,1]$.

The parametric domain D is delimited by a collection of p-curves (a typical p-curve is denoted here by $\boldsymbol{p}$), and the restriction of the mapping $\boldsymbol{F}$ to D defines the trimmed patch in R^3. In addition, explicit boundary information may also be present. Sometimes [3], [11], [12] this may take the form of explicit curves $\boldsymbol{b}(t)$ taking values in R^3, due to the convenience of having such explicit representations available. This curve is analogous to the common edge curve specified for combined subdivision surfaces. Alternatively, explicit boundary information in R^3 may be represented in other ways; for example, it may be represented approximately by scan conversion [1].

Even with an explicit boundary curve provided, joining algorithms are not necessarily reliable, and it is this fact that led to the development of the algorithms described below.

We present joining algorithms for both cases: when an explicit curve $\boldsymbol{b}(t)$ is provided, and when it is not. The algorithms described are based on the use (as a supplement to absolute error criteria) of normal-vector error criteria [13], [14], [15] for the discrepancy between the surface patch and the mesh-patch. A difficulty, with algorithms that do not use such criteria, is that they may cause large changes in the normal direction of a triangle near the joined boundary, which may in turn introduce undesired visual effects. In fact, it could even happen that triangles near the bound-

Proceedings 21st European Conference on Modelling and Simulation
Ivan Zelinka, Zuzana Oplatková, Alessandra Orsoni ©ECMS 2007
ISBN 978-0-9553018-2-7 / ISBN 978-0-9553018-3-4 (CD)

ary are turned upside down, so that computed meshes self-intersect. The nature of the difficulty is illustrated in Figure 2, in the case where joining moves mesh vertices on the basis of interpolation along a polygonal path that is not a

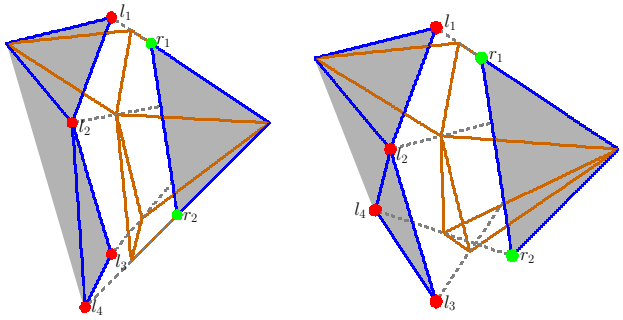

Fig. 2. Sewing based on midpoints of pairs of points interpolated along mesh edges.

straight line. In both illustrations, vertex l_1 is paired with vertex r_1, and vertex l_4 is paired with vertex r_2. The intervening joining vertices are obtained by joining the midpoints of pairs of points obtained by linear interpolation along the polylines l_1-l_2-l_3-l_4 and r_1-r_2. In the first illustration, this leads to a well-behaved triangulation, but in the second illustration, the position of the vertex l_4 is different: it is further towards the interior of the segment r_1-r_2, but still within the joining tolerance, relative to r_1-r_2. This phenomenon is called "folding" [1], and can result in a mesh triangle that has flipped, as in the second illustration of Figure 2. Such phenomena can be avoided by using normal-vector criteria, and in fact, if the normals of the triangles in the mesh-patch can be bounded, they can be used to rigorously exclude the possibility of extraneous intersections between neighboring mesh-patches [16], [17], [18]. In the context of graphical simulation, it is clearly of interest to do so.

The algorithms presented here use the Whitney extension theorem [19] to ensure that a proxy for the normal-vector error (defined below), and the absolute error, should not be any larger than the corresponding errors already present along the edges of the patch. Thus, in addition to avoiding the difficulty described in the previous paragraph, the procedure smooths the input mesh patches, in the sense just described of error minimization. The algorithms apply to the case of general trimmed patches, and we describe an implementation.

Whitney extension can be viewed as a way to perform transfinite interpolation between boundary curves. Amongst many other applications, it has been suggested for use as a meshing method in [5]. The algorithms below will adjust the vertices of the input mesh patch in a way that constrains them to lie in a transfinite interpolant defined by Whitney extension.

Numerical properties of one of our algorithms were discussed, in the special case of planar patches with straight-line boundaries, in [20].

Related areas of work include mesh simplification (finding a "... concise, yet geometrically faithful ... representation of a surface ..." [14, Sec. 1]), remeshing [14, Sec. 1.1] [15], [5], [21] and mesh fairing [22]. A good overview is given in [14]. Yet other work deals with computation of meshes over imperfect geometry [3], [23], and methods for mesh repair [2], [4], [24], [25].

Other work on meshing can be related to ours in another way, namely, by examining the metrics used to compare surfaces. The general concept of the absolute error in a mesh, relative to a given surface, is ubiquitous (see for example [26]). Again, the reference [14] gives a good overview. As already mentioned, other authors [13], [15] have introduced normal-vector criteria similar to ours. For example, in [15], although priority is given to other mesh-smoothness criteria, it is verified from time to time that a criterion, similar to the mean-square criterion discussed in Section II, is not above a certain threshold. Somewhat different criteria are used in other applications. For example, in the context of snakes on triangular meshes, [27] refers to bending-energy and curvature-distribution criteria that are different from but nonetheless similar to the height-field-slope criterion introduced below.

