Hierarchical Triangular Splines

ALEX YVART and STEFANIE HAHMANN
LMC-IMAG†, Grenoble, France
and
GEORGES-PIERRE BONNEAU
GRAVIR-IMAG⋆, Grenoble, France

Smooth parametric surfaces interpolating triangular meshes are very useful for modeling surfaces of arbitrary topology. Several interpolants based on this kind of surfaces have been developed over the last fifteen years. However, with current 3D acquisition equipments, models are becoming more and more complex. Since previous interpolation methods lack of a local refinement property, there is no way to locally adapt the level of detail. In this paper, we introduce a hierarchical triangular surface model. The surface is overall tangent plane continuous and is defined parametrically as a piecewise quintic polynomial. It can be adaptively refined while preserving the overall tangent plane continuity. This model enables designers to create a complex smooth surface of arbitarry topology composed of a small number of patches, to which details can be added by locally refining the patches until an arbitrary small size is reached. It is implemented as a hierarchical data structure where the top layer describes a coarse, smooth base surface, and the lower levels encode the details in local frame coordinates.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*surfaces and object representations, Hierarchy and geometric transformations, Object hierarchies*; G.1.1 [**Numerical Analysis**]: Interpolation—*Spline and piecewise polynomial interpolation*; J.6 [**Computer-aided engineering**]: Computer-aided design (CAD)

General Terms: Design, Theory, Algorithm
Additional Key Words and Phrases: geometric modeling, hierarchical splines, arbitrary topology, Bézier patches, interpolation, local frames, multiresolution, triangular meshes, G^1-continuity

1. INTRODUCTION

1.1 Motivation

Computer-based free-form surface modeling is now being used in every possible industry. Roughly stated, two types of surfaces are widely used depending on the application targeted:

—Parametric surfaces are used in CAD/CAM software,

—Subdivision surfaces are used in computer graphics.

Author's address: Stefanie Hahmann, IMAG-LMC, B.P. 53, 38041 Grenoble Cedex 9, France. Email: [Stefanie.Hahmann—Georges-Pierre.Bonneau]@imag.fr, alex.yvart@renault.com

Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, the ACM copyright/server notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.
© 20YY ACM 0730-0301/20YY/0100-0001 $5.00

In the CAD/CAM industry, free form surfaces are usually represented by tensor product polynomial or rational maps. The knowledge of a simple explicit parameterization is required in CAD/CAM for common surface manipulations including offsetting, trimming, and surface interrogation. In order to model surfaces of arbitrary topology, several tensor product pieces need to be patched together, while maintaining an overall continuity for the surface across the patch boundaries. This is a difficult problem that has given rise to years of active research in Computer Aided Geometric Design. Specific solutions have been developed for combining N tensor product patches around a vertex. These solutions are restricted to $N = 3$ or 4, otherwise requiring the use of so-called n-sided patches [Gregory 1986], [Van Wijk 1986], [Loop and DeRose 1989], [Sarraga 1987].

In computer graphics, subdivision surfaces are widely used. These surfaces are defined as the limit of a refinement process starting on an initial control polyhedron. A refinement consists in inserting new vertices, connecting them to the mesh and computing new positions of the vertices. These surfaces are very easy and fast to compute, they can naturally handle arbitrary topological types, and they support adaptive refinement. But a simple and explicit analytical expression of the subdivided surface is generally unknown [Stam 1998].

This paper tends to merge the advantages of both methods: Our surface model exhibits an explicit polynomial parameterization as required in CAD/CAM, supports arbitrary topology and adaptive refinement, as subdivision surfaces do. Another advantage of our surface model in comparison with subdivision schemes is the ability to interpolate smoothly a given mesh. While some subdivision schemes allow for interpolation of arbitrary triangle meshes, like the butterfly scheme [Dyn et al. 1990], the results exhibit unwanted undulations requiring pre-processing of the mesh [Alkalai and Dyn 2004]. In this paper we introduce a piecewise polynomial triangular surface model, that smoothly interpolates a base triangular mesh, and that can be adaptively refined in a hierarchical manner.

1.2 Related Work

Forsey and Bartels pionered the idea of hierarchical B-splines [Forsey and Bartels 1988]. H-Splines can be locally refined using overlays. Based on this model, they were able to create a complex surface like a dragon's head from a single square with hierarchical edition. However, since this model is established over tensor product splines, it is restricted to tensor product mesh and topology.

Localized-hierarchy surface splines [Gonzalez-Ochoa and Peters 1999] extended the hierarchical spline paradigm to surfaces of arbitrary topology. They are defined locally on a hierarchy of meshes using the "reference plus offset" model of Forsey, Bartels for encoding the details. Since they are based on C^1-surface-splines [Peters 1995], the surface is defined explicitly by low degree triangular and quandrangular Bézier patches, while requiring the structure to satisfy a particular regularity property through all the levels of the hierarchy.

For triangular meshes, many works exist to build polynomial interpolants. A limited survey can be found in [Mann et al. 1992]. [Piper 1987], [Shirman and Sequin 1987] and [Jensen 1987] have developed interpolation methods over triangular meshes. Yet, these models can't be made hierarchical, since they use a Clough-Tocher like split. That is to say a new vertex is inserted in the interior of each

input triangle. New edges are created by connecting the new vertex with the existing triangle vertices. Thus each input triangle is divided into three "flattened" sub-triangles. If iterated, it leads to degenerated triangles very fast. [Loop 1994] approximated a triangular mesh without splitting the triangles. He could in theory also interpolate the mesh but with severe undulations. Quasi G^1 surfaces can be found in [Vlachos et al. 2001]. Although this method interpolates the control mesh vertices it does not guarantee G^1 continuity across patches.

The theory on subdivision surfaces was discovered in the 70's, whereas they became popular only recently in computer graphics. There exist many different schemes, a survey of which can be found in [Schröder and Zorin 1998].

All these methods construct smooth surfaces in the sense that they are at least tangent plane continuous (G^1). But people also worked directly on meshes building multiresolution mesh hierarchies [Eck et al. 1995], [Zorin et al. 1997], [Kobbelt et al. 1998]. [Certain et al. 1996] and [Lounsbery et al. 1997] developed a multiresolution analysis for meshes based on linear wavelets. Therefore the surfaces produced are piecewise linear (that is to say meshes) and not smooth, in contrast to the parametric surfaces we consider here.

1.3 Contributions & Overview

In this paper, we present a new method for hierarchical modeling of smooth surfaces of arbitrary topology. Based on the previously developed triangular interpolation scheme [Hahmann and Bonneau 2003], this method enables level of detail construction and surface editing by interpolating the vertices of a hierarchy of locally refined meshes. The initial mesh, referred to as base-mesh, can be any triangular 2-manifold mesh. Given a base mesh, a polynomial interpolating surface is computed. It is smooth in the sense that it is overall tangent plane continuous. Level of detail is then added by iterative local refinement and editing the surface. Each local surface refinement replaces a set of coarse surface patches by a set of finer surface patches while maintaining both the overall tangent plane continuity and the shape. The user can add detail by editing the refined surface patches. A hierarchical editing tool is also provided thanks to a "reference plus offset" representation.
The main features that clearly distinguish the hierarchical surface presented here from previous works include:

- Any *triangular mesh* can be dealt with, which means there are *no restrictions* on topology, geometry, genus or boundaries. It only has to be a 2-manifold mesh.
- The surface *interpolates* a hierarchy of meshes thus offering direct control for surface modifications.
- *Uniform surface model*: For both, the initial surface and for each refinement step, the same interpolant is applied in order to recompute locally the new surface part. Thus only a *few geometric quantities* need to be stored for each surface patch in order to completely compute the surface.

