G^1 rational blend interpolatory schemes: a comparative study

Maria Boschiroli[a,b,*], Christoph Fünfzig[c], Lucia Romani[b], Gudrun Albrecht[a]

[a] *Univ Lille Nord de France, UVHC, LAMAV-CGAO, FR no. 2956, Le Mont Houy, F-59313 Valenciennes, France*

[b] *Dipartimento di Matematica e Applicazioni, Università di Milano-Bicocca, Via Cozzi 53, 20125 Milano, Italia*

[c] *LE2I (UMR CNRS 5158), Université de Bourgogne, 9 Avenue Alain Savary, F-21078 Dijon, France*

Abstract

Interpolation of triangular meshes is a subject of great interest in many computer graphics related applications, as, for example, gaming and realtime rendering. One of the main approaches to interpolate the positions and normals of the mesh vertices is the use of parametric triangular Bézier patches. As it is well known, any method aiming at constructing a parametric, tangent plane (G^1) continuous surface has to deal with the vertex consistency problem. In this article, we propose a comparison of three recently appeared methods that use a particular technique called *rational blend* to avoid this problem. Together with these three methods we present a new scheme, a cubic Gregory patch, that has been inspired by one of them. Our comparison includes an analysis of their computational costs on CPU and GPU, a study of their capabilities of reproducing analytic surfaces and their response to different surface interrogation methods on arbitrary triangle meshes with a low triangle count that actually occur in their real-world use.

*Corresponding author: Maria Boschiroli, Tel. +39 347 1566378, Fax +33 (0)3 27 51 19 00.

Email addresses:
`Maria.Boschiroli@univ-valenciennes.fr,maria.boschiroli@gmail.com` (Maria Boschiroli), `c.fuenfzig@gmx.de` (Christoph Fünfzig), `lucia.romani@unimib.it` (Lucia Romani), `Gudrun.Albrecht@univ-valenciennes.fr` (Gudrun Albrecht)

Preprint submitted to Graphical Models *July 11, 2011*

Keywords:
triangular mesh, G^1 local interpolation, rational blend, Gregory patch, Bézier triangle, shape interrogation.
2010 MSC: 65D05,
2010 MSC: 65D17,
2010 MSC: 65D18,
2010 MSC: 68U05,
2010 MSC: 68U07.

1. Introduction

Triangular meshes, namely meshes in which the faces are triangular and any number of faces may join at a vertex, are sufficiently general to represent surfaces of arbitrary genus. For this reason their interpolation is a subject of great interest in many computer graphics related applications such as gaming and realtime rendering.

Parametric triangular Bézier patches are a simple geometric primitive that can be used to interpolate scattered data on triangular meshes while locally controlling the surface by manipulating its control points. The idea behind the use of these patches is that each original flat triangle of the input mesh is replaced by a curved shape, defined as a parametric triangular Bézier patch interpolating the three vertex positions and the associated normals.

Not surprisingly, every method that tries to solve a data fitting problem encounters the same main difficulty: dealing with the smoothness of the surface. In [1] a survey on the existing methods for the construction of continuous (C^0) parametric interpolants on triangular meshes can be found. These schemes, which construct Bézier patches using only the information related to the underlying triangle, emerged as attractive solutions responding to the requirements of resource-limited hardware environments.

However, to be useful for surface design, a parametric data fitting scheme must produce a smooth surface. From a geometric point of view, the concept of C^1 continuity is not suitable to characterise the smoothness of a surface since a change in the parameterisation of one of two adjacent patches changes the cross boundary derivatives of that patch, thus destroying the C^1 continuity. Therefore, in practice, the concept of tangent plane continuity, also known as G^1 continuity, is used (see e.g. [2] for a formal definition of G^1 continuity between triangular Bézier patches).

Constructing *two* patches that meet with G^1 continuity is straight-forward. On the contrary, a complex problem called *vertex consistency problem* arises when constructing a closed network of *more than two* G^1 joined patches incident to a vertex [3, 4]. Every scheme aiming at constructing a tangent plane continuous surface has to cope with this problem. The G^1 methods proposed in the literature either bypass it avoiding the computation of the solution of the associated linear system or find a way to make it solvable.

In [5] a survey of the G^1-continuous parametric interpolatory schemes for triangular meshes proposed up to the beginning of the nineties is provided. The authors classify several of the most famous methods, like Shirman-Séquin [6], Nielson [7], and triangular Gregory Patch [8], and offer a detailed comparison of them.

In the present article, we focus on three methods appeared after this survey that use a particular technique called *rational blends*. Together with these three methods we present a new approach, a cubic Gregory patch that has been inspired by one of them.

The remainder of the paper article is organised as follows. In section 2, the rational blend technique is presented in detail followed by an explanation of the three methods and the presentation of our new cubic scheme. In section 3, we first analyse their computational costs (section 3.1) and then compare the schemes by looking at the reproduction of analytic surfaces like the sphere and the torus (section 3.2). Finally, in section 3.3 we investigate their response to surface interrogation methods on arbitrary triangle meshes with a low triangle count, which actually occur in real-world use of these schemes. To conclude, in section 4 we summarise the main results of our comparative study.

2. G^1 rational blend interpolatory schemes

The key idea behind the schemes we are going to present is that each original flat triangle of the input mesh can be replaced by a curved shape, namely a parametric triangular Bézier patch interpolating the three vertex positions and vertex normals. Therefore, the patch's control net is constructed only by means of the point and normal information at the vertices of the input mesh.

In order to introduce the schemes let us consider a subset of 4 triangles as illustrated in Figure 1, the central on with vertices $\mathbf{p}_0$, $\mathbf{p}_1$, $\mathbf{p}_2$, and respective unit normal vectors $\mathbf{n}_0$, $\mathbf{n}_1$, $\mathbf{n}_2$, as well as edge vectors $\mathbf{e}_1 = \mathbf{p}_1 - \mathbf{p}_0$, $\mathbf{e}_2 =$

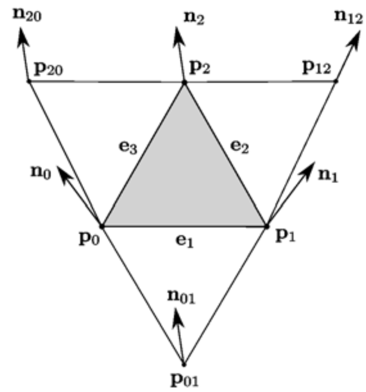


Figure 1: Notation for the vertices and respective normals of the input flat triangles.

$\mathbf{p}_2 - \mathbf{p}_1$, $\mathbf{e}_3 = \mathbf{p}_0 - \mathbf{p}_2$. Considering the neighbouring triangle adjacent to the edge $\mathbf{e}_1$, let us use the notation $\mathbf{p}_{01}$ for its remaining vertex and $\mathbf{n}_{01}$ for its associated normal, and analogously we define $\mathbf{p}_{12}$, $\mathbf{n}_{12}$ with respect to the edge $\mathbf{e}_2$ and $\mathbf{p}_{20}$, $\mathbf{n}_{20}$ with respect to the edge $\mathbf{e}_3$. Additionally, we refer to the tangent plane in $\mathbf{p}_i$, which is defined by $\mathbf{n}_i$, by $\boldsymbol{\tau}_i$, $i = 0, 1, 2, 01, 12, 20$.

Using a triangular network of control points $\mathbf{b}_{ijk}$ ($i + j + k = n$, $i, j, k \geqslant 0$) and degree-n bivariate Bernstein polynomials $B^n_{ijk}(u, v, w) = \frac{n!}{i!j!k!} u^i v^j w^k$ ($u + v + w = 1$), a degree-n triangular Bézier patch is defined by

$$\mathbf{t}(u, v, w) = \sum_{i+j+k=n} \mathbf{b}_{ijk} B^n_{ijk}(u, v, w).$$

It maps a triangular domain $D \subset \mathbb{R}^2$ to an affine space, typically $\mathbb{R}^3$, where u, v and w are the barycentric coordinates in D.

The approach we survey here is based on the creation of a triangular Bézier patch by means of *rational blends*. Multiple triangular Bézier patches are created such that each patch is G^1-continuous to its neighbour along only one triangle edge. To evaluate the resulting rational blend interpolant at some parameter values (u, v, w), each of the constructed Bézier patches is evaluated at these parameter values, then an affine combination of these points is taken. The coefficients of the affine combination are rational functions of the parameters, hence the name rational blend. Therefore a rational blend degree-n triangular Bézier patch is defined by

$$\mathbf{s}(u,v,w) = \sum_{i+j+k=n} \mathbf{b}_{ijk}(u,v,w) B^n_{ijk}(u,v,w),$$

where the control points $\mathbf{b}_{ijk}(u,v,w)$ are affine combinations of the constructed points using rational blending functions.

Each boundary of the resulting interpolant has the tangent plane field of one of the constructed patches and therefore the patch has G^1 joins along all the boundaries. The only points on the boundary that have contributions from more than one patch are the corners. The two patches that contribute to tangent plane continuity at the corner will in general have different mixed second order partial derivatives. Vertex consistency is bypassed by allowing inconsistent mixed partial second order derivatives at the corner points.

In Figure 2 the points to be blended to define the control points $\mathbf{b}_{ijk}(u,v,w)$ are shown schematically for the four schemes compared in the next sections. We review parametric hybrid triangular Bézier patches in section 2.1, PNG1 triangles in section 2.2 and Walton and Meek's Gregory patch in section 2.3. Finally, in section 2.4 we propose a new cubic Gregory patch inspired by Walton and Meek's patch.