II. Error criteria to measure mesh-patch quality

One measure of the quality of a mesh patch M is the *absolute error*. Let $\boldsymbol{v}_1,\ldots,\boldsymbol{v}_n \in R^3$ be the vertices of M, and $T_1,\ldots,T_r$ its triangles, where $T_j =< \boldsymbol{v}_{i_1},\boldsymbol{v}_{i_2},\boldsymbol{v}_{i_3} >$, $1 \leq i_1,i_2,i_3 \leq n$. We assume that the Jacobian of the mapping $\boldsymbol{F}$ is of full rank, i.e., the rank is equal to 2. Let

$$\boldsymbol{n}(u,v) = (\boldsymbol{F}_u(u,v) \times \boldsymbol{F}_v(u,v))/\|\boldsymbol{F}_u(u,v) \times \boldsymbol{F}_v(u,v)\|$$

be the unit normal of the surface $\boldsymbol{F}$ at (u,v), and let the height $\eta(u,v) \in R$ be the scalar such that

$$\boldsymbol{M}(u,v) = \boldsymbol{F}(u,v) + \eta(u,v)\boldsymbol{n}(u,v) \in |M|,$$

where $|M|$ denotes the mesh viewed as a subset of R^3, if a unique such η exists. We suppose in fact that for all mesh patches considered, the mapping

$$\boldsymbol{M}^{-1} : |M| \mapsto [0,1]^2$$

is well-defined and injective. Thus, it is assumed that for any $\boldsymbol{m} \in |M|$,

$$|\eta| = dist(\boldsymbol{m},\boldsymbol{F}) \doteq \min_{\boldsymbol{y}\in\boldsymbol{F}} \|\boldsymbol{m}-\boldsymbol{y}\|$$

is uniquely defined, and furthermore, that the corresponding point (u,v) is well defined and lies in $[0,1]^2$. (It follows that the mapping $\boldsymbol{F}$ itself must be injective, at least on the part of the domain of interest. Note also that the symbol $\boldsymbol{F}$ has been used to denote both the mapping and the image of the mapping, which is a pointset.)

A possible definition of the absolute error in M is the supremum of $|\eta|$ over $I \subseteq [0,1]^2$, where I is the inverse image of $|M|$. Meshes are in practice close enough to $\boldsymbol{F}[D]$ that the assumption above, that $|\eta|$ is well defined, does not present a problem, provided $I \subseteq [0,1]^2$. (The mesh must be close relative to the local minimum normal curvature of $\boldsymbol{F}$.) On the other hand, there is a theoretical difficulty in simply defining the absolute error to be

$$\sup_{(u,v)\in I} |\eta| \qquad (1)$$

because there is nothing in this criterion to force full coverage of the surface patch by the mesh. For example, a degenerate mesh M consisting of a single vertex lying in $\boldsymbol{F}[D]$ would produce an error of zero. As observed in [14, Sec. 2.1], use of (1) amounts to using a one-sided version of the Hausdorff metric. In spite of the difficulty we have just described, this approach is often used in practice [14], and we will do so here. The coverage of practical meshes is usually quite good.

We also assume that D lies strictly inside $[0,1]^2$, *i.e.*, that the patch is trimmed on all sides. There is no theoretical problem in the opposite case, since normally [26] the mapping $\boldsymbol{F}$ is defined outside $[0,1]^2$. If, however, the inverse image of a point in $|M|$ lies outside $[0,1]^2$, there may be numerical difficulties in the calculation of η.

A second measure of the quality of M is the *normal-vector error*, defined here as the largest, over all triangles T_j, of the maximum slope (in absolute value) of the height field. Let I_j be the inverse image of T_j under $\boldsymbol{M}$, and let L_j be the smallest value of L for which η satisfies the Lipschitz condition

$$|\eta(\boldsymbol{p}_1) - \eta(\boldsymbol{p}_2)| \leq L \cdot \|\boldsymbol{p}_1 - \boldsymbol{p}_2\|$$

for all points $\boldsymbol{p}_1 = (u_1, v_1)$ and $\boldsymbol{p}_2 = (u_2, v_2)$ in I_j. Our second criterion is then $\max_j L_j$.

To relate this criterion to similar normal-vector measures introduced elsewhere [13], [14], [15], we note that

$$\sup_{(u,v)} \|\boldsymbol{n}(u,v) - \boldsymbol{n}_j\|$$

(where $\boldsymbol{n}_j$ is the unit normal of the triangle T_j, and the supremum is taken over I_j) is analogous to the mean-square norm [14, Sec. 2.3.1] [15] of $\boldsymbol{n}(u,v) - \boldsymbol{n}_j$, normalized to allow for the area of the region I_j:

$$\|\boldsymbol{n} - \boldsymbol{n}_j\|_2 \doteq \left[\frac{1}{Area(I_j)} \int_{I_j} \|\boldsymbol{n}(u,v) - \boldsymbol{n}_j\|^2 du dv\right]^{1/2}.$$

It is, however, a more strict criterion, since

$$\|\boldsymbol{n} - \boldsymbol{n}_j\|_2 \leq \sup_{(u,v)} \|\boldsymbol{n}(u,v) - \boldsymbol{n}_j\|.$$

On the other hand, $\sup_{(u,v)} \|\boldsymbol{n}(u,v) - \boldsymbol{n}_j\|$ and the criterion L_j, defined above, are equivalent metrics, a fact which follows from our assumptions about the Jacobian of $\boldsymbol{F}$, and the Implicit Function Theorem. This justifies the terminology "normal-vector error" for the maximum slope of the height field.