Further properties that the hierarchical surface inherits from the underlying surface model include:

- overall tangent plane continuity

- each surface patch is represented as a parametric polynomial triangular Bézier patch of degree five.
- the surface is local, i.e. the modification of a mesh vertex modifies only the surrounding surface patches while maintaining the tangent plane unchanged outside that region

This paper is organized as follows. Section 2 presents the general problem of refining a parametric surface. Section 3 describes the topology of our refinement process and the data structures used to encode it. the local interpolation scheme of [Hahmann and Bonneau 2003], which is the core of our hierarchical surface model, is summarized in Section 4. Section 5 explains in detail the actual computation of one refinement step. The issues related to the application of successive refinements are dealt with in Section 6. Section 7 presents the 3D modeler based on our hierarchical model, and shows several results. Finally we conclude and list some future works in Section 8.

2. PARAMETRIC REFINEMENT

The development of our scheme begins with the statement that all existing parametric surface models that interpolate a triangular mesh with tangent plane continuity (see sect. 1.2) are locally supported but not refinable. Locally supported means that when editing the surface by moving a mesh vertex, only the adjacent surface patches are modified while maintaining the overall tangent plane continuity. Consequently no details can be added to the surface that are smaller than the local support. Therefore, in order to add the advantages of multiresolution surfaces, we need a way to refine the surface and if possible, to do it locally. Once it is refined, finer details can be added, creating a new level of details. These details should not alter the smoothness (at least the tangent plane continuity) of the surface. Such a refinement and local editing process is easy with subdivision surfaces. By nature, a subdivision scheme enables to compute from the initial triangulation a refined mesh that defines the same limit surface. Editing the refined mesh results in a local editing of the limit surface.

In our setting, the input mesh acts like the control mesh of a spline surface. It controls the multiple polynomial pieces while maintaining the continuity between them. Following this analogy, we will refine the interpolating surface by a process equivalent to the knot insertion of splines. As with the knot insertion, this process will result in the subdivision of the parameter domain and the control mesh without modifying the surface itself. Afterwards, the modification of a vertex in the refined control mesh leads to a more local modification of the surface since the local support of the newly inserted vertices is smaller, and yet still preserves the continuity of the surface.

Since our interpolant is composed of polynomial pieces parameterized by triangular Bézier patches, the obvious way of subdividing the surface would be to use Bézier subdivision as in [Goldman 1983]. Nevertheless this intuitive idea is not applicable in our case, since subsequent modification of the Bézier control points after subdivision would not necessarily preserve the overall continuity of the surface. This can be illustrated with a curve example. The best equivalent of our surface

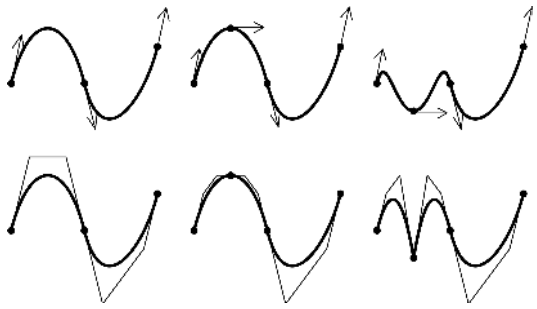


Fig. 1. Spline refinement and editing preserves tangent plane continuity, while Bézier subdivision and editing can destroy the continuity. Top row: Hermite splines - local refinement - local edition while maintaining C^1-continuity. Bottom row: Bézier curves - Bézier subdivision - local deformation breaking the tangent continuity

interpolation scheme for curves is Hermite interpolation. Figure 1 (left) shows two cubic polynomial curve segments joining with C^1-continuity, on the top generated by Hermite interpolation, on the bottom described by their Bézier control polygons. The first part of the curve is then refined (Figure 1, middle). The top figure shows the resulting Hermite parameters (points and derivatives) and the bottom shows the new subdivided Bézier control points. Changing the newly inserted Bézier control point destroys the C^1-continuity (Figure 1, bottom right) while changing the new control point of the Hermite interpolant will not affect the C^1-continuity (Figure 1, top right).

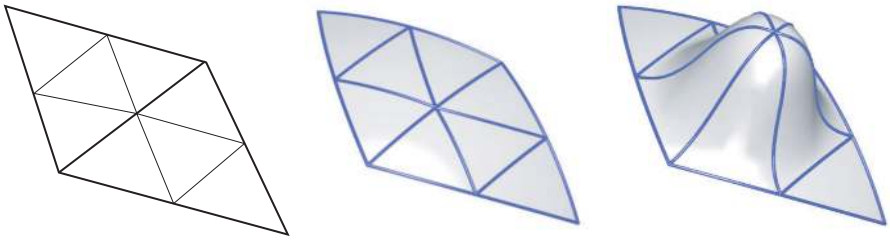
This simple curve example illustrates why Bézier subdivision of the interpolating triangular G^1 Bézier surface is alone not suitable for refinement, and justifies the need for a higher level, spline-like refinement scheme as as described in Section 5.

3. TOPOLOGICAL SPLIT / DATA STRUCTURE

The construction of the hierarchical spline surface is a process that first interpolates a given triangular base mesh, and then builds a hierarchy of surfaces by iteratively refining the base surface locally. In order to offer the possibility to edit that surface at any level of resolution, this process is based on a data structure that contains both the topological information encoding the hierarchical subdivision of the parameter domain, and the geometrical information related to the mapping from the domain to the surface.

The basic operation for refining the parameter domain consists in subdividing an edge into two pieces, and splitting its two neighboring triangular faces into four sub-faces as shown in Figure 2, left. This particular choice of local refinement has been made for the following reasons. Triangular Bézier patches can be subdivided into two parts, three parts (Clough-Tocher like), or four parts, as described all in [Goldman 1983]. But the Clough-Tocher subdivision and the subdivision into two parts are unstable, since repeated subdivision leads to long and fine triangles.

This 4-split operation inserts five new vertices in the parameter domain. The surface points corresponding to the four surrounding vertices in the parameter domain may not be edited, whereas the position corresponding to the new central vertex



(a) refinement of the parameter domain (b) refined surface (c) refined surface after edition

Fig. 2. Two adjacent patches are subdivided into four sub-patches by inserting a new vertex in the middle of the common edge. The vertex becomes an *editable* vertex. Its position can be modified leading to a *local* modification of the surface while maintaining the overall tangent plane continuity of the surface.

can be freely modified, as shown in Figure 2, right. Therefore the central vertex is called *editable*. During a subsequent subdivision of another edge, a non-editable vertex can change its status and become editable.