2.1. Parametric hybrid triangular Bézier patches

This first scheme was proposed in [9, 10] and is based on a method introduced in [11] by Foley and Opitz for interpolation of scattered data above a plane using a functional hybrid cubic Bézier patch.

The idea of Davidchuck and Mann is to "parameterise" this method by choosing a plane for each triangle pair, project the vertices of the triangle and its neighbour onto that plane and then perform the functional Foley-Opitz C^1 construction on the projected points. In Figure 3 one example of the projection of a triangle pair is shown. Once a plane is chosen as a natural parameterisation, five points for each neighbour are constructed using only the triangle vertices and the associated normals. The control points for the cubic boundary curve are defined by Hermite interpolation and the Foley-Opitz cross boundary construction [11] determines the first line of interior control points. In Figure 4, for example, the five red points constructed from the edge $\mathbf{e}_1$ are shown. Thus, finally, three sets of five points $\mathbf{b}_{ijk,1}$, $\mathbf{b}_{ijk,2}$ and $\mathbf{b}_{ijk,3}$ are computed, each set representing a C^1 construction along one triangle edge. These three sets of points share the same triangle vertices but, in general, differ in the rest of the boundary and in the interior. Figure

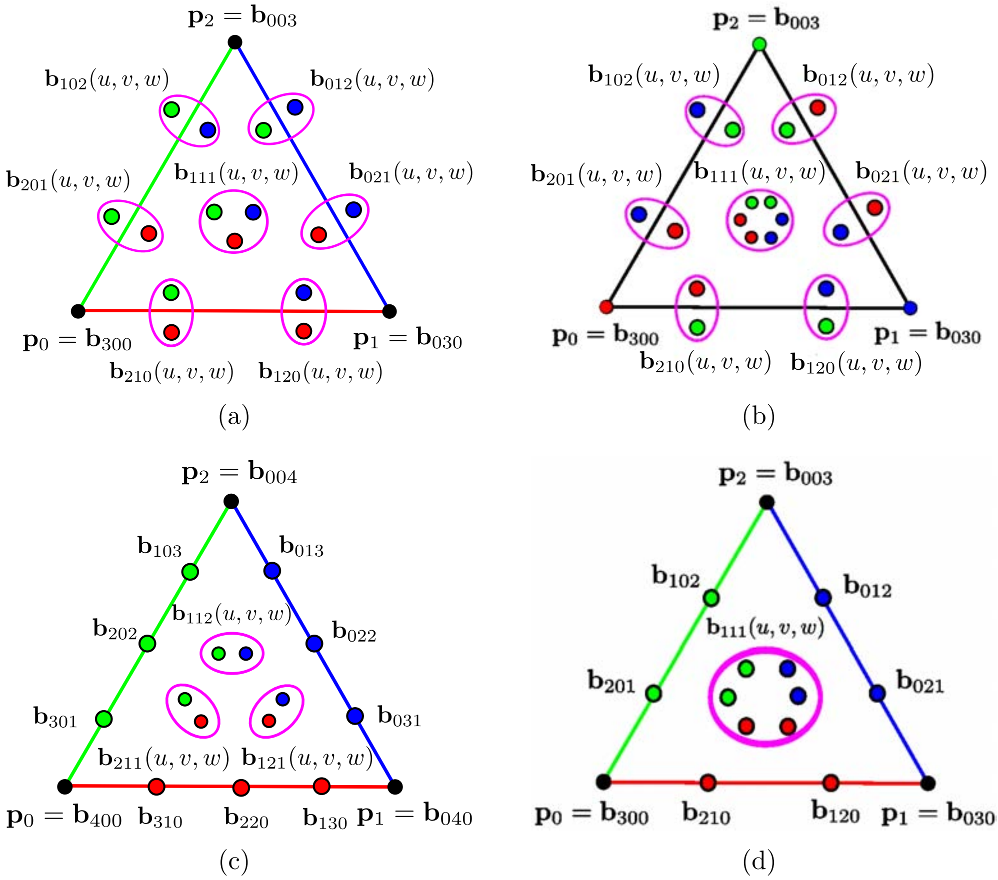


Figure 2: Points defining the control points $\mathbf{b}_{ijk}(u,v,w)$ for (a) Parametric hybrid patch, (b) PNG1 triangles, (c) Walton and Meek's Gregory patch and (d) Cubic version of Walton and Meek's patch.

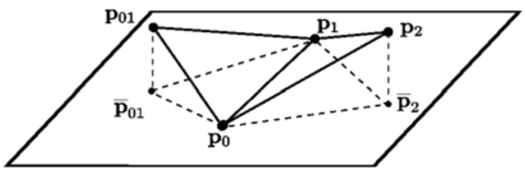


Figure 3: Plane used to parameterise neighbouring patch pairs.

2(a) shows the entire domain control net for the *parametric hybrid triangular Bézier patch*.

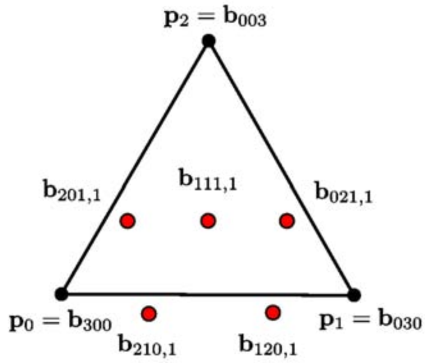


Figure 4: Five points (in red) are constructed from the edge $\mathbf{e}_1$.

These three sets of points are then blended together to define the control points $\mathbf{b}_{ijk}(u, v, w)$. As concerns the control points on the border, they are obtained by an asymmetric blend. On the edge $\mathbf{e}_1$, for example,

$$\mathbf{b}_{210}(u, v, w) = \frac{(1-w)v^2\mathbf{b}_{210,1} + (1-v)w^2\mathbf{b}_{210,3}}{(1-w)v^2 + (1-v)w^2},$$
$$\mathbf{b}_{120}(u, v, w) = \frac{(1-w)u^2\mathbf{b}_{120,1} + (1-u)w^2\mathbf{b}_{120,2}}{(1-w)u^2 + (1-u)w^2};$$

using Nielson's blending functions, firstly used in [7], the central control point is defined by

$$\mathbf{b}_{111}(u, v, w) = a_0(u, v, w)\mathbf{b}_{111,1} + a_1(u, v, w)\mathbf{b}_{111,2} + a_2(u, v, w)\mathbf{b}_{111,3},$$

where

$$a_i(t_0, t_1, t_2) = \frac{t_j t_k}{t_i t_j + t_i t_k + t_j t_k}, \; i \neq j, \; i \neq k, \; j \neq k.$$

We observe that this construction heavily depends on the plane chosen for the parameterisation and, as a consequence, this choice is crucial for controlling the control points' positions. In particular, the orientation of the plane is extremely important. Two different planes are proposed in [10]. One failsafe method is to take the plane that is perpendicular to the bisecting plane of the two neighbouring triangles and that also contains their common edge. Another possibility is to use the information provided by the normals at the triangle vertices to construct the plane, by taking, for example, the plane orthogonal to the average of the normals at the two triangle vertices on the common border. Although the second construction does not always guarantee a valid plane, in general it creates better shaped surfaces. In section 3 we show some examples of meshes exhibiting stability problems related to an inconvenient choice of this plane (more details can be found in [10]).

2.2. PNG1 Triangles

PNG1 triangles [12] are similar only in spirit to the hybrid parametric patches since cubic triangular Bézier patches for each edge of a triangle are constructed. Actually, as shown in Figure 2, this scheme differs from the previously described one, as the points to be blended to define the Bézier control points are obtained starting from the vertices of the triangle. For example, the red points in Figure 2(b) are computed using $\mathbf{p}_0$ and $\boldsymbol{\tau}_0$.

For the sake of simplicity, let us explain how the points are constructed with respect to the edge $\mathbf{e}_1$, i.e., the eight points $\mathbf{b}_{201,0}$, $\mathbf{b}_{201,1}$, $\mathbf{b}_{021,0}$, $\mathbf{b}_{021,1}$, $\mathbf{b}_{210,0}$, $\mathbf{b}_{120,1}$, $\mathbf{b}_{111,\mathbf{p}_0,01}$ and $\mathbf{b}_{111,\mathbf{p}_1,01}$, shown in Figure 5. The other points are generated similarly.

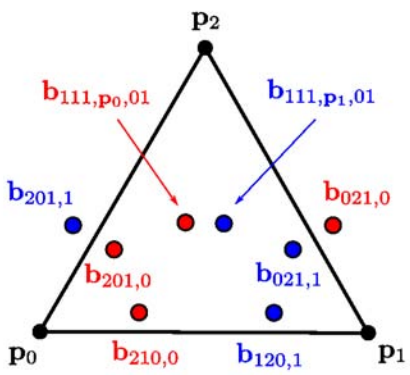


Figure 5: The eight points $\mathbf{b}_{201,0}$, $\mathbf{b}_{201,1}$, $\mathbf{b}_{021,0}$, $\mathbf{b}_{021,1}$, $\mathbf{b}_{210,0}$, $\mathbf{b}_{120,1}$, $\mathbf{b}_{111,\mathbf{p}_0,01}$ and $\mathbf{b}_{111,\mathbf{p}_1,01}$ constructed from $\mathbf{p}_0$ and $\mathbf{p}_1$ with respect to the edge $\mathbf{e}_1$.