It was stated in Section I that our algorithms control only a proxy for the normal-vector error. This proxy is obtained as follows. First of all, the slope of η on I_j is replaced by the slope measured only between the three corner points of I_j. This process can increase the error in the case of long thin triangles, but the difficulty can be avoided by mesh-edge splitting. (The error estimates given below, in Section III-D, take account of the potential error introduced in this way, *i.e.*, it is not assumed that mesh-edge splitting has been used to reduce the error.) Secondly, in order to reduce computational cost, we estimate $\max_j L_j$ by using the Whitney theorem with the ordinary Euclidean norm of $\boldsymbol{p}_1 - \boldsymbol{p}_2$, over all of $I = \bigcup_{j=1}^{r} I_j$, which could in principle (see Section III-A) lead to the minimization not of $\max_j L_j$ but, rather, the minimization of a certain upper bound for $\max_j L_j$.

III. Joining algorithms

As mentioned in the introduction, joining algorithms that do not use normal-vector criteria may cause large changes in the normal direction of a triangle near the boundary. The nature of the difficulty was shown in the second illustration in Figure 2. Thus, even though the input mesh patches satisfy the assumptions of Section II, and have small height η along the edges of the two patches, folding may occur within (or approximately within) the curvilinear surface $\boldsymbol{F}$. In this section we present algorithms that avoid this problem, and which, at the same time, smooth the mesh. Both of these are of obvious importance in graphical simulation. An example will be given below, in Section IV, which shows the possible ill effects of folding.

We begin by giving a brief summary of Whitney extension, which is used in both of the algorithms presented. We then give an algorithm in the case when the boundary curve $\boldsymbol{b}(t)$ is provided as part of the input, and in a subsequent subsection, we deal with the opposite case, by constructing ourselves a boundary curve $\boldsymbol{b}(t)$ based on the input mesh patches. The algorithms adjust the mesh vertices to ensure that the proxy, mentioned above, for the normal-vector error, and the absolute error, should not be any larger than the errors already present along the edges of the input patch. In fact, they will not be any larger than those associated with the boundary curves $\boldsymbol{b}(t)$ bordering the mesh patch. This of course represents only *part* of the error present in the input data: the error in the edge of the input mesh patch itself could in principle be even larger (and this fact makes our bound even more attractive).

A. Whitney extension

As mentioned at the end of Section I, our algorithms adjust the vertices of mesh patches in a way that constrains them to lie in a transfinite interpolant defined by Whitney extension. This process is referred to as *reprojection* in the algorithm outlines given below. The reprojected mesh interpolates the curves $\boldsymbol{b}(t)$, and the assumption of injectivity of $\boldsymbol{M}^{-1}$, at the beginning of Section II, includes in particular the assumption that we can compute the height $\eta(u_0, v_0)$ corresponding to a given $\boldsymbol{b}(t_0) \in R^3$, where $(u_0, v_0) = \boldsymbol{M}^{-1}(\boldsymbol{b}(t_0))$. This is done, as for vertices in a given mesh patch, by computing $dist(\boldsymbol{b}(t_0), \boldsymbol{F})$. (As in Section II, the assumption requires that $\boldsymbol{b}(t_0)$ be close to $\boldsymbol{F}[D]$, relative to the local minimum normal curvature of $\boldsymbol{F}$.)

Now, suppose given a mesh patch M with m edges, and corresponding boundary curves $\boldsymbol{b}^k(t), k = 0, \ldots, m-1, t \in [0,1]$. Let $\varepsilon(\boldsymbol{p})$ be the height $\eta(\boldsymbol{M}^{-1}(\boldsymbol{b}^k(t)))$ defined for a point $\boldsymbol{p} \in \partial R$, the inverse image of $\{\boldsymbol{b}^k(t) : k = 0, \ldots, m-1, t \in [0,1]\}$. We suppose that ∂R is the boundary of a well-defined region $R \subseteq [0,1]^2$.