An appropriate data structure must provide an easy access to the different levels of detail from top to bottom and from bottom to top. Similar to [Certain et al. 1996] we build a forest of quad trees, using the following data structure:

type *Vertex* = **record**
    parentFace [2]: **array of pointers to** *Face*
    (geometric information)
end record

type *Face* = **record**
    cornerVertex [3]: **array of pointers to** *Vertex*
    edgeVertex [3]: **array of pointers to** *Vertex*
    childFace [4]: **array of pointers to** *Face*
    (geometric information)
end record

Each node in the quad tree encodes a *Face*. The root nodes of the forest represent the triangular faces of the input mesh. When a face is subdivided, it points to the four *childFaces*. Furthermore each face points to its three vertices (*cornerVertex* in the face record) and to the three middle edge vertices (*edgeVertex* in the face record). The three edge vertices are created as soon as the face is created. But they become editable only when their two *parentFaces* exist and are subdivided. Note that at level 0, the *parentFace* pointers in the vertex record encode the adjacencies between the input triangles.

The geometrical information that is stored in the *vertex* and *face* records is explained in Section 4.

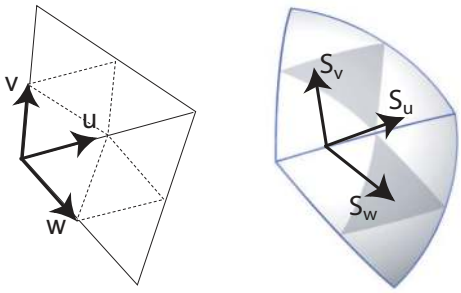


Fig. 3. Parameterization of the interpolant. Left: parameter domain; right: macro-patches with tangent directions.

4. POLYNOMIAL INTERPOLATION METHOD

Our hierarchical surface construction starts by interpolating a base mesh by a tangent plane continuous surface before refining it. We require the initial interpolating surface to be a polynomial of degree as low as possible, and to be defined locally. In the related work section 1.2, a list of previous methods that fulfill these constraints is given. Unfortunately these methods are based on the Clough-Tocher split and therefore can't be used in a hierarchical refinement context. Therefore we choose the method of [Hahmann and Bonneau 2003] and more specifically its subdivision invariant version [Hahmann et al. 2003] for interpolating the base mesh. Originally the first method was developed to produce high quality interpolating surfaces, and did so by giving the possibility to freely choose the tangent directions of the patch boundary curves at the mesh vertices. It turns out later that the arbitrary choice of the first derivatives together with the fact that the method is not based on a Clough-Tocher split but on a 4-split, makes it suitable with some modifications for refinement, as shown in [Hahmann et al. 2003].

4.1 Overview of the interpolation method

In this section, we briefly recall the main points of this interpolant, useful for understanding the remaining of the paper. As stated earlier, the surface is parameterized over the input mesh. Each triangle is mapped to a group of four Bézier patches of degree five. This group is referred to as a *macro-patch* consisting of four subtriangles that split the input triangle regularly as shown in Figure 3. This regular 4-split of the parameter domain is a key ingredient making the interpolant refinable.

Referring to the notations of Figure 3, the tangent plane continuity constraint between the two macro patches is written:

$$\Phi(u)\frac{\partial S}{\partial u} = \mu(u)\frac{\partial S}{\partial v} + \nu(u)\frac{\partial S}{\partial w} \ . \tag{1}$$

Equation (1) states that the three partial derivatives along and across the common curve between two macro patches are coplanar, thus defining a continuous tangent plane. Φ, μ and ν are scalar functions. They are chosen to be piecewise linear,

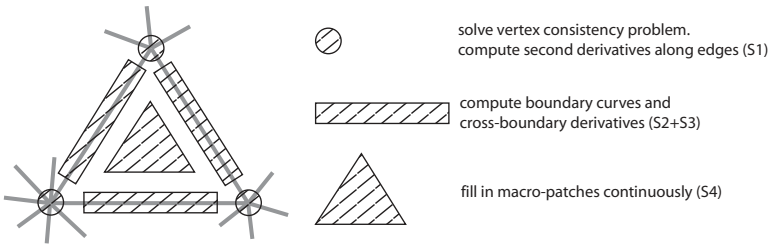


Fig. 4. Computing the interpolating triangular spline in 4 steps

that is the lowest possible degree. (1) has to be fulfilled along all edges of the input mesh. At a mesh vertex this leads to the so-called vertex consistency problem, which states in essence that the derivatives of the surface must be consistently chosen [DeRose 1990; Peters 2002; Sarraga 1987; Watkins 1988].

The interpolant is built in four consecutive steps (Figure 4):

(S1) The vertex consistency problem is solved *(all vertices)*.
(S2) The boundary curves are computed *(all edges)*.
(S3) The tangent plane continuity between the macro-patches is fulfilled *(all edges)*.
(S4) The continuity inside each macro-patch is ensured *(all faces)*.

Figure 5 shows an example of our interpolant applied on a mesh with a complex topology. We briefly elaborate these four steps focusing on the salient features, in particular the description of the free parameters, and their influence on the interpolating surface. A complete description of the interpolant is given in [Hahmann and Bonneau 2003].

Vertex consistency (S1). The position of a macro-patch corner vertex can be freely chosen. It's a free parameter. The scheme is interpolating this vertex. For a vertex of order N, the free parameters are the N first derivatives at the vertex in the direction of each edge, and the N twists (mixed second partial derivatives) of the patches joining at that vertex. From these free parameters, the remaining second derivatives along the edges are computed, and the scalar functions Φ, μ and ν are fixed. Changing these parameters affect the N macro-patches joining at the vertex. In Figure 6, the free parameters correspond to the control-points symbolized by solid discs, while the circles symbolize the control-points corresponding to the second derivatives.

Boundary curves (S2). They are chosen so that

$$\frac{\partial S}{\partial u} = \mu(u)\,\nu(u)\,H(u) \quad \text{along the boundary curve,} \tag{2}$$

where H is a C^0 piecewise degree 2 polynomial function. The boundary curves are therefore piecewise quintic curves, that is the lowest possible degree fulfilling the continuity constraints. It turns out that once the derivatives at the vertices are consistently chosen by (S1) the boundary curves are completely determined by (2),

free parameters	**influence**	**symbol in Fig. 6**
corner vertex of macro-patch	affect all macro-patches around that vertex	crosses
N first derivatives and N twists at each interpolated vertex of degree N	affect all macro-patches around that vertex	solid circles
6 control points inside each macro-patch	affect a single macro-patch	solid triangles

Table I. The free parameters controlling the interpolating surface, and their influence on the surface

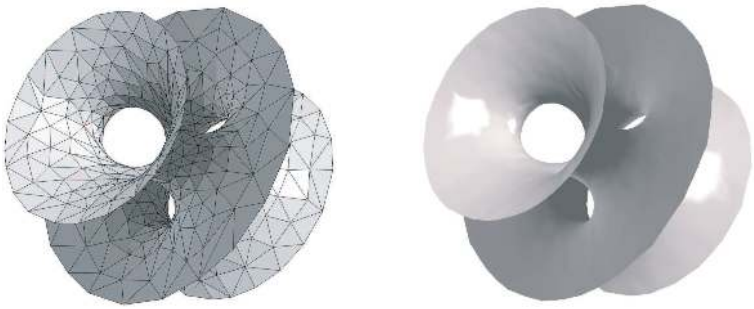


Fig. 5. G^1 continuous surface interpolation. Left: input mesh. Right: interpolating surface.

hence no free parameters in step (S2).