First, the points $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_{01}$ are projected in the direction of $\mathbf{n}_0$ onto the tangent plane $\boldsymbol{\tau}_0$, Figure 6 left, and the points $\mathbf{p}_0$, $\mathbf{p}_{01}$, and $\mathbf{p}_2$ in the direction of $\mathbf{n}_1$ onto the tangent plane $\boldsymbol{\tau}_1$, Figure 6 right. The result of these projections are two adjacent triangles $\triangle \mathbf{p}_0\overline{\mathbf{p}}_1^{\tau_0}\overline{\mathbf{p}}_2^{\tau_0}$, $\triangle \mathbf{p}_0\overline{\mathbf{p}}_1^{\tau_0}\overline{\mathbf{p}}_{01}^{\tau_0}$ in the plane $\boldsymbol{\tau}_0$ and $\triangle \overline{\mathbf{p}}_0^{\tau_1}\mathbf{p}_1\overline{\mathbf{p}}_2^{\tau_1}$, $\triangle \overline{\mathbf{p}}_0^{\tau_1}\mathbf{p}_1\overline{\mathbf{p}}_{01}^{\tau_1}$ in $\boldsymbol{\tau}_1$. Subdivision of the edges $\mathbf{p}_0\overline{\mathbf{p}}_1^{\tau_0}$, $\mathbf{p}_0\overline{\mathbf{p}}_2^{\tau_0}$ and $\mathbf{p}_0\overline{\mathbf{p}}_{01}^{\tau_0}$ by factor 1/3 provides a pair of subtriangles (marked in red in Figure 6 left) whose vertices on the edges $\mathbf{p}_0\overline{\mathbf{p}}_1^{\tau_0}$ and $\mathbf{p}_0\overline{\mathbf{p}}_2^{\tau_0}$ define, respectively, the points $\mathbf{b}_{210,0}$ and $\mathbf{b}_{201,0}$. Analogously, on $\boldsymbol{\tau}_1$ subdivision of the triangle edges provides a pair of subtriangles (marked in blue in Figure 6 right) that defines the points $\mathbf{b}_{120,1}$ and $\mathbf{b}_{021,1}$.

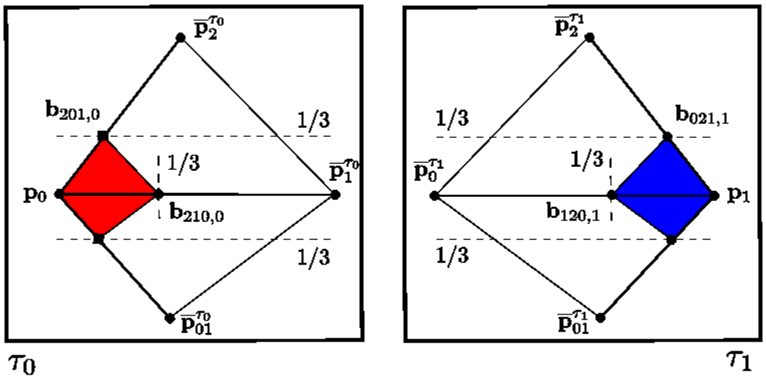


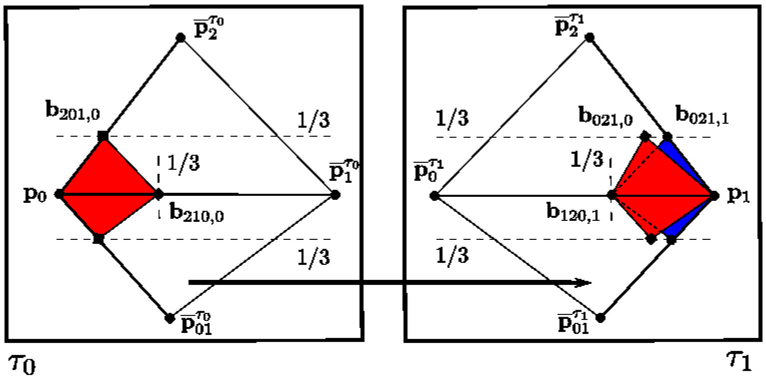
Figure 6: Construction of points $\mathbf{b}_{201,0}$, $\mathbf{b}_{210,0}$, $\mathbf{b}_{120,1}$ and $\mathbf{b}_{021,1}$.

An affine transformation of the triangle $\triangle \mathbf{p}_0\mathbf{b}_{201,0}\mathbf{b}_{210,0}$ from the tangent plane $\boldsymbol{\tau}_0$ into the tangent plane $\boldsymbol{\tau}_1$ provides the point $\mathbf{b}_{021,0}$ (Figure 7(a)) and an affine transformation of the triangle $\triangle \mathbf{p}_1\mathbf{b}_{021,1}\mathbf{b}_{120,1}$ from $\boldsymbol{\tau}_1$ into $\boldsymbol{\tau}_0$ provides the point $\mathbf{b}_{201,1}$ (Figure 7(b)).

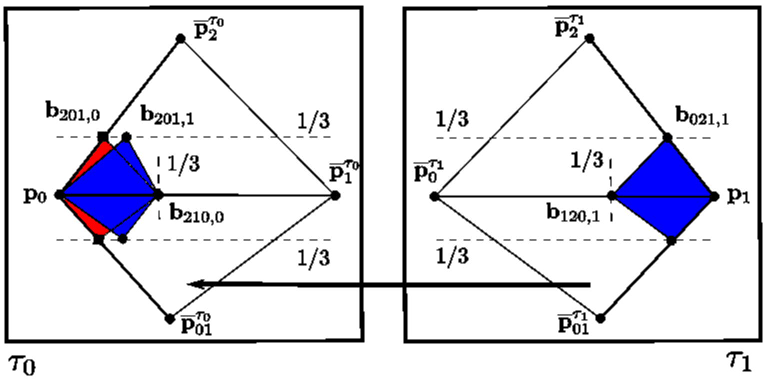
For the two interior points, let $\boldsymbol{\tau}_{e01}$ be the plane defined by $\mathbf{n}_{e01}$, where

$$\begin{aligned}\mathbf{g}_1 &= \mathbf{b}_{120,1} - \mathbf{b}_{210,0}\\ \mathbf{g}_2 &= (\mathbf{n}_T + \mathbf{n}_{T01}) \times \mathbf{g}_1\\ \mathbf{n}_{e01} &= \mathbf{g}_1 \times \mathbf{g}_2,\end{aligned}$$

$\mathbf{n}_T$ is the normal of the triangle plane, and $\mathbf{n}_{T01}$ denotes the normal of the neighbour triangle plane. As illustrated in Figure 8, a transfer of the red triangle $\triangle \mathbf{p}_0\mathbf{b}_{201,0}\mathbf{b}_{210,0}$ from $\boldsymbol{\tau}_0$ to $\boldsymbol{\tau}_{e01}$ provides the points $\mathbf{b}_{111,\mathbf{p}_0,01}$, and a transfer of the blue triangle $\triangle \mathbf{p}_1\mathbf{b}_{021,1}\mathbf{b}_{120,1}$ from $\boldsymbol{\tau}_1$ to $\boldsymbol{\tau}_{e01}$, provides $\mathbf{b}_{111,\mathbf{p}_1,01}$.



(a)



(b)

Figure 7: (a) Affine transformation of the triangle from tangent plane $\boldsymbol{\tau}_0$ into the tangent plane $\boldsymbol{\tau}_1$. (b) Affine transformation of the triangle from tangent plane $\boldsymbol{\tau}_1$ into the tangent plane $\boldsymbol{\tau}_0$.

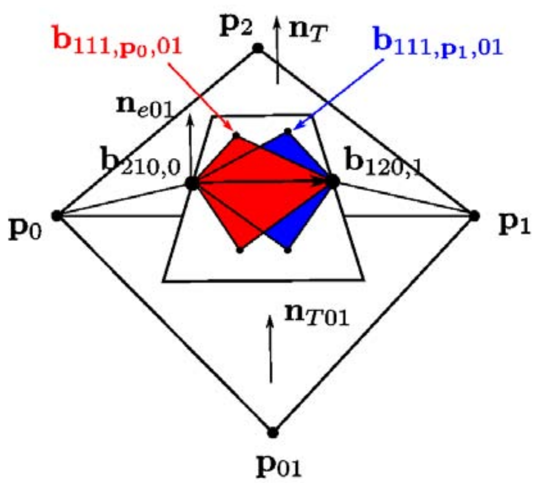


Figure 8: Construction of the middle plane by two adjacent triangle normals and $\mathbf{b}_{120,1} - \mathbf{b}_{210,0}$ to provide the points $\mathbf{b}_{111,\mathbf{p}_0,01}$ and $\mathbf{b}_{111,\mathbf{p}_1,01}$.

Applying the described procedure to the edges $\mathbf{e}_2$ and $\mathbf{e}_3$, provides all the points as schematically grouped and coloured in Figure 2(b).