The optimality of the reprojection obtained by Whitney extension can be described as follows. We view the height associated with the curves $\boldsymbol{b}^k(t)$ as a discrepancy between the surface data $\boldsymbol{F}$ and the boundary data. Let $\varepsilon(\boldsymbol{p})$ be the discrepancy $\eta(\boldsymbol{p})$ defined by $\boldsymbol{M}^{-1}(\boldsymbol{b}^k(t)) = \boldsymbol{p}$, *i.e.*,

the discrepancy defined by the boundary curves $\boldsymbol{b}^k(t)$ for $k = 0, \ldots, m-1$ and $t \in [0,1]$. Then, if the reprojected mesh (denoted $\bar{M}$) is to interpolate the boundary curves, the maximum absolute discrepancy $|\varepsilon(\boldsymbol{p})|$ of $\bar{M}$, measured over *all* of R, cannot be less than $\max_{p \in \partial R} |\varepsilon(\boldsymbol{p})|$, and the maximum slope of the reprojected mesh over all of R cannot be less than the slope on ∂R, defined by

$$L = \sup_{\boldsymbol{p}_1, \boldsymbol{p}_2 \in \partial R, \boldsymbol{p}_1 \neq \boldsymbol{p}_2} \frac{|\varepsilon(\boldsymbol{p}_1) - \varepsilon(\boldsymbol{p}_2)|}{\|\boldsymbol{p}_1 - \boldsymbol{p}_2\|}. \tag{2}$$

This follows from the fact that $\partial R \subseteq R$.

Now, a continuous extension of $\varepsilon(\boldsymbol{p})$ from ∂R to R will be called *Whitney* if it satisfies the Lipschitz condition

$$|\varepsilon(\boldsymbol{p}_1) - \varepsilon(\boldsymbol{p}_2)| \leq L \cdot \|\boldsymbol{p}_1 - \boldsymbol{p}_2\|$$

everywhere on R (and not just on the boundary ∂R). There exist [29] a bracketing pair of extensions $l(\boldsymbol{p})$ and $u(\boldsymbol{p})$ that are Whitney, and such that for any extension $\varepsilon(\boldsymbol{p})$ that is Whitney, we have

$$l(\boldsymbol{p}) \leq \varepsilon(\boldsymbol{p}) \leq u(\boldsymbol{p}), \quad \boldsymbol{p} \in R.$$

(The explicit definitions of $l(\boldsymbol{p})$ and $u(\boldsymbol{p})$ are given below, in (3) and (4).) Furthermore, if we take the average

$$a(\boldsymbol{p}) = \frac{1}{2}[l(\boldsymbol{p}) + u(\boldsymbol{p})],$$

then $a(\boldsymbol{p})$ is Whitney, and

$$|a(\boldsymbol{p})| \leq \sup_{q \in \partial R} |\varepsilon(\boldsymbol{q})|, \quad \boldsymbol{p} \in R.$$

Thus, using $a(\boldsymbol{p})$ to reproject the mesh, as we do below, provides an extension that has absolute error no greater than that already present along the boundary ∂R, and which has slope no greater than that already imposed by the slope of $\eta(\boldsymbol{p})$ on ∂R. It is therefore optimal (and the errors minimal) in the sense that we cannot do better.

In [28, Sec. 3.5] an alternate but computationally more expensive version of the Whitney theorem is given, appropriate for severely non-convex domains. There is a possibility in such cases, if the ordinary Whitney theorem is used, of over-estimation of $\max_j L_j$. The practical risk is small. Also, there exist [19] extensions that are smoother than the C^0-continuous extension described above, when the data along the boundary is smooth. These might be used to permit specification of joining with a given level of continuity. We have not explored this possibility.

B. Case 1: The $\boldsymbol{b}^k(t)$ are provided as input

The outline of the joining algorithm, in the case when the boundary curves $\boldsymbol{b}^k(t)$ are provided as part of the input, is as follows:

1. Project the vertices $\boldsymbol{v}_i$ of the input mesh M into $[0,1]^2$ in the *u*-*v* domain, to produce a *projected mesh*. (There is of course an approximation involved here, since the inverse images of triangles T_j are typically curvilinear sets in the *u*-*v* domain.)
2. Project a piecewise-linear approximation of each $\boldsymbol{b}^k(t)$ into $[0,1]^2$ in the *u*-*v* domain.
3. Remove a sufficient number of peripheral triangles from the projected mesh (in the *u*-*v* domain) to guarantee that the projected mesh does not extend beyond the projection of the boundary curves $\boldsymbol{b}^k(t)$, but with at least one layer of triangles removed from the periphery of the projected mesh. The remaining part of the projected mesh will be referred to as the *central mesh*. See Figure 3.
4. Triangulate the region between the projection of the boundary curves and the central mesh. (This will be referred to as the triangulation of the *external region*. See Figure 3.)
5. Reproject the vertices of the combined mesh (the central mesh and the triangulation of the external region) to R^3 using Whitney extension, as described in Sec. III-A.
6. Merge the reprojected combined mesh, along the joint boundary (in R^3) between the two parts of the combined mesh, to obtain $\bar{M}$.

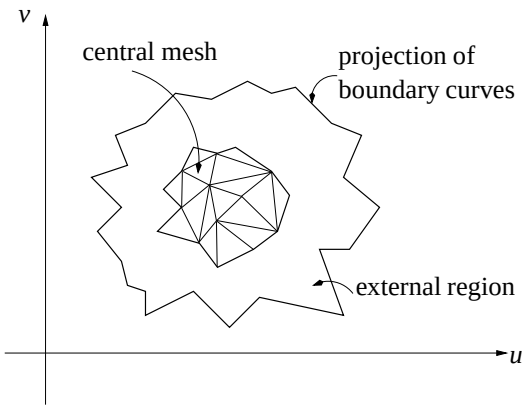

Fig. 3. Meshing domain.