Cross derivatives (S3). In order to ensure the tangent plane continuity between the macro-patches, the cross partial derivatives $\frac{\partial S}{\partial v}$ and $\frac{\partial S}{\partial w}$ have to be computed. In terms of the Bézier patches, that amounts to compute the first inner row of Bézier control points on each side of the boundary curves in such a way that:

$$\begin{cases} \frac{\partial S}{\partial v} &= \frac{1}{2}\big(\Phi(u)\nu(u)H(u)\big) + \nu(u)W(u) \\ \frac{\partial S}{\partial w} &= \frac{1}{2}\big(\Phi(u)\mu(u)H(u)\big) - \mu(u)W(u) \end{cases} \tag{3}$$

where W is a piecewise cubic polynomial function. Referring to Figure 3, the first inner row of Bézier control points for the upper macro-patch is determined by the first equation in (3), whereas the second equation gives these control points for the bottom macro-patch. There are no free parameters in step (S3). In Figure 6, the control-points computed by step (S2) and (S3) are shown as squares.

Continuity inside a macro-patch (S4). In this step, for each macro-patch, there remains 15 unknown Bézier control points, symbolized as triangles in Figure 6. Six of these points can be freely chosen (the solid triangles), the 9 others are computed such as to ensure C^1-continuity between the 4 Bézier patches of the macro-patch. Changing one of these 6 free parameters affects a single macro-patch.

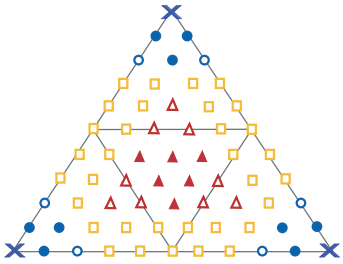


Fig. 6. Control points of a macro-patch which is composed of four quintic Bézier triangles. The control points symbolized by crosses and circles are dealt with by step (S1). Squared control points are computed by steps (S2) and (S3). Step (S4) involves the control points symbolized by triangles.

4.2 Free parameters

Table I summarizes the free parameters and their influence on the interpolating surface, and Figure 6 symbolizes them as Bézier control points of a macro-patch. All solid crosses, circles and triangles represent free parameters of the interpolation scheme. The user is allowed to move them and the G^1 constraints are then satisfied by solving for the position of the hollow quads and triangles following steps (S1)-(S4).

A good choice of the degrees of freedom is crucial for the final shape. For a given input mesh quite different shapes of the final surface are possible. It depends on the artist's intent. It's possible to create a tight surface imitating the control net, or a rounded and blew up surface. A fair and visually pleasant shape would probably lie between. It is possible to use a minimization of an energy norm that keeps the scheme local and gives smooth results, as explained in [Hahmann and Bonneau 2003].

The *geometric information* field in the *vertex* and *face* records of the data structure introduced in Section 3 is used to store the free parameters of the interpolation scheme. Notice, that these free parameters are sufficient to completely and uniquely determine the surface. In other words, it's not necessary to store all the control points for each macro-patch. The remaining control points are determined by the G^1 constraints.

5. LOCAL SURFACE REFINEMENT

Once a surface is computed over the initial coarse mesh, the user can interact and change the surface. It can be done either by editing one of the free parameters detailed in Section 4, or by editing the vertices of the base mesh. However, these modifications would alter the surface corresponding to the edited face or around the edited vertex. It means that only details as large as the base mesh can be dealt with. The interpolation scheme does not permit to add high frequency details.

In this section we detail the basic refinement step sketched earlier in Sections 2 and 3. Recall that the refinement results from 4-splitting two adjacent domain triangles, see Section 3. The domain vertex inserted in the middle of the common edge becomes "editable", meaning that the corresponding surface point can be

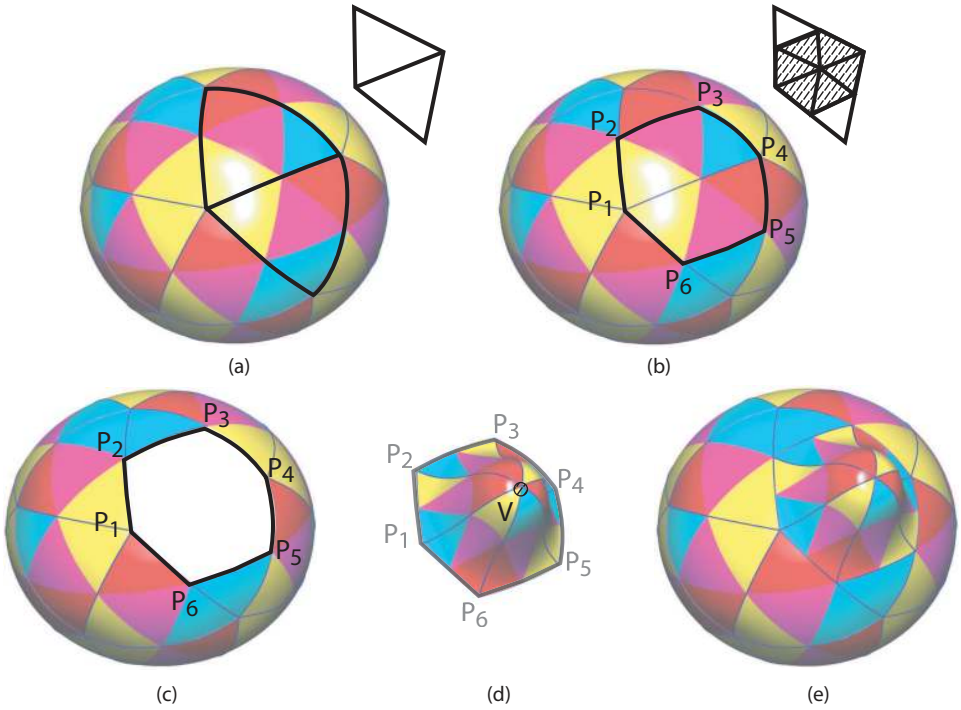


Fig. 7. Local Surface Refinement

freely specified by the user. The problem we face now is to construct the mapping from the subdivided domain to the refined surface in such a way that: (1) the editable vertex is interpolated with the new surface position specified by the user, (2) the surface is modified only locally, and (3) the overall tangent plane continuity is preserved.

Before giving the details, let us emphasize that this mapping will be constructed using *the same interpolation scheme* that has been applied to compute the base surface presented in Section 4. In the following discussion, we refer to the notations in Figure 7. For clarity, this figure illustrates one refinement step starting with the base surface, though the following discussion applies to successive refinements as well. Figure 7(a) shows the base surface with bold curves highlighting the mapping of the two domain triangles, which are shown on the top right. The different colors correspond to the different triangular Bézier patches on the surface. Recall that each domain triangle is mapped to a group of 4 Bézier patches, referred to as a macro-patch.

The refinement procedure subdivides the two domain triangles as shown in Figure 7(b) on the top right. The gray area maps to the surface region surrounded by vertices $P_1, P_2, P_3, P_4, P_5, P_6$, see Figure 7(b). The refinement procedure replaces this region, which is initially composed of 6 triangular Bézier patches, by 6 macro-patches, i.e. 24 triangular Bézier patches interpolating the new editable vertex (see

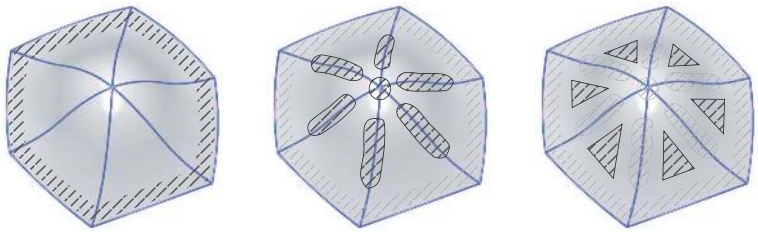


Fig. 8. Local Surface Refinement. Left: tangent continuity with the surrounding surface is ensured by Bézier subdivision. Middle and right: the interpolation method of Section 4 is applied locally to compute the refined surface.