The blending functions for the final control points $\mathbf{b}_{ijk}(u, v, w)$ are derived by imposing the conditions to get a G^1 join across the edges. Again an asymmetric blend is proposed for the boundary control points. On the edge $\mathbf{e}_1$, for example,

$$\mathbf{b}_{210}(u, v, w) = \frac{1}{w^2 + (1-w)^2} \left((1-w)^2 \mathbf{b}_{210,0} + w^2 \mathbf{b}_{210,2}\right),$$
$$\mathbf{b}_{120}(u, v, w) = \frac{1}{w^2 + (1-w)^2} \left((1-w)^2 \mathbf{b}_{120,1} + w^2 \mathbf{b}_{120,2}\right),$$

and Nielson-like functions for the six interior points yield

$$\begin{aligned}\mathbf{b}_{111}(u, v, w) = & \frac{uw}{uv + uw + vw} \left(\frac{w(1-u)\mathbf{b}_{111,\mathbf{p}_2,20} + u(1-w)\mathbf{b}_{111,\mathbf{p}_0,20}}{w + u - 2uw}\right) + \\ & + \frac{uv}{uv + uw + vw} \left(\frac{u(1-v)\mathbf{b}_{111,\mathbf{p}_0,01} + v(1-u)\mathbf{b}_{111,\mathbf{p}_1,01}}{u + v - 2uv}\right) + \\ & + \frac{vw}{uv + uw + vw} \left(\frac{v(1-w)\mathbf{b}_{111,\mathbf{p}_1,12} + w(1-v)\mathbf{b}_{111,\mathbf{p}_2,12}}{w + v - 2vw}\right). \end{aligned} \tag{1}$$

2.3. Walton and Meek's Gregory patch

In 1996 Walton and Meek proposed a new quartic Gregory patch in [13]. Walton and Meek's definition of the patch heavily depends on the construction of the cubic boundary curves $\mathbf{c}_i(t)$, $i = 1, 2, 3$, described in two previous articles [14, 15]. They create a specific tangent ribbon along each boundary curve and then they generate a surface patch with cross-boundary directional derivatives that lie in that plane.

A reasonable candidate for this plane is the one spanned by the derivative of the curve, i.e., the tangent vector

$$\dot{\mathbf{c}}_i(t) = 3\sum_{k=0}^{2} \mathbf{w}_k^i B_k^2(t) \tag{2}$$

and the vector

$$\mathbf{h}_i(t) = \sum_{k=0}^{2} \mathbf{a}_k^i B_k^2(t), \quad 0 \leqslant t \leqslant 1,\ i = 1, 2, 3, \tag{3}$$

where

$$\begin{aligned} \mathbf{a}_0^i &= \mathbf{n}_{i-1} \times \frac{\mathbf{w}_0^i}{\|\mathbf{w}_0^i\|}, \\ \mathbf{a}_2^i &= \mathbf{n}_i \times \frac{\mathbf{w}_2^i}{\|\mathbf{w}_2^i\|}, \\ \mathbf{a}_1^i &= \frac{\mathbf{a}_0^i + \mathbf{a}_2^i}{\|\mathbf{a}_0^i + \mathbf{a}_2^i\|}, \end{aligned} \tag{4}$$

with $\mathbf{n}_3 = \mathbf{n}_0$. See Figure 9 for an example.

A triangular quartic Gregory patch can now be constructed. The control points of the quartic boundary curves $\mathbf{c}_i$ (degree raised from cubic) are used as control points of the patch boundaries. Let the interior control points adjacent to a boundary (e.g. $\mathbf{b}_{112}$ and $\mathbf{b}_{121}$ with respect to the boundary corresponding to $\mathbf{e}_2$) be $\mathbf{g}_{i,1}$ and $\mathbf{g}_{i,2}$, $i = 1, 2, 3$. This implies that each interior control point is determined twice, once for each boundary it is associated with, as shown in Figure 2(c). These points $\mathbf{g}_{i,1}$ and $\mathbf{g}_{i,2}$, $i = 1, 2, 3$, are obtained by requiring that the directional derivatives

$$\mathbf{s}_i^{d_i}(t) = \sum_{k=0}^{3} \widehat{\boldsymbol{\Delta}}_k^i B_k^3(t),\ i = 1, 2, 3,$$

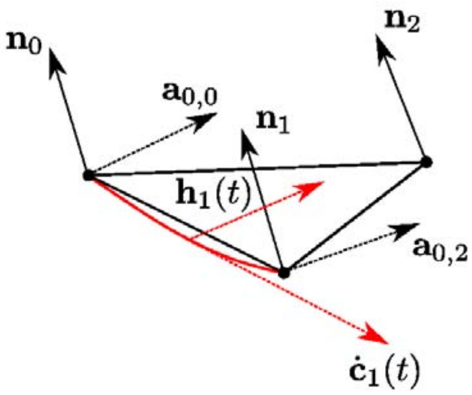


Figure 9: The plane spanned by the tangent vector $\dot{\mathbf{c}}_1(t)$ and the vector $\mathbf{h}_1(t)$.

corresponding to the directions

$$\mathbf{d}_1 = (1, -1/2, -1/2),\ \mathbf{d}_2 = (-1/2, -1/2, 1) \text{ and } \mathbf{d}_3 = (-1/2, 1, -1/2), \quad (5)$$

lie in the tangent ribbon constructed for the corresponding boundary. Namely,

$$\mathbf{s}_i^{d_i}(t) = \frac{1}{3}\alpha_i(t)\dot{\mathbf{c}}_i(t) + \beta_i(t)\mathbf{h}_i(t), \quad i = 1, 2, 3,$$

where $\alpha_i(t)$ and $\beta_i(t)$ are linear polynomials in t.

Once the points $\mathbf{g}_{i,1}$ and $\mathbf{g}_{i,2}$ are obtained, a simple symmetric blending is used to define the three central control points:

$$\mathbf{b}_{211} = \frac{v\mathbf{g}_{1,1} + w\mathbf{g}_{3,2}}{v + w},\ \mathbf{b}_{121} = \frac{u\mathbf{g}_{1,2} + w\mathbf{g}_{2,1}}{u + w},\ \mathbf{b}_{112} = \frac{v\mathbf{g}_{2,2} + u\mathbf{g}_{3,1}}{u + v}. \quad (6)$$

2.4. A new cubic Walton and Meek-like Gregory patch

The study of the three methods presented above inspired us to investigate if it is possible to create a new cubic Gregory patch starting from Walton and Meek's construction. In the following it will be called cubicWM patch to distinguish it from the original quartic patch of Walton and Meek.

Let us consider the cubic patch $\mathbf{s}(u, v, w)$ with boundary curves expressed in cubic Bézier form by $\mathbf{c}_i(t)$, $i = 1, 2, 3$. The derivatives of these curves are quadratic Bézier curves defined by (2). If we want to construct a cubic patch, differently from the quartic patch of Walton and Meek, the directional derivatives $\mathbf{s}_i^{d_i}(t)$ in the directions (5) are quadratic Bézier curves

$$\mathbf{s}_i^{d_i}(t) = \sum_{k=0}^{2} \mathbf{\Delta}_k^i B_k^2(t), \quad i = 1, 2, 3. \tag{7}$$

The control vectors $\mathbf{\Delta}_k^i$ are shown in Figure 10. Explicitly, for the edge $\mathbf{e}_1$, we obtain

$$\begin{aligned}
\mathbf{\Delta}_0^1 &= -\frac{1}{2}\mathbf{b}_{300} - \frac{1}{2}\mathbf{b}_{210} + \mathbf{b}_{201}, \\
\mathbf{\Delta}_1^1 &= -\frac{1}{2}\mathbf{b}_{210} - \frac{1}{2}\mathbf{b}_{120} + \mathbf{b}_{111}, \\
\mathbf{\Delta}_2^1 &= -\frac{1}{2}\mathbf{b}_{120} - \frac{1}{2}\mathbf{b}_{030} + \mathbf{b}_{021}.
\end{aligned}$$

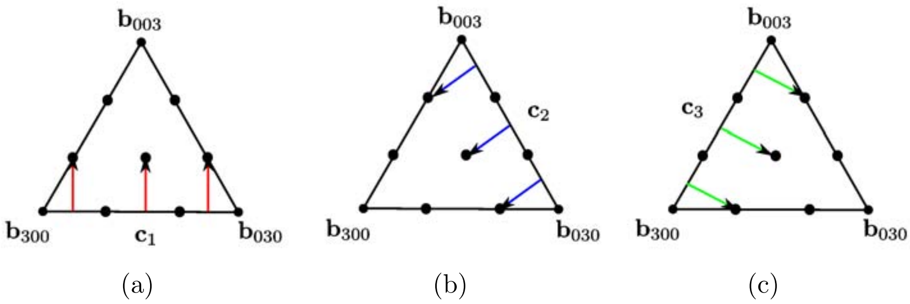


Figure 10: The control vectors $\mathbf{\Delta}_k^i$ for the three directional derivatives $\mathbf{s}_i^{d_i}(t)$, $i = 1, 2, 3$: (a) $\mathbf{\Delta}_0^1$, $\mathbf{\Delta}_1^1$, $\mathbf{\Delta}_2^1$, (b) $\mathbf{\Delta}_0^2$, $\mathbf{\Delta}_1^2$, $\mathbf{\Delta}_2^2$ and (c) $\mathbf{\Delta}_0^3$, $\mathbf{\Delta}_1^3$, $\mathbf{\Delta}_2^3$.

As explained in section 2.3, Walton and Meek's method generates a specific tangent ribbon along each boundary curve. Then, they create a surface patch with cross-boundary directional derivatives that lie in that plane to ensure G^1-continuity with the neighbouring triangles. We choose to define this plane exactly as they proposed, namely as the plane spanned by the tangent vector $\dot{\mathbf{c}}_i(t)$ and the vector $\mathbf{h}_i(t)$ previously defined in equations (2) and (3).

Therefore, the conditions on the final patch control points to ensure G^1-continuity with the neighbouring triangles are as in Walton and Meek's construction

$$\mathbf{s}_i^{d_i}(t) = \frac{1}{3}\alpha_i(t)\dot{\mathbf{c}}_i(t) + \beta_i(t)\mathbf{h}_i(t), \quad i = 1, 2, 3, \tag{8}$$

except that here $\mathbf{s}_i^{d_i}(t)$ are quadratic instead of cubic. In the following, for the sake of simplicity we consider only the condition on the border corresponding to $i = 1$. The same construction can be done on borders corresponding to $i = 2$ and $i = 3$.