The projection of the input mesh (step 1), and of the curves $\boldsymbol{b}^k(t)$ (step 2), can be dealt with in several ways [6], [30], [31], [32]; here we simply used the Fletcher-Reeves gradient algorithm provided in the GNU Scientific Library [33].

The reprojection (step 5) requires calculation of the functions $l(\boldsymbol{p})$ and $u(\boldsymbol{p})$, mentioned in Sec. III-A. The functions $l(\boldsymbol{p})$ and $u(\boldsymbol{p})$ are defined by

$$l(\boldsymbol{p}) = \sup_{\boldsymbol{q} \in \partial R} \{\varepsilon(\boldsymbol{q}) - L \cdot \|\boldsymbol{p} - \boldsymbol{q}\|\}, \quad \boldsymbol{p} \in R, \tag{3}$$

and

$$u(\boldsymbol{p}) = \inf_{\boldsymbol{q} \in \partial R} \{\varepsilon(\boldsymbol{q}) + L \cdot \|\boldsymbol{p} - \boldsymbol{q}\|\}, \quad \boldsymbol{p} \in R, \tag{4}$$

[29]. Due to the use of the piecewise-linear approximation (step 2), the calculation of the supremum in the definition of $l(\boldsymbol{p})$, and the infimum in the definition of $u(\boldsymbol{p})$, together require only 8 floating-point operations for each piecewise linear segment.

The triangulation of the external region (step 4) is done using a slightly modified version of Ruppert's Delaunay refinement algorithm [34], namely the variant [35]. Suppose that the triangulation producing the projection mesh is done using the same algorithm. Then, because we remove at least one layer of triangles in step 3, it follows that the minimum angle in the boundary of the external region is

at least $\theta = 26.45$ degrees, provided that this condition is also satisfied by the projections of the $\boldsymbol{b}^k(t)$. Consequently, it follows [35] under these hypotheses that the minimum angle in the triangulated external region is no smaller than $\arctan[(\sin\theta)/(2-\cos(\theta)]$, which is approximately 21.96 degrees.

The merging required in step 6 refers to triangle splitting when there are extra vertices along the boundary, between the two parts of the combined mesh, arising from the triangulation of the external region.

C. Case 2: Certain of the $\boldsymbol{b}^k(t)$ are not provided as input

The procedure in the case when certain of the $\boldsymbol{b}^k(t)$ are not provided is exactly the same as in Sec. III-B, except that before projecting a piecewise-linear approximation of the curves $\boldsymbol{b}^k(t)$, it may be necessary to calculate surrogates for the missing boundary curves. Note that we need $\boldsymbol{b}^k(t)$ (or a surrogate) for all k, even if no mesh patch is to be joined along certain edges.

If a curve $\boldsymbol{b}^k(t)$ is present, for a given k, it is used as in Sec. III-B.

If $\boldsymbol{b}^k(t)$ is not present, for a given k, then there are two possibilities. If there is not an adjoining mesh along edge k, then we simply use the boundary of the input mesh to compute ∂R along that edge. If there is an adjoining mesh along edge k, then we compute a piecewise linear median polyline, deleting loops if necessary. Folding causes no problem here: there is no requirement that the external region be convex in order to triangulate it.

D. Error estimates

Use of the Whitney theorem (step 5) in Sec. III-B guarantees that the slope of the reprojected mesh points, between corners of the combined-mesh triangles, will be less than or equal to the value of L along the boundary of the mesh. It does not, however, guarantee that the minimum slope of the actual triangles in the combined mesh will be less than or equal to L, as can be seen by consideration of a long thin triangle. On the other hand, if the triangulation in the u-v domain has minimum angle equal to 21.96 degrees, then it can be shown that the cosine of the angle of inclination, of a triangle in the reprojected mesh, is greater than or equal to $\{(1+L^2)[1+\left(\frac{2L}{\sin 21.96}\right)^2]\}^{-1/2}$. This follows from a straightforward trigonometric argument using spherical coordinates. The value of $\sin 21.96$ is approximately 0.384.

The problem just mentioned, related to long thin triangles, can be avoided if a long edge of such a triangle is split, and the Whitney reprojection calculated at the inserted vertex. Note however that the worst-case risk of neglecting to do the mesh-edge split is that the slope of the triangle could be unnecessarily large. There is no danger of a flipped triangle (Figure 2).

IV. Computational examples

A. Examples illustrating the two algorithms

In the accompanying figures, examples of the use of the joining algorithms are given. The examples involve joining of trimmed patches: the trimmed patch illustrated in Figure 4 is exactly the input patch shown in the upper right corner of each of Figure 5 and Figure 6. The second input patch, in the upper left corner of Figure 5 and Figure 6, is, similarly, a trimmed patch obtained from a larger untrimmed surface (not shown). The joined patches are shown in the lower part of Figure 5 and Figure 6, respectively.

Figure 7 shows two input patches with folding present. The result of joining by means of linear interpolation along polylines, as described in Section I, is shown in Figure 8. The result of using the algorithm of this paper is shown in Figure 9.