Figure 7(d, e) as an example). Outside this region, the surface is *not modified*. This implies in particular that the new surface portion that is computed has to have the same curve and the same tangent planes along its boundary in order to fill in smoothly. Inside this region, if the editable vertex is not displaced, the surface will theoretically not exactly be reproduced. But in all our examples the maximal relative error between the refined surface (without vertex displacement) and the original one is always less than 2%.

The six new macro-patches are obtained by applying the interpolation scheme to a local mesh composed of the six triangles around the common interior vertex, see 7(d). This central vertex V is freely chosen by the user. The other vertices $P_1, \ldots, P_6$ are fixed by the surrounding surface, see Figure 7(b).

The construction of the interpolant that fills in the hole is done in two steps. First, a smooth (tangent plane continuous) joint to the previous surface along the boundary is established by fixing the boundary control points and the first inner row of control points of the inner macro-patches, see Figure 8 left. All these control points are obtained from the previous surface patches by applying a Bézier subdivision, that splits each patch into four sub-patches, see [Goldman 1983]. Note that in practice this Bézier subdivision is not applied to the whole patches but only to the control points that are needed here.

Secondly, the interpolant is applied. After the user has chosen the free parameters related to the interior vertex and the six inner faces (see Table I), the interpolant computes the surface following the algorithm of Section 4: The vertex consistency problem is solved for the interior vertex and for the outer vertices (step S1); the boundary curves (step S2) and cross boundary derivatives (step S3) are computed for the six interior boundary curves; and then the macro-patches are filled in (step S4), see Figures 8 middle, right.

6. HIERARCHICAL EDITION AND REFINEMENT

In the previous section, the procedure for a single refinement and local edition of the surface is described. The present section deals with the different issues that arise when *successive* refinement and local editions of the initial surface are carried

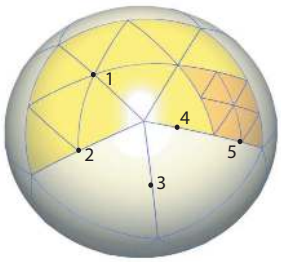


Fig. 9. 5 types of edges in the hierarchical surface model

out in a hierarchical manner.

6.1 Topological issues

At any time during the design process the user faces a hierarchical mesh structure comprising of faces, edges and vertices at different levels. Let V denote the set of all vertices and V_0 denote the set of vertices at level 0 (i.e. V_0 are the vertices of the input mesh). There is a bijection between the set of all edges and $V \backslash V_0$: each edge in the mesh is uniquely labeled by an *edge vertex* pointer in the data structure introduced in Section 3.

There are five types of edges at any time during the design process. Figure 9 illustrates these five types.

Type 1: The edge has two parent faces, both of which are subdivided.

Type 2: The edge has two parent faces, only one of which is subdivided.

Type 3: The edge has two parent faces, none of which is subdivided.

Type 4: The edge has only one parent face, which is not subdivided.

Type 5: The edge has only one parent face, which is subdivided.

Type 1 edges correspond to vertices that are editable. Type 2 and type 3 edges may be refined using the procedure described in Section 5. Type 4 and Type 5 edges may not be directly refined since one of their parent faces does not exist. Before refining them, their parent faces must be created by subdividing an edge at an upper level.

The necessary and sufficient condition for a vertex to be editable is either that the vertex belongs to V_0, or that the vertex has two parent faces and these parent faces are subdivided. As pointed out in Section 3, a non-editable vertex can become editable without explicitly refining its edge. This is illustrated in figure 10: in 10a the edge of vertex A is refined, the outer vertex B is not editable yet. Then in 10b the edge of vertex C is refined and the B becomes editable.

6.2 Geometrical issues

When one or several free parameters of an editable vertex are modified at a coarse level, all the surface patches affected by their modification in the finer levels of the

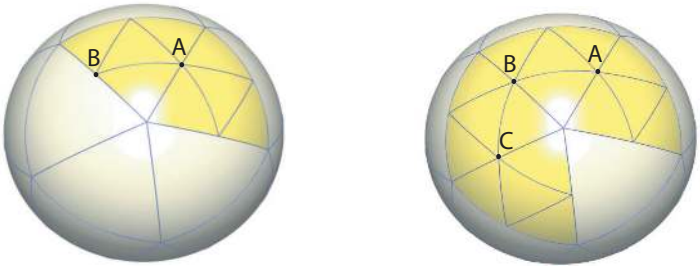


Fig. 10. After successive refinement of vertices A and C the vertex B becomes editable.

hierarchy must be updated. For an intuitive editing the details encoded in the finer levels of the model hierarchy must follow the movement of the surface at the coarse level. We use a "reference plus offset" model developed by [Forsey and Bartels 1988] so that the local coordinates of the surface are tied to the upper level, dynamically following the edition.

6.2.1 *Encoding the free parameters in a local frame.* The free parameters of the input mesh, including position, derivatives, twists of vertices in V_0, and the 6 free inner control-point of the macro-patches, are encoded using the global coordinate system. Each free parameter of the surface at level i, $i > 0$, is defined in a local frame which depends only on the position, tangent plane and normal of the surface at the level $i - 1$. More precisely, to encode the position, derivatives and twists for an editable vertex v at level i subdividing an edge e at level $i - 1$, we use a right-handed orthonormal frame defined by the following vectors: one vector is the normalized tangent at the mid-point of the curve corresponding to e in the surface at level $i - 1$; another vector is the normal of the surface at that point. To encode the 6 free inner control points of one macro-patch at level i, we use a right-handed orthonormal frame with its origin at the point on the surface at level $i-1$ corresponding to the barycenter of the parameter domain, two unit orthogonal vectors in the tangent plane, and the normal.

6.2.2 *Editing a free parameter.* When editing a free parameter of the surface at any level of the hierarchy the surface is modified only locally inside the corresponding zone of influence of the parameter. The zone of influence of a vertex parameter (position, tangents, twists) corresponds to a ring of macro-patches around that vertex. The zone of influence of a face parameter (6 inner control points) is the corresponding macro-patch. When editing a free parameter at level i, all macro patches inside its zone of influence Z and all corresponding hierarchies need to be updated. This includes local frames related to free parameters at finer level $j > i$, and the corresponding macro-patches. This update is performed by the following breadth-first traversal of the forest of macro-patches. In an initial step, the local coordinates of the new parameter at level i are computed, and the macro-patches at level i in Z are updated. Note that the local frame corresponding to the modified parameter is unchanged. When the traversal reaches level j, all local frames of free parameters at level $k \leq j$ and all corresponding macro-patches are already

updated. The transition from level j to level $j+1$ works as follows. Firstly, the local frames of free parameters at level $j+1$ whose zone of influence lies completely inside Z must be updated. Therefore the value of these free parameters is changed, although their local coordinates are not modified. Secondly, the macro-patches at level $j+1$ inside Z are updated. The same procedure is repeated through all levels down to the finest level.