The simplest choice for the polynomials $\alpha_1(t)$ and $\beta_1(t)$ are two constants α and β. With this choice, in fact, we obtain three quadratic polynomials in (8). By substitution of their Bézier forms (7), (**??**) and (3) in (8), we compare their control points obtaining three points $\mathbf{b}_{111}^1$, $\mathbf{b}_{111}^2$ and $\mathbf{b}_{111}^3$ to be blended to define the interior control point $\mathbf{b}_{111}$. Unfortunately, this substitution leads to a system of equations which is not always solvable.

Let us therefore consider linear functions $\alpha_1(t) = \alpha_0^1(1-t) + \alpha_1^1 t$ and $\beta_1(t) = \beta_0^1(1-t) + \beta_1^1 t$. This means that (8) becomes

$$\sum_{k=0}^{2} \boldsymbol{\Delta}_k^1 B_k^2(t) = \left(\sum_{k=0}^{1} \alpha_k^1 B_k^1(t)\right)\left(\sum_{k=0}^{2} \mathbf{w}_k^1 B_k^2(t)\right) + \left(\sum_{k=0}^{1} \beta_k^1 B_k^1(t)\right)\left(\sum_{k=0}^{2} \mathbf{a}_k^1 B_k^2(t)\right).$$

By degree elevation of the directional derivative $\mathbf{s}_1^{d_1}(t)$ we obtain cubic polynomials on both sides

$$\sum_{k=0}^{3} \widetilde{\boldsymbol{\Delta}}_k^1 B_k^3(t) = \sum_{k=0}^{1}\sum_{j=0}^{2} \frac{\binom{1}{k}\binom{2}{j}}{\binom{3}{k+j}} (\alpha_k^1 \mathbf{w}_j^1 + \beta_k^1 \mathbf{a}_j^1) B_{k+j}^3(t). \tag{9}$$

By comparing the coefficients of the control points of the cubic polynomials in eq. (9) we obtain the following system of equations

$$\begin{aligned}
\widetilde{\boldsymbol{\Delta}}_0^1 &= \alpha_0^1 \mathbf{w}_0^1 + \beta_0^1 \mathbf{a}_0^1, \\
\widetilde{\boldsymbol{\Delta}}_1^1 &= \frac{2}{3}(\alpha_0^1 \mathbf{w}_1^1 + \beta_0^1 \mathbf{a}_1^1) + \frac{1}{3}(\alpha_1^1 \mathbf{w}_0^1 + \beta_1^1 \mathbf{a}_0^1), \\
\widetilde{\boldsymbol{\Delta}}_2^1 &= \frac{1}{3}(\alpha_0^1 \mathbf{w}_2^1 + \beta_0^1 \mathbf{a}_2^1) + \frac{2}{3}(\alpha_1^1 \mathbf{w}_1^1 + \beta_1^1 \mathbf{a}_1^1), \\
\widetilde{\boldsymbol{\Delta}}_3^1 &= \alpha_1^1 \mathbf{w}_2^1 + \beta_1^1 \mathbf{a}_2^1.
\end{aligned}$$

From the first and the last equation we can compute α_0^1, α_1^1, β_0^1 and β_1^1 as

$$\begin{aligned}
\alpha_0^1 &= \frac{\widetilde{\boldsymbol{\Delta}}_0^1 \cdot \mathbf{w}_0^1}{\mathbf{w}_0^1 \cdot \mathbf{w}_0^1}, \quad \beta_0^1 = \widetilde{\boldsymbol{\Delta}}_0^1 \cdot \mathbf{a}_0^1, \\
\alpha_1^1 &= \frac{\widetilde{\boldsymbol{\Delta}}_2^1 \cdot \mathbf{w}_2^1}{\mathbf{w}_2^1 \cdot \mathbf{w}_2^1}, \quad \beta_1^1 = \widetilde{\boldsymbol{\Delta}}_2^1 \cdot \mathbf{a}_2^1,
\end{aligned}$$

since $\det(\widetilde{\boldsymbol{\Delta}}_0^1, \mathbf{w}_0^1, \mathbf{a}_0^1) = 0$, $\det(\widetilde{\boldsymbol{\Delta}}_3^1, \mathbf{w}_2^1, \mathbf{a}_2^1) = 0$, and $\mathbf{w}_0^1 \cdot \mathbf{a}_0^1 = 0$ (see (4)). Once α_0^1, α_1^1, β_0^1 and β_1^1 are calculated, the two central equations can be used to compute two interior points $\mathbf{b}_{111}^{11}$ and $\mathbf{b}_{111}^{12}$. Repeating this procedure for the three borders we obtain six points $\mathbf{b}_{111}^{11}$, $\mathbf{b}_{111}^{12}$, $\mathbf{b}_{111}^{21}$, $\mathbf{b}_{111}^{22}$, $\mathbf{b}_{111}^{31}$ and $\mathbf{b}_{111}^{32}$ to be blended to define the interior control point $\mathbf{b}_{111}$, as shown in Figure 2(d).

2.4.1. Cubic boundary curves and blending functions

Once cubic boundary curves are constructed, the six points $\mathbf{b}_{111}^{11}$, $\mathbf{b}_{111}^{12}$, $\mathbf{b}_{111}^{21}$, $\mathbf{b}_{111}^{22}$, $\mathbf{b}_{111}^{31}$ and $\mathbf{b}_{111}^{32}$ can be obtained with the procedure described above. These points need to be blended to define the interior control point $\mathbf{b}_{111}(u, v, w)$.

We analysed and compared four different surfaces obtained by using different cubic interpolants for the boundary curves and different blending functions for the central control point. We tested the cubic patch by using the cubic boundary curves proposed in PN triangles [16] and the cubic boundary curves proposed by Walton and Meek for their quartic patch in [14, 15]. As blending functions, instead, we use the PNG1 triangles formula (1), and we define a simpler formula similar to that used by Walton and Meek for the their three interior control points:

$$\mathbf{b}_{111}(u, v, w) = u\left(\frac{v\mathbf{b}_{111}^{11} + w\mathbf{b}_{111}^{32}}{v + w}\right) + v\left(\frac{w\mathbf{b}_{111}^{21} + u\mathbf{b}_{111}^{12}}{w + u}\right) + w\left(\frac{u\mathbf{b}_{111}^{31} + v\mathbf{b}_{111}^{22}}{u + v}\right). \tag{10}$$

To summarise we tested the following four different combinations:

cubicPN-B1: Cubic boundary curves constructed as in PN triangles and blending function defined by (10).

cubicPN-B2: Cubic boundary curves constructed as in PN triangles and blending function from PNG1 triangles (1).

cubicWM-B1: Cubic boundary curves constructed as in Walton and Meek and blending function defined by (10).

cubicWM-B2: Cubic boundary curves constructed as in Walton and Meek and blending function from PNG1 triangles (1).

As already pointed out by Mann et al. in their survey [5], the boundary curves play an important role in the shape quality of the interpolating surface. In the case of the two Gregory patches presented here, in particular, the interior control points heavily depend on the boundary curves. All the tests described in the next section have been applied to these four different combinations. These tests showed us that the use of Walton and Meek boundary curves yields surfaces with better shape quality.

On the contrary, the blending function for the interior control point does not affect the shape of the surface as much as the boundary curves. But, as shown in section 3.1, it deeply affects the computational cost, as far as the normal computation is concerned. Therefore, in the next section we use cubicWM-B1 for comparison with the other methods in order to keep the patch formulation as simple as possible, while the surface quality the best possible at the lowest computational cost. More details on these tests can be found in [17].

3. Comparisons

We implemented all the schemes as an Autodesk Maya® plug-in (*MPx-HwShaderNode*), based on the plug-in from [12]. The Polygons part of Autodesk Maya® is a classic polygonal modeller, and lots of low-level and high-level functions are available for surface creation.

3.1. Computational costs

Before comparing the surface quality of the four schemes, we compare their computational costs. We decided to compute manually the number of scalar additions/subtractions, scalar multiplications and scalar divisions required for the evaluation of the control points $\mathbf{b}_{ijk}(u, v, w)$. In fact, once these control points are computed, the cost for the evaluation of a parametric hybrid patch, a PNG1 triangle and the cubicWM-B1 patch is the same as that of a cubic Bézier triangle, and the evaluation of a Walton and Meek's patch costs as much as the evaluation of a quartic Bézier triangle. Then, to verify these computational costs in practice, we measured the time required for the tessellation on the CPU by using a 1000 triangles Bunny mesh, tessellating every triangle patch into 55 points (tessellation factor $f = 10$), and into 210 points (tessellation factor $f = 20$). In the vertex shader on the GPU, we tessellated the patch into 210 points (tessellation factor $f = 20$), which are handled as OpenGL vertex arrays. As the shading is completely vertex shader-bound, we measured the time for vertex shading and fragment shading together. These measurements were performed in Maya 2008 on a MS

Scheme	Boundary cps			Interior cps			Total		
	add/sub	mult	div	add/sub	mult	div	add/sub	mult	div
Hybrid	36	60	6	4	9	1	40	69	7
PNG1	36	36	6	19	27	4	55	63	10
WM	-	-	-	6	6	3	6	6	3
cubicWM-B1	-	-	-	8	9	3	8	9	3

Table 1: Number of operations required for the evaluation of $\mathbf{b}_{ijk}(u, v, w)$ for each scheme.