The triangulations of the input trimmed patches were obtained using Maya [4]. The triangulations of the exterior regions were obtained, as explained in Section III-B, using a variant of the Ruppert algorithm.

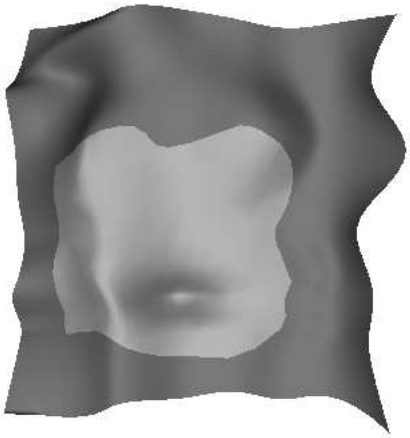

Fig. 4. Trimmed patch together with its original surface.

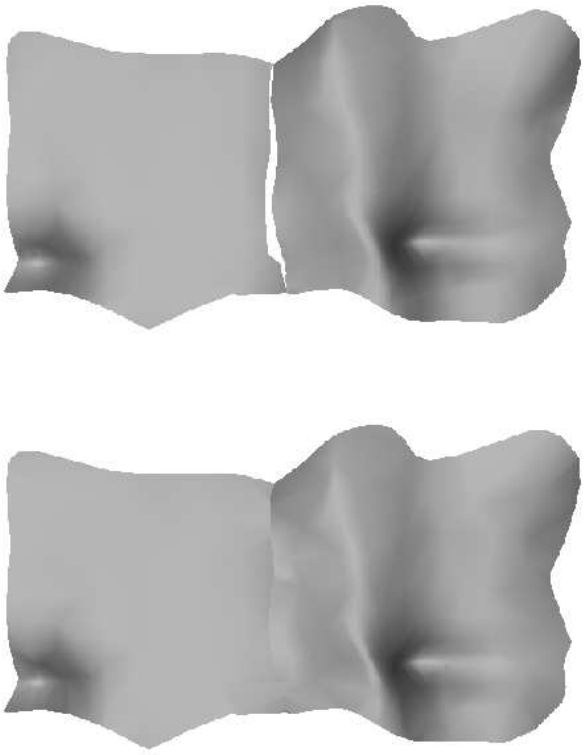

Fig. 5. Example with $\boldsymbol{b}(t)$ not provided. Top: the input trimmed patches; bottom: the result of joining.

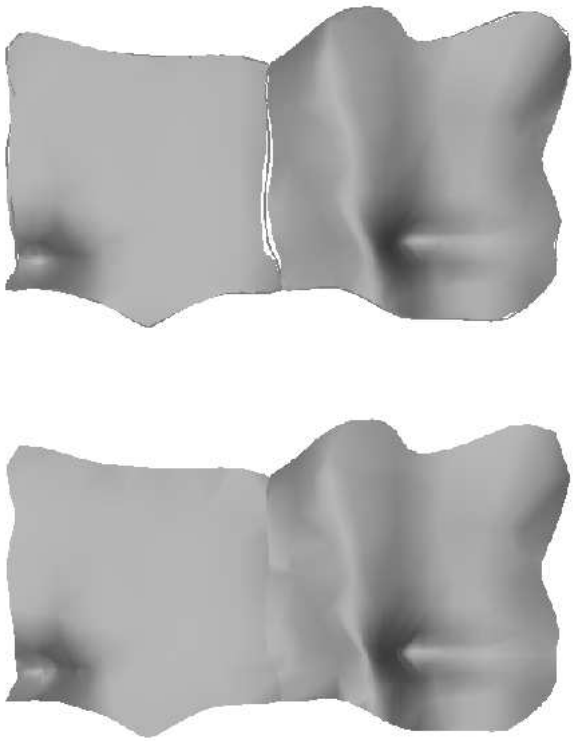

Fig. 6. Example with $\boldsymbol{b}(t)$ provided. Top: the input trimmed patches; bottom: the result of joining.

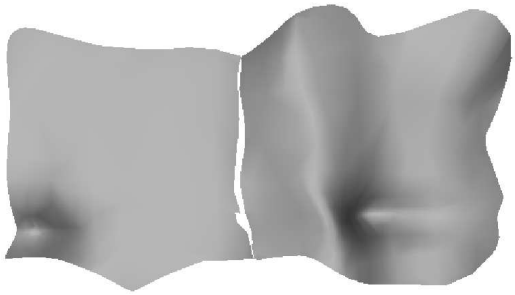

Fig. 7. Input patches with folding present.

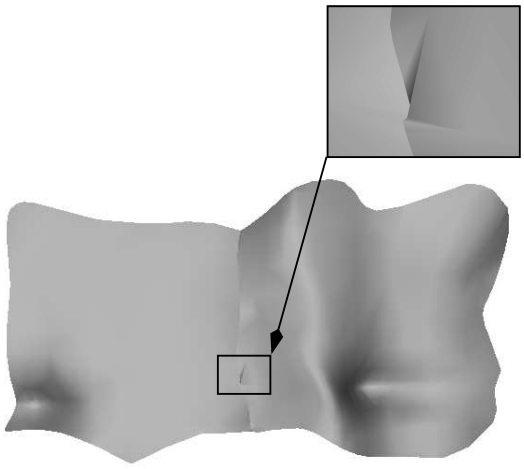

Fig. 8. Result with flipped triangle.