Note that during this traversal, free parameters whose zone of influence lies partially in Z correspond to vertices on the boundary of Z. The local frame and hence the value of these free parameters are not affected by the modification inside Z, because the surface is G1 continuous and does not change outside Z. Thus first order geometric quantities that are needed to define the local frames, don't change along the boundary of Z.

Let us now describe the updating of the surface hierarchy with the example of Figure 11. It shows a hierarchical triangular surface with 4 levels of detail, $i = 0, 1, 2, 3$. Let us modify the position of vertex P_{11} at level 1. Its zone of influence Z consists of a ring of 6 macro-patches at level 1, delineated by the black ring.

Level 1: Once the new local coordinates of P_{11} have been computed the 6 macro-patches inside Z are re-evaluated by applying locally the interpolation method of Section 4. Outside the zone Z the surface is not modified, and G^1-continuity is maintained across the boundary of Z. This implies that the position and and first derivatives of all points along that boundary haven't been modified as well. The modification of the 6 macro-patches inside Z now percolates through the corresponding hierarchies.

Level 2: First the local frames of all free parameters of level 2 whose zone of influence lies completely inside Z are updated, i.e. the local frames corresponding to the editable vertices P_{21} and P_{22} and all local frames of the inner control points of the level 2 macro-patches, highlighted in red. Then all red level 2 macro-patches inside Z need to be re-evaluated. Again, the algorithm of Section 4 is applied around P_{21}, P_{22}, P_{23}. Note that the macro-patches around P_{23} outside Z don't undergo any modification, since the values of the free parameters at P_{23} haven't changed.

Level 3: At the finest level 3 the local frames of the green point P_{31} and of the green macro-patches can now be updated and the green macro-patches can be recomputed. Here again the algorithm of Section 4 is applied at P_{31}, P_{32}, but all macro-patches around P_{32} outside Z are not modified.

7. RESULTS

Based on the hierarchical triangular spline presented in the previous sections a 3D modeler has been implemented in C++ using OpenGL. The user can load a mesh, compute a surface on this mesh, modify the surface, refine locally the surface, edit the surface at the finer level, and edit the surface at any level. The program allows interactively designing and editing of a complex surface in "real time". In fact the reaction time for an update of the surface (recomputation and display) after modification of a vertex position or any other free parameter is less than 10 ms on

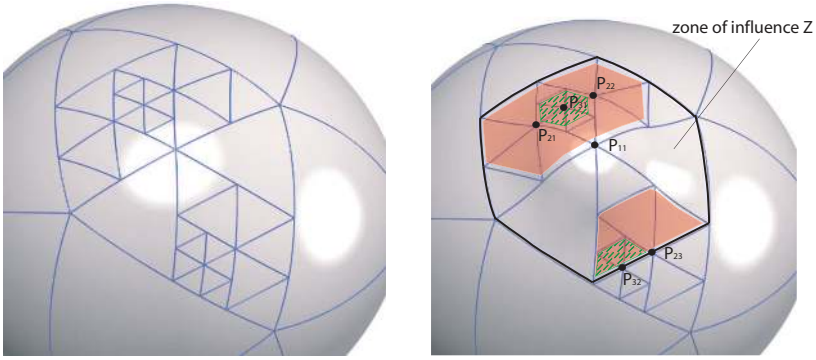


Fig. 11. Editing a free parameter related to a vertex P_{11} at level 1 of the hierarchy leads to an updating of all macro-patches around and their corresponding hierarchies. The surface is modified only inside the ring of macro-patches delineated by the fat black lines.

a 1.4Ghz portable PC. The speed doesn't depend on the total number of patches, but only on the depth of the subtree having the modified vertex at root. The speed of 10ms for an update is the worst case for a hierarchy of 5 levels. In that particular case, we refined a test surface up to level 4. Each initial coarse surface patch at the top level 0 in the hierarchy is now replaced by $4^4 = 256$ fine patches. We then edited a vertex at level 0. Since that vertex was adjacent to 5 coarse surface patches, the number of fine surface patches that had to be recomputed is therefore $1280 = 5 \times 256$. Additional to that, all local frames through the 4 levels of the hierarchy need to be updated.

Different interactive modeling tools are supported by the hierarchical triangular spline. The surface model itself offers some degrees of freedom that can easily be made available to the designer as intuitive design handles. For example the editable vertices that are interpolated by the surface can be picked on the surface and displaced while the surrounding surface is following continuously. Furthermore at each editable vertex, all the tangent directions of the incoming patch boundary curves are free, but subject to lie in the same plane. They define the surface tangent plane and the normal vector at these points. By offering the designer the possibility to interact directly with these geometric quantities, several design effects can be obtained. Modifying the normal direction gives a new orientation to the tangent plane, while modifying the length of the normal vector has a tension effect influencing the local curvature. A twisting effect is obtained by rotating the tangent plane. Some of these design tools are illustrated together with the modeler in Figure 12.

In addition to these tools the hierarchical structure of the surface together with the representation of finer surface parts as offsets in local coordinate frames of the coarser surface parts allow for two kind of editions:

—when the surface around a vertex is edited *at a finer level*, the surface is only modified locally to add details

—when a vertex *at a coarser level* is modified, the surface is modified more globally