Windows 7 (64bit) system with Intel P8700 (2.5 GHz) processor and NVidia Geforce 9600GT (512 MB) mobile graphics with driver version 258.96.

Table 1 shows the number of operations required for the evaluation of the rational blending functions defining the control points for each method. The Gregory patches have the important advantage that only the interior control points are blended. Thus the operations required for the evaluation of the control points in Walton and Meek's patch and in our cubicWM-B1 are considerably reduced with respect to the other two schemes.

In general, the evaluation of a surface point and normal for the quartic patch is more expensive than for a cubic patch, which makes a difference for the scalar CPU implementation (not using SIMD extensions). Surprisingly, our CPU tests in Table 2 show that this is not necessarily the case when considering rational blend schemes. In fact, contrary to our expectations, we obtain that for both tessellation factors, cubicWM-B1 is slightly slower than Walton and Meek's quartic patch. This is due to the fact that here we evaluate the point and the real analytic normal of the patch. Even if the use of a cubic patch, instead of a quartic, allows a faster evaluation of the point on the surface, the more complicated blending function (10) for six points yields more expensive derivative formulas than the simpler blending functions (6) for the quartic patch. On the other hand, on the GPU we obtain that the point-normal evaluation of our cubicWM-B1 patch is faster than that of all the other schemes. Here, the control point computation is performed once on the CPU and is included in the GPU timings. Point and normal evaluations are then performed on the GPU.

The parametric hybrid patch is slower than PNG1 triangles on the CPU, probably because the construction of its control points is more complex, while it is faster on the GPU as its blending functions are simpler than those of PNG1 triangles.

Scheme	CPU				GPU	
	$f = 10$		$f = 20$		$f = 20$	
Hybrid	331ms	3fps	1080ms	0.91fps	38.74ms	25.81fps
PNG1	202ms	4.9fps	730ms	1.37fps	47.14ms	21.21fps
WM	76.9ms	13fps	266ms	3.77fps	22.40ms	44.63fps
cubicWM-B1	83ms	12fps	286ms	3.5fps	19.40ms	51.54fps

Table 2: Time required for the tessellation on the CPU and on the GPU.

Therefore, on the CPU hybrid parametric patch's blending functions are the most expensive, followed by those from PNG1 triangles, cubicWM-B1 and Walton and Meek's patch, while on the GPU cubicWM-B1 performs best, followed by Walton and Meek's patch, hybrid parametric patch and PNG1 triangles.

3.2. Sphere and Torus approximation

In this section we compare the behaviour of the three schemes with respect to a known surface. We compare the signed distance between the analytic surface (a sphere and a torus) and the piecewise parametric interpolants computed by the schemes on a sampling of points from that surface. We are especially interested in the schemes behaviour when refining the base mesh of the piecewise parametric surface.

The base mesh for the sphere is an icosahedron sampled from a sphere of radius $r = 1$ centred in the origin. At any refinement step i, it is refined by means of a 4-split division of the triangles, which results in triangle meshes with $20 \cdot 4^i$ triangles, i.e., 20 for $i = 0$, 80 for $i = 1$, 320 for $i = 2$, 1280 for $i = 3$ and 5120 triangles for $i = 4$.

The base mesh for the torus of radii $r_1 = 1$ and $r_2 = 0.5$ centred in the origin is generated by a subdivision of the bivariate parameter domain $[0, 2\pi) \times [0, 2\pi)$ into j^2 quadrangular regions. After the refinement, the quadrangular mesh is triangulated adding the diagonals. This results in $2 \cdot j^2$ triangles at any refinement step j $(j = 1, 2, 3, \dots)$.

We measure the signed distance between the analytic surface and the piecewise parametric interpolant along the patch normal for the refinement steps $i = 0, 1, 2, 3, 4$, in the case of the sphere, and for $j = 5, 10, 15, 20, 25, 35$, in the case of the torus. Iterations $i = 4$ and $j = 35$, respectively, yield mean distance values close to zero.

Figures 11 and 12 show, respectively, the approximation behaviour of the mean signed distance to the sphere and to the torus. Concerning the sphere,

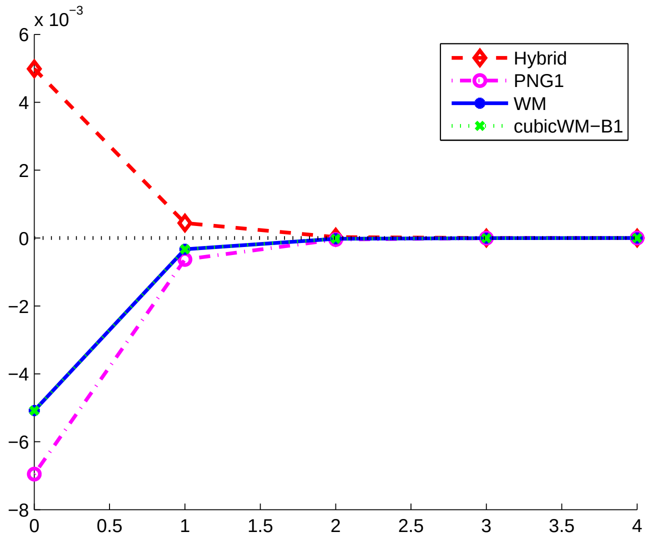


Figure 11: Mean signed distance of sphere interpolation depending on the refinement step i.

the plot in Figure 11 shows that for all four methods the mean distance tends to zero when refining the mesh. While PNG1 triangles, Walton and Meek's patch and cubicWM-B1 approximate the analytic surface remaining always in the interior, hybrid parametric patches have in all refinement steps positive mean distances. This behaviour is confirmed by the statistical data collected in Table 3 where the maximum signed distances for all methods except hybrid parametric patch are zero, especially for steps $i = 0$ and $i = 1$, confirming that, not only the mean distance, but all the distances collected are always negative. CubicWM-B1 and Walton and Meek's patches have always the smallest standard deviations; whereas, PNG1 triangles have always the biggest, except for $i = 0$. All the mean distances for the four methods decrease with the same order of magnitude except for $i = 4$, where PNG1 triangles have the worst mean distance (behaviour confirmed also by the minimum and maximum values). We compare their absolute values for all the steps in Figure 13. Discarding the sign of the distances, cubicWM-B1 and Walton and Meek's patch have the best approximation behaviour, followed by hybrid parametric patch and PNG1 triangles.

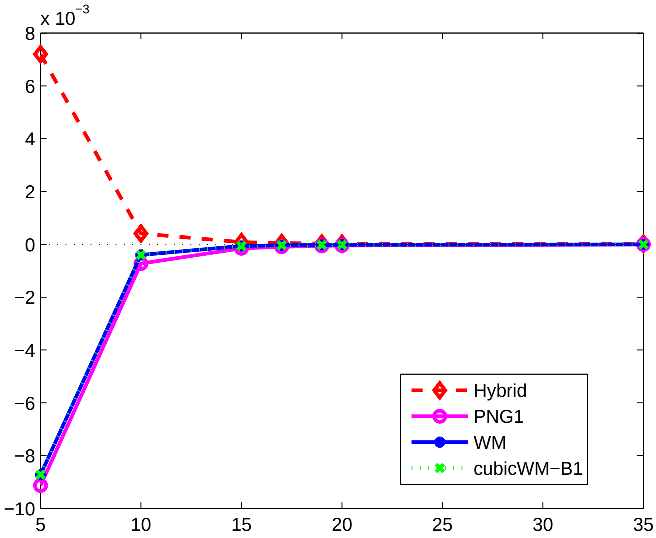


Figure 12: Mean signed distance of torus interpolation depending on the refinement step j.

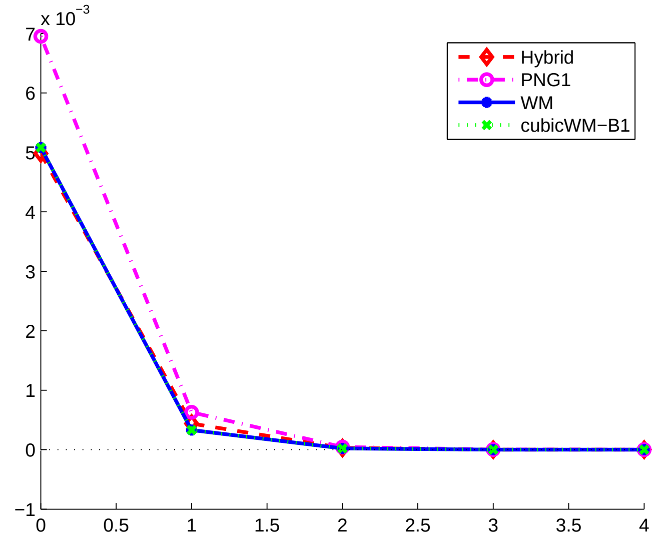


Figure 13: Absolute values of the mean distance of sphere interpolation depending on the refinement step i.