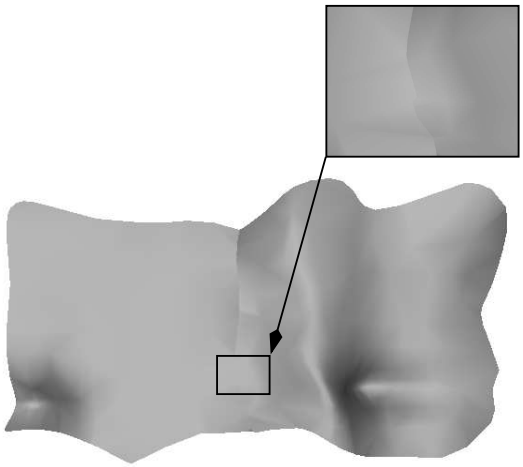

Fig. 9. Sewing result with Whitney extension.

B. Computational cost

Let σ be the number of segments in the piecewise linear approximation of the boundary curves $\boldsymbol{b}^k(t)$ (step 2 in Section III-B). The time required to do the joining, including the projection and reprojection, varies directly with $\sigma \cdot n$, where n is defined (Section II) to be the number of vertices in M. The constant of proportionality in our experiments (run on a 2.2 GHz AMD Athlon 64 3500+ processor), was approximately $0.5 \cdot 10^{-4}$. Thus, for a pair of meshes comprising $2.1K$ nodes, with $\sigma = 80$, the total time required was 8.16 seconds. (The examples shown in Figures 5 - 9 had fewer nodes, and required less time.) Whitney reprojection accounts for 65-85% of the total time cost.

V. Conclusion

Our first conclusion, as suggested in Section I, is that normal-vector criteria will be necessary if we wish to devise reliable algorithms. Note that the purpose of presenting examples like those of Figure 2 and Figure 8 is not to suggest that such examples will occur frequently when using any particular algorithm but, rather, to illustrate possibilities that must be excluded if we want provably reliable methods. One of the two main contributions of the paper is to set out the minimal requirements for an eventual proof of reliability.

Our second conclusion is that it is possible to devise algorithms, operating at reasonable cost, that will join given mesh patches together while maintaining a proxy for the normal-vector error, as well as the absolute error, at a level below that already present in the given mesh. Furthermore, the mesh in the *u*-*v* domain is not disturbed by the reprojection process, and the triangulations of the central mesh and the external region in the *u*-*v* domain can be done using the best available method. In this paper the central mesh was triangulated using Maya, while the external region was triangulated using a variant of Ruppert's algorithm, but if better methods become available, they can be used directly. Similarly, the *u*-*v* coordinates of any previously-applied mesh-fairing or smoothing algorithm will not be disturbed—only the height field is modified in order to ensure that its slope over the whole patch will not be larger than the slope along the edge of the patch.

The advantage of using normal-vector criteria for graphical simulation is clearly evident from the example of Figure 8. Further research should focus on the estimation of normal-vector error by using the mesh itself.

VI. Acknowledgments

The authors are grateful to F. Duranleau, P. Poulin, I. Stewart and T. Tautges, who provided several useful comments. Responsibility for any errors or omissions lies solely with the authors.

The research of the second author was supported in part by the Natural Sciences and Engineering Research Council of Canada.

References

[1] Kahlesz, F., Bal´azs, A. and Klein, R.; *Multiresolution rendering by sewing trimmed NURBS surfaces*, 281-288, SM '02, June 17-21, 2002, Saarbrücken, Germany.

[2] Borodin, P., Novotni, M. and Klein, R. Progressive gap closing for mesh repairing. Manuscript, Computer Graphics Group, University of Bonn, 2002.

[3] Steinbrenner, J. P., Wyman, J. and Chawner, J. R. Fast surface meshing on imperfect CAD models. Proceedings of the 9th International Meshing Roundtable, 33-41, 2000.

[4] Maya Version 7, Help Manual, Autodesk, 2006.

[5] Haimes, R. and Aftosmis, M. J. Watertight anisotropic surface meshing using quadrilateral patches. Proceedings of the 13th International Meshing Roundtable, 2004.

[6] Patrikalakis, N. M. and Maekawa, T. Shape Interrogation for Computer Aided Design and Manufacturing, Springer, 2002.

[7] Campbell, R. J. and Flynn, P. J. A survey of free-form object representation and recognition techniques. *Computer Vision and Image Understanding* 81, 166-210, 2001.

[8] Litke, N., Levin, A. and Schröder, P. Trimming for subdivision surfaces. *Computer Aided Geometric Design* (18), 463-481, 2001.

[9] Mandal, C., Qin, H. and Vermuri, B. C. A novel FEM-based dynamic framework for subdivision surfaces. *Computer-Aided Design* (32), 479-497, 2000.