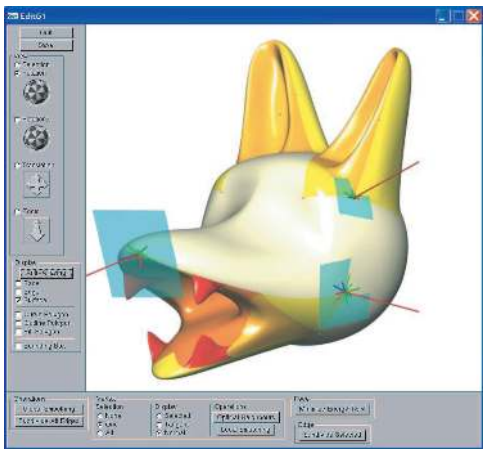


Fig. 12. The 3D Modeler

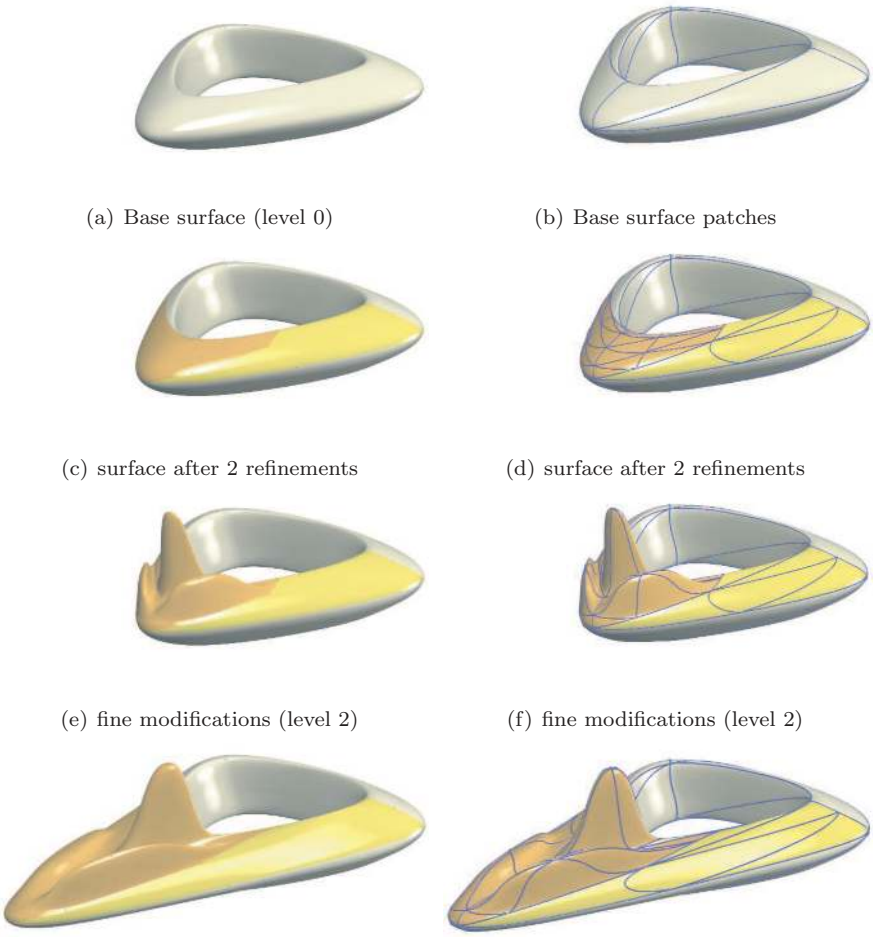
keeping the local details unchanged.

The successive steps of the algorithm are illustrated in Figure 13 with a genus 1 base mesh. A smooth surface interpolating the base mesh is shown in Figure 13(a). 13(b) shows the same surface together with boundary curves of the macro-patches. The surface is refined in Figure 13(c), 13(d): two edges are subdivided at level 1, and three edges are subdivided at level 2. Three vertices are edited at level two to obtain the surface shown in Figure 13(e), 13(f). With this technique, the size of details are determined by the level at which they are added. Finally, a vertex at level 0 is edited to illustrate a large scale hierarchical edition. The previously created detail follow naturally the displacement, see Figure 13(g), 13(h).

Figure 14 shows the result of a design session creating a complex character. Starting from an icosahedron (20 triangles, 12 vertices) as base mesh, the interpolating base surface looks like a sphere (Figure 14(a)). Editing one vertex, the user gets a muzzle (Figure 14(b)). Refining locally the end of the muzzle and the head, the user can open a mouth and add ears (Figure 14(c)). Once again refining, the ears and mouth are enhanced with more details (Figure 14(d)). To finalize this dog's head, four fangs are added to the mouth (Figure 14(i)). The mouth can later be closed by editing only one vertex high enough in the hierarchy (at the end of the muzzle) (Figure 15).

8. CONCLUSIONS AND FUTURE WORK

In this paper, a new hierarchical surface model has been introduced. Similar to H-splines but dealing with arbitrary topology, we have developed an interpolation scheme with level of details. This method works on any 2-manifold triangular mesh: it can have boundaries or not; it can be of arbitrary genus; there is no restriction on the degree of the vertices. Our model simplifies the design process through hierarchical construction : a base mesh is built first, representing a coarse approximation



(a) Base surface (level 0)

(b) Base surface patches

(c) surface after 2 refinements

(d) surface after 2 refinements

(e) fine modifications (level 2)

(f) fine modifications (level 2)

(g) coarse modifications (level 0)

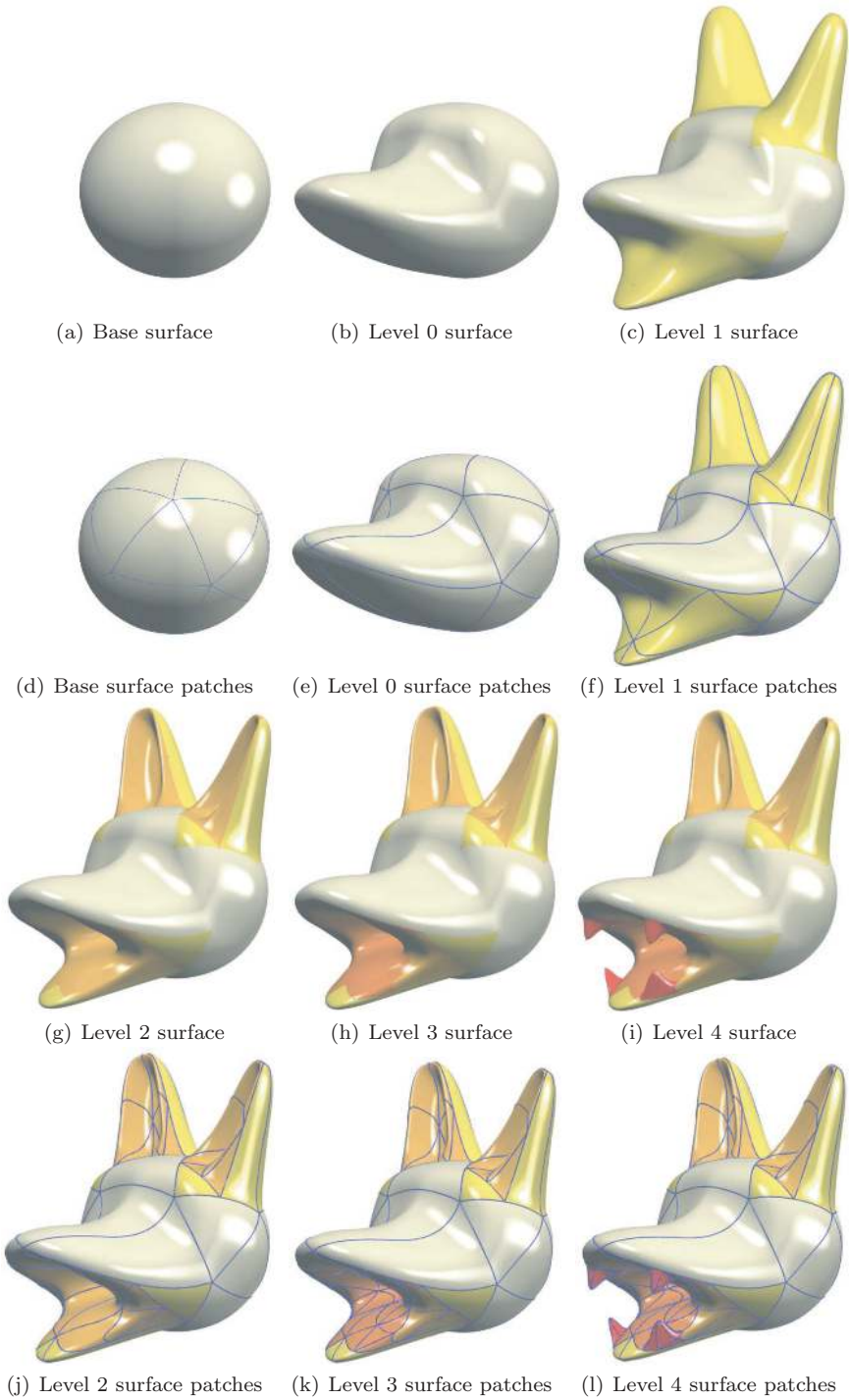
(h) coarse modifications (level 0)

Fig. 13. A smooth surface based on a mesh of genus 1: refinement, local edition, large scale edition.

of the object and setting its topology. A smooth base surface is computed over this base mesh with free parameters to change its global shape. Finer details are then added wherever needed thanks to local refinement. This model allows to easily deform objects on a large scale while preserving local details by editing only a few points in the top level of the hierarchy.