Step	Methods	Min,Max distance	Mean distance ± std. dev.
$i = 0$	Hybrid	[-0.00274014, 0.0131108]	0.00498299 ± 0.00479724
	PNG1	[-0.0111526, 0]	-0.00695174 ± 0.00452694
	WM	[-0.0105214, 0]	-0.00507952 ± 0.00375258
	cubicWM-B1	[-0.0105212, 0]	-0.0050795 ± 0.00375258
$i = 1$	Hybrid	[-0.000161588, 0.00126064]	0.000440331 ± 0.000403261
	PNG1	[-0.00230867, 0]	-0.000629614 ± 0.000615845
	WM	[-0.000817776, 0]	-0.000329054 ± 0.000245696
	cubicWM-B1	[-0.000817716, 0]	-0.000329053 ± 0.000245695
$i = 2$	Hybrid	[-9.35793$\cdot 10^{-6}$, 8.9407$\cdot 10^{-5}$]	2.92289$\cdot 10^{-5}$ ± 2.57138$\cdot 10^{-5}$
	PNG1	[-0.00026983, 8.07047$\cdot 10^{-5}$]	-4.43266$\cdot 10^{-5}$ ± 7.07615$\cdot 10^{-5}$
	WM	[-5.41806$\cdot 10^{-5}$, 0]	-2.04634$\cdot 10^{-5}$ ± 1.51959$\cdot 10^{-5}$
	cubicWM-B1	[-5.4121$\cdot 10^{-5}$, 0]	-2.04574$\cdot 10^{-5}$ ± 1.51922$\cdot 10^{-5}$
$i = 3$	Hybrid	[-7.15256$\cdot 10^{-7}$, 5.84126$\cdot 10^{-6}$]	1.82351$\cdot 10^{-6}$ ± 1.59932$\cdot 10^{-6}$
	PNG1	[-6.02603$\cdot 10^{-5}$, 3.0756$\cdot 10^{-5}$]	-3.00776$\cdot 10^{-6}$ ± 1.02762$\cdot 10^{-5}$
	WM	[-3.45707$\cdot 10^{-6}$, 0]	-1.28954$\cdot 10^{-6}$ ± 9.54655$\cdot 10^{-7}$
	cubicWM-B1	[-3.45707$\cdot 10^{-6}$, 0]	-1.28604$\cdot 10^{-6}$ ± 9.5211$\cdot 10^{-7}$
$i = 4$	Hybrid	[-2.38419$\cdot 10^{-7}$, 5.96046$\cdot 10^{-7}$]	8.79169$\cdot 10^{-8}$ ± 1.11149$\cdot 10^{-7}$
	PNG1	[-1.46627$\cdot 10^{-5}$, 8.10623$\cdot 10^{-6}$]	-2.0942$\cdot 10^{-7}$ ± 1.69522$\cdot 10^{-6}$
	WM	[-4.17233$\cdot 10^{-7}$, 2.38419$\cdot 10^{-7}$]	-9.45127$\cdot 10^{-8}$ ± 9.17229$\cdot 10^{-8}$
	cubicWM-B1	[-3.57628$\cdot 10^{-7}$, 2.38419$\cdot 10^{-7}$]	-9.28638$\cdot 10^{-8}$ ± 8.45791$\cdot 10^{-8}$

Table 3: Statistics of signed distances to the sphere: mean distance with standard deviation (defined as $\sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \overline{x})^2}$, x_k being the distance values) and minimum and maximum distance.

The plot in Figure 12 seems to show the same behaviour for the torus interpolation. But, the statistical data in Table 4 reveal some differences. Here, minimum and maximum values vary between negative and positive values for all the methods; PNG1 triangles, Walton and Meek's patch and cubicWM-B1 patch have always negative mean distances, while parametric hybrid patches always positive. PNG1 triangles have in all the steps the biggest mean distances in absolute value. Except in the first step $j = 5$, cubicWM-B1 and Walton and Meek's distances have the smallest values in absolute value. For the three methods minimum and maximum values show almost the same behaviours, and decrease with the same order of magnitude. Nevertheless parametric hybrid distances vary always in a smaller interval. In fact, parametric hybrid patches have, in general, the smallest standard deviations.

In summary, cubicWM-B1 and Walton and Meek's patch show almost identical behaviours and they perform best, followed by the hybrid parametric patch, whereas PNG1 triangles exhibits the worst approximation behaviour.

3.3. Arbitrary triangle meshes

We now compare the surfaces obtained by the four schemes on arbitrary triangle meshes with a low triangle count, because, in general, the real-world use of these methods concerns this kind of meshes.

By using highlight lines and Gaussian curvature plots [18] we analysed the surfaces generated from the following seven meshes: Sphere, Torus, RoundedCube, Head, Pawn, Bunny and Dinousaur. See Table 5 for statistical information about these meshes. We chose them because they represent arbitrary triangle meshes and also because for some of the schemes they exhibit certain specialities.

We use the exact surface normals $\mathbf{n}(u,v) = \frac{\frac{\partial \mathbf{s}}{\partial u}(u,v)\times\frac{\partial \mathbf{s}}{\partial v}(u,v)}{\left\|\frac{\partial \mathbf{s}}{\partial u}(u,v)\times\frac{\partial \mathbf{s}}{\partial v}(u,v)\right\|}$ for computing the Gaussian curvature. Table 6 contains the minimum, maximum and mean values of the Gaussian curvature computed on a dense sampling grid of 210 points per patch, with standard deviation defined as $\sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \overline{x})^2}$. Since the second order derivatives do not exist in the vertices of the patch, these data are not taken into account in the statistics and are plotted in black in the figures. A common scale is used to compare the curvature plots of the four methods and the maximum and minimum values of each of them are converted to that scale.

Step	Methods	Min,Max distance	Mean distance ± std. dev.
$j=5$	Hybrid	[-0.0227294, 0.0478425]	0.00720392 ± 0.015833
	PNG1	[-0.158901, 0.044207]	-0.00913514 ± 0.0326367
	WM	[-0.0977793, 0.0493978]	-0.00872526 ± 0.0304518
	cubicWM-B1	[-0.0977793, 0.0493978]	-0.00872526 ± 0.0304518
$j=10$	Hybrid	[-0.00262406, 0.00356019]	0.000410239 ± 0.000964081
	PNG1	[-0.00952291, 0.00411868]	-0.000726871 ± 0.00221538
	WM	[-0.00544271, 0.00396627]	-0.000399186 ± 0.00178612
	cubicWM-B1	[-0.00544268, 0.00396627]	-0.000399186 ± 0.00178612
$j=15$	Hybrid	[-0.000710934, 0.000884116]	$8.30311{\cdot}10^{-5}$ ± 0.000249906
	PNG1	[-0.00201935, 0.00107235]	-0.000149719 ± 0.000488205
	WM	[-0.00117564, 0.00107831]	$-6.02477{\cdot}10^{-5}$ ± 0.000400663
	cubicWM-B1	[-0.00117561, 0.00107831]	$-6.02466{\cdot}10^{-5}$ ± 0.000400662
$j=17$	Hybrid	[-0.000484645, 0.0006001]	$5.06274{\cdot}10^{-5}$ ± 0.000167361
	PNG1	[-0.00122964, 0.000724077]	$-9.13885{\cdot}10^{-5}$ ± 0.000309585
	WM	[-0.000749379, 0.000720561]	$-3.39081{\cdot}10^{-5}$ ± 0.000259712
	cubicWM-B1	[-0.000749409, 0.000720561]	$-3.39079{\cdot}10^{-5}$ ± 0.000259712
$j=19$	Hybrid	[-0.000337839, 0.000417173]	$3.25315{\cdot}10^{-5}$ ± 0.000117689
	PNG1	[-0.000791073, 0.000496805]	$-5.89122{\cdot}10^{-5}$ ± 0.00020772
	WM	[-0.000505209, 0.000505209]	$-2.05269{\cdot}10^{-5}$ ± 0.000178293
	cubicWM-B1	[-0.000505149, 0.000505209]	$-2.05266{\cdot}10^{-5}$ ± 0.000178293
$j=20$	Hybrid	[-0.000292093, 0.000354946]	$2.6755{\cdot}10^{-5}$ ± 0.000100177
	PNG1	[-0.000647008, 0.000430048]	$-4.78646{\cdot}10^{-5}$ ± 0.00017302
	WM	[-0.000424266, 0.000427842]	$-1.60885{\cdot}10^{-5}$ ± 0.000150207
	cubicWM-B1	[-0.000424266, 0.000427842]	$-1.60887{\cdot}10^{-5}$ ± 0.000150205
$j=35$	Hybrid	[$-5.30481{\cdot}10^{-5}$, $5.91278{\cdot}10^{-5}$	$2.66562{\cdot}10^{-6}$ ± $1.78022{\cdot}10^{-5}$
	PNG1	[$-8.57115{\cdot}10^{-5}$, $7.31945{\cdot}10^{-5}$]	$-5.37519{\cdot}10^{-6}$ ± $2.57699{\cdot}10^{-5}$
	WM	[$-6.89626{\cdot}10^{-5}$, $7.31945{\cdot}10^{-5}$]	$-1.6294{\cdot}10^{-6}$ ± $2.50909{\cdot}10^{-5}$
	cubicWM-B1	[$-6.89328{\cdot}10^{-5}$, $7.31945{\cdot}10^{-5}$]	$-1.62901{\cdot}10^{-6}$ ± $2.50907{\cdot}10^{-5}$

Table 4: Statistics of signed distances to the torus: mean distance with standard deviation (defined as $\sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k-\overline{x})^2}$, x_k being the distance values) and minimum and maximum distance.