[10] Kumar, G. V. V. R. and Srinivasan, P. and Shastry, K. G. and Prakash, B. G. Geometry based triangulation of multiple trimmed NURBS surfaces. *Computer-Aided Design*, (33) 439-454, 2001.

[11] ACIS 3D Toolkit, 1998. Spatial Technology, Boulder, CO.

[12] STEP International Standard, ISO 10303-42, 1997. Industrial automation systems and integration—Product data representation and exchange—Part 41. International Organization for Standardization (ISO), Reference Number ISO 10303-42 1994(E). First Edition 1994-12-15.

[13] Frey, P. J. and Borouchaki, H. Surface mesh evaluation. Proceedings of the 5th *International Meshing Roundtable*, 1996.

[14] Cohen-Steiner, D. Alliez, P. and Desbrun, M. Variational shape approximation. *ACM Trans. Graph.* (23), No. 3, 905-914, 2004.

[15] Guskov, I. Manifold-based approach to semi-regular meshing. *Graphical Models*, 2006. In press.

[16] Grinspun, E. and Schröder, P. Normal bounds for subdivision-surface interference detection. Proceedings of the IEEE Conference on Visualization, 333-340, October 2001.

[17] Volino, P. and Thalmann, N. M. Efficient self-collision detection on smoothly discretized surface animations using geometrical shape regularity. In: M. Daehlen and M. Kjelldahl K. (editors), Eurographics '94, vol. 13, No. 3. Blackwell Publishers, pp. C-155–C-164.

[18] Andersson, L.-E., Stewart, N. F. and Zidani, M. Conditions for use of a non-selfintersection conjecture. *Computer Aided Geometric Design* (23), 599-611, 2006.

[19] Whitney, H. Analytic extensions of differentiable functions defined in closed sets. *Trans. Am. Math. Soc.* (36), 63-89, 1934.

[20] Jiang, D. and Stewart, N. F. Maintaining validity of mesh-solids using floating-point arithmetic. SCAN2006: 12th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics. Duisburg, Germany, 26-29 September, 2006. Abstracts Book pp. 119-120.

[21] Alliez, P. and Cohen-Steiner, D. and Devillers, O. and L´evy, B. and Desbrun, M. Anisotropic polygonal remeshing. *ACM Trans. Graph.*, (22), No. 3, 485-493, 2003.

[22] Desbrun, M. and Meyer, M. and Schröder, P. and Barr, A. H. Implicit fairing of irregular meshes using diffusion and curvature flow. SIGGRAPH Proceedings, 1999.

[23] Beall, M. W. and Walsh, J. and Shephard, M. S. Accessing CAD geometry for mesh generation. Proceedings of the 12th International Meshing Roundtable, 2003.

[24] Liepa, P. Filling holes in meshes. Eurographics Symposium on Geometry Processing, Kobbelt, L. and Schröder, P. and Hoppe, H.(editors), 2003.

[25] Barequet, G. and Kumar, S. Repairing CAD models. Proceedings of the IEEE Conference on Visualization, 363-370, 1997.

[26] Piegl, L. A. and Tiller, W. Geometry-based triangulation of trimmed NURBS surfaces. *Computer-Aided Design*, (30) No. 1, 11-18, 1998.

[27] Bischoff, S. and Weyand, T. and Kobbelt, L. Snakes on triangle meshes. Bildverarbeitung für die Medizin, 208-212, 2005.

[28] Aronsson, G. Extension of functions satisfying Lipschitz conditions. *Ark. Mat.*, (6) No. 28, 551-561, 1967.

[29] Andersson, L.-E., Stewart, N. F. and Zidani, M. Error analysis for operations in solid modeling in the presence of uncertainty. *SIAM J. Scientific Computing*, 2007. In press.

[30] Kim, M. -S. and Elber, G. Problem reduction to parameter space. Proceedings of the 9th IMA Conference on the Mathematics of Surfaces, 82-98, 2000, London.

[31] Seong, J. -K. and Johnson, D. E. and Cohen, E. A higher dimensional formulation for robust and interactive distance queries. Proceedings of Solid and Physical Modeling, 197-205, 2006.

[32] Sud, A. and Otaduy, M. A. and Manocha, D. DiFi: fast 3D distance field computation using graphics hardware. *Computer Graphics Forum*, (33) No. 3, 2004.

[33] Galassi,M. and Davies,J. and Theiler,J. and Gough,B. and Jungman,G. and Booth,M. and Rossi,F. GNU Scientific Library Reference Manual. Revised second edition. August 2006.

[34] Ruppert, J. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. of Algorithms* (18), 548-585, 1995.

[35] Miller, G. L., Pav, S. E. and Walkington, N. J. When and why Ruppert's algorithm works. Proceedings of the 12th International Meshing Roundtable, 2003.

Author Biographies

Di Jiang is studying for her Ph.D. degree in computer science at the University of Montreal. Her e-mail address is: jiangdi@iro.umontreal.ca.

Neil F. Stewart received his Ph.D. in computer science from the University of Toronto. He is Full Professor in computer science at the University of Montreal. His e-mail address is: stewart@iro.umontreal.ca.