This model combines the advantages of CAD/CAM tensor product surfaces since it has an explicit polynomial parameterization, with the advantages of subdivision surfaces, since it can handle arbitrary topology, and multiresolution modeling.

In the near future, we intend to investigate:



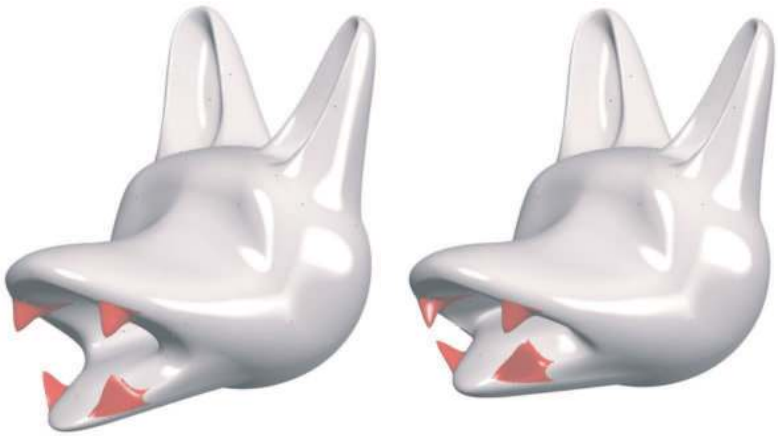
(a) Base surface (b) Level 0 surface (c) Level 1 surface

(d) Base surface patches (e) Level 0 surface patches (f) Level 1 surface patches

(g) Level 2 surface (h) Level 3 surface (i) Level 4 surface

(j) Level 2 surface patches (k) Level 3 surface patches (l) Level 4 surface patches

Fig. 14. A dog's head model



(a) Mouth open (b) Mouth closed

Fig. 15. Editing a surface at a coarse level.

—3D reconstruction: Real objects can be represented with this model thanks to a surface fitting technique. The coarse mesh, free parameters and local refinements can be computed to best fit a given dataset. With the same method, it is possible to transform a model from any other object representation to this hierarchical scheme.

—Topological issues: Our current system doesn't allow to change the topology of the object during the design process: the final object has to have the same genus as the base mesh. This restriction does not arise from the continuity condition between patches but is due to the hierarchical data structure used in this paper. By using a winged-edge data structure as in [Gonzalez-Ochoa and Peters 1999], we will remove this restriction.

REFERENCES

Alkalai, N. and Dyn, N. 2004. Optimising 3d triangulations: improving the initial triangulation for the butterfly subdivision scheme. In *Advances in Multiresolution for Geometric Modeling*, N. Dodson, M. Sabin, and M. Floater, Eds. Springer-Verlag.

Certain, A., Popovic, J., DeRose, T., Duchamp, T., Salesin, D., and Stuetzle, W. 1996. Interactive multiresolution surface viewing. *Computer Graphics 30,* Annual Conference Series, 91–98.

DeRose, T. D. 1990. Necessary and sufficient conditions for tangent plane continuity of Bézier surfaces. *Comput. Aided Geom. Des. 7,* 1-4, 165–179.

Dyn, N., Levine, D., and Gregory, J. A. 1990. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Trans. Graph. 9,* 2, 160–169.

Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M., and Stuetzle, W. 1995. Multiresolution analysis of arbitrary meshes. *Computer Graphics 29,* Annual Conference Series, 173–182.

Forsey, D. R. and Bartels, R. H. 1988. Hierarchical B-spline refinement. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*. ACM Press, 205–212.

Goldman, R. N. 1983. Subdivision algorithms for Bézier triangles. *Computer-Aided Design 22,* 3, 159–166.

Gonzalez-Ochoa, C. and Peters, J. 1999. Localized-hierarchy surface splines (less). In *Proceedings of the 1999 symposium on Interactive 3D graphics*. ACM Press, 7–15.

Gregory, J. A. 1986. N-sided surface patches. In *The Mathematics of Surfaces*, J. A. Gregory, Ed. Clarendon Press, Oxford, 217–232.

Hahmann, S. and Bonneau, G.-P. 2003. Polynomial surfaces interpolating arbitrary triangulations. *IEEE Transactions on Visualization and Computer Graphics 9,* 1, 99–109.

Hahmann, S., Bonneau, G.-P., and Yvart, A. 2003. Subdivision invariant polynomial interpolation. In *Visualization and Mathematics III*, K. P. e. H.C. Hege, Ed. Mathematics and Visualization. Springer, 191–202.

Jensen, T. 1987. Assembling triangular and rectangular patches and multivariate splines. In *Geometric Modeling: Algorithms and New Trends*, G. Farin, Ed. SIAM, 203–220.

Kobbelt, L., Campagna, S., Vorsatz, J., and Seidel, H.-P. 1998. Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. ACM Press, 105–114.

Loop, C. 1994. A G^1 triangular spline surface of arbitrary topological type. *Computer Aided Geometric Design 11*, 303–330.

Loop, C. T. and DeRose, T. D. 1989. A multisided generalization of Bézier surfaces. *ACM Trans. Graph. 8,* 3, 204–234.

Lounsbery, M., DeRose, T. D., and Warren, J. 1997. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transactions on Graphics 16,* 1, 34–73.

Mann, S., Loop, C., Lounsbery, M., Meyers, D., J. Painter and, T. D., and Sloan, K. 1992. A survey of parametric scattered data fitting using triangular interpolants. In *Curve and Surface Design*, H. Hagen, Ed. SIAM, 145–172.

Peters, J. 1995. C1-surface splines. *SIAM J. Numer. Anal. 32,* 2, 645–666.

Peters, J. 2002. Geometric continuity. In *Handbook of Computer Aided Geometric Design*. Elsevier, 193–229.

Piper, B. 1987. Visually smooth interpolation with triangular Bézier patches. In *Geometric Modeling: Algorithms and new Trends*, G. Farin, Ed. SIAM, 221–233.

Sarraga, R. F. 1987. G1 interpolation of generally unrestricted cubic Bézier curves. *Comput. Aided Geom. Des. 4,* 1-2, 23–39.

Schröder, P. and Zorin, D. 1998. Subdivision for modeling and animation. Course notes of Siggraph 98. ACM SIGGRAPH.

Shirman, L. A. and Sequin, C. H. 1987. Local surface interpolation with Bézier patches. *Computer Aided Geometric Design 4*, 279–295.

Stam, J. 1998. Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values. *Computer Graphics 32,* Annual Conference Series, 395–404.

Van Wijk, J. J. 1986. Bicubic patches for approximating non-rectangular control meshes. *Constructive Approximation 3*, 1–13.

Vlachos, A., Peters, J., Boyd, C., and Mitchell, J. L. 2001. Curved pn triangles. In *SI3D '01: Proceedings of the 2001 symposium on Interactive 3D graphics*. ACM Press, 159–166.

Watkins, M. A. 1988. Problems in geometric continuity. *Computer Aided Design 20,* 8, 499–502.

Zorin, D., Schröder, P., and Sweldens, W. 1997. Interactive multiresolution mesh editing. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 259–268.