Name	#V/#E/#T	Mean angle normals ± std. dev.	Min,max angle normals	#E convex/concave/inflection
 Sphere	12/30/20	0.794654 ± 0.794654	$[0.794654, 2.04327 \cdot 10^{-8}]$	20/0/10
 Torus	25/75/50	0.726595 ± 0.0808356	$[0.620812, 0.826384]$	32/7/35
 RCube	30/84/56	0.812907 ± 0.136409	$[0.688079, 1]$	42/1/40
 Head	102/300/200	0.8214 ± 0.186177	$[-0.0221268, 0.999741]$	106/30/163
 Pawn	154/456/304	0.801564 ± 0.236914	$[-0.115779, 0.999559]$	144/32/280
 Bunny	502/1500/1000	0.911182 ± 0.113563	$[-0.147974, 0.999985]$	556/71/873
 Dinosaur	927/2775/1850	0.935497 ± 0.0966668	$[-0.625247, 0.999996]$	1096/205/1473

Table 5: Statistics of triangle meshes: number of vertices/edges/triangles, angle cosine between vertex and triangle normals (mean ± standard deviation, minimum and maximum), number of convex, concave, inflection edges.

Mesh	**Methods**	**Min,Max curv**	**Mean curv $\pm$ std. dev.**
Sphere	Hybrid	[0.327537, 1.83937]	0.901277 $\pm$ 0.414196
	PNG1	[0.807412, 1.45757]	1.04739 $\pm$ 0.131043
	WM	[0.768366, 1.20612]	0.971769 $\pm$ 0.123814
	cubicWM-B1	[0.768408, 1.1719]	0.971776 $\pm$ 0.123817
Torus	Hybrid	[-820.956, 63.2143]	-2.50575 ± 29.3715
	PNG1	[-8.33147, 8.23699]	-0.585947 $\pm$ 2.33862
	WM	[-9.89538, 10.9907]	-0.61217 $\pm$ 2.58268
	cubicWM-B1	[-9.8852, 10.9848]	-0.61218 $\pm$ 2.58269
Cube	Hybrid	[-0.422122, 0.150237]	-0.0479029 $\pm$ 0.123352
	PNG1	[-3.08765, 5.62103]	1.26259 $\pm$ 1.76093
	WM	[-9.90188, 13.1909]	1.49746 $\pm$ 3.27045
	cubicWM-B1	[-9.80683, 13.1662]	1.50137 $\pm$ 3.27251
Head	Hybrid	[-11347, 31374.6]	0.63273 $\pm$ 204.498
	PNG1	[-266.758, 245.246]	0.367094 $\pm$ 6.93173
	WM	[-5327, 19960]	2.04325 $\pm$ 141.829
	cubicWM-B1	[-5173.04, 19957.7]	2.04977 $\pm$ 141.381
Pawn	Hybrid	[-205.12, 5.76324	-0.0720587 $\pm$ 3.05287
	PNG1	[-20.7914, 78.5121]	0.0699347 $\pm$ 1.59282
	WM	[-618.648, 964.635]	0.143624 $\pm$ 18.6824
	cubicWM-B1	[-543.238, 969.831]	0.152935 $\pm$ 18.5275
Bunny	Hybrid	[-998.402, 239.024]	-0.00863217 $\pm$ 3.26164
	PNG1	[-11.0328, 28.9473]	0.0112989 $\pm$ 0.296384
	WM	[-102.571, 2954.82]	0.0485715 $\pm$ 7.87631
	cubicWM-B1	[-102.584, 2951.82]	0.0486907 $\pm$ 7.8803
Dinosaur	Hybrid	[-144.925, 94.1432	-0.00141404 $\pm$ 0.487428
	PNG1	[-18.138, 287.879]	0.0042961 $\pm$ 0.65595
	WM	[-1136.1, 195.25]	0.000524591 $\pm$ 2.06327
	cubicWM-B1	[-7.13831, 119.032]	0.00372546 $\pm$ 0.298818

Table 6: Statistics on Gaussian curvature. The mean value for Gaussian curvature (mean $\pm$ standard deviation) and the minimum and maximum value measured from the surfaces.

Comparing these statistics for all the meshes, we found that the mean Gaussian curvature is negative for the hybrid parametric patch, except for the Sphere and the Head mesh, while for the other three methods the mean curvature is positive, with again the Torus as exception. Walton and Meek's, cubicWM and parametric hybrid patches have, especially in the Torus and Head mesh, high standard deviations, revealing a more accentuated variation of the curvature with respect to minimum and maximum values. It is surprising that the stability problem of hybrid parametric patches, shown in the following, does not highly affect the curvature values in the other meshes. Again cubicWM-B1 behaves similarly to Walton and Meek's patch, except in the Dinosaur mesh where the curvature does not vary as much as for Walton and Meek's patch since minimum, maximum and standard deviation values are considerably lower.

To show the behaviour of the four schemes on well known shapes, we first graphically analyse the sphere and the torus studied in the previous section with $i = 0$ and $j = 5$, respectively. Although according to Table 6 the four methods seem to be faithful to the analytic shape, their plots in Figure 14 and Figure 15 show several differences. In the first line, by comparing the shaded surfaces we find a more oscillating surface for the hybrid parametric method (in particular by looking at the silhouettes) and this behaviour is confirmed by the highlight lines and the curvature plots. More precisely, curvature plots in the sphere reveal that Walton and Meek's and cubicWM-B1 surfaces better simulate the behaviour of a real sphere, while the other two methods exhibit higher curvature variations near the borders of the patches. On the other hand, for the curvature plots of the Torus all four methods present high curvature variations in the regions where zero curvature is expected, but again highlight lines and curvature plots show that the hybrid parametric surface is the worst.

As the sphere and the torus, the RoundedCube mesh (Figure 16) and the Head mesh have a small triangle count and a quite high number of inflection edges. As expected we found the same behaviours observed in the sphere and the torus. The surface constructed by the parametric hybrid patches is very wavy, and the statistics confirm this behaviour. On the other hand, the curvature statistics and the highlight lines show that PNG1 triangles yield the surface with the best appearance since the maximum and minimum values are in a smaller range. Besides, although all the surfaces are G^1 continuous, the PNG1 triangles RoundedCube gives the visual impression to be smoother than Walton and Meek's and cubicWM-B1 surfaces.

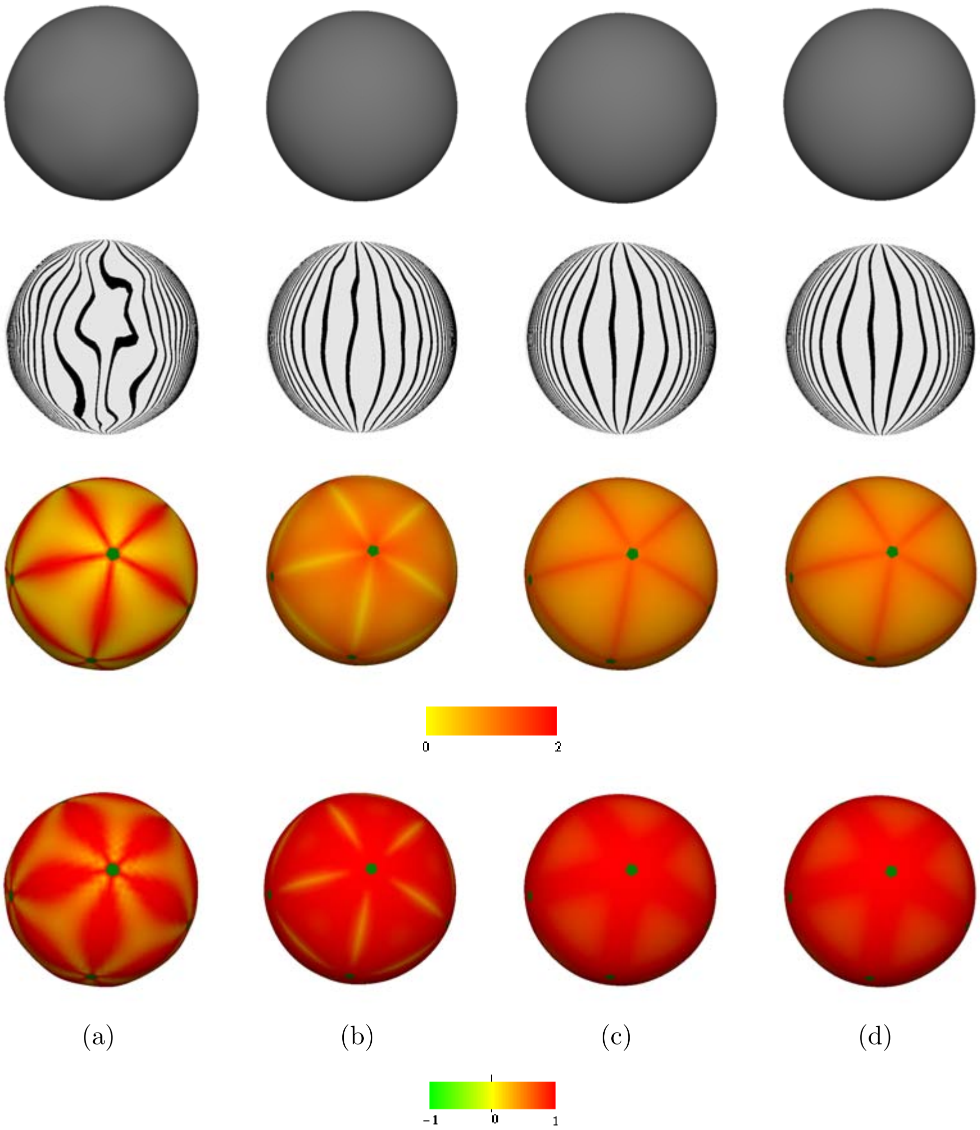


Figure 14: Sphere $i = 0$: surfaces obtained from (a) hybrid parametric patch, (b) PNG1 triangles, (c) Walton and Meek's quartic patch and (d) cubicWM-B1 patch. First row: shaded surfaces; second row: highlight lines; third and fourth row: Gaussian curvature plots with different scale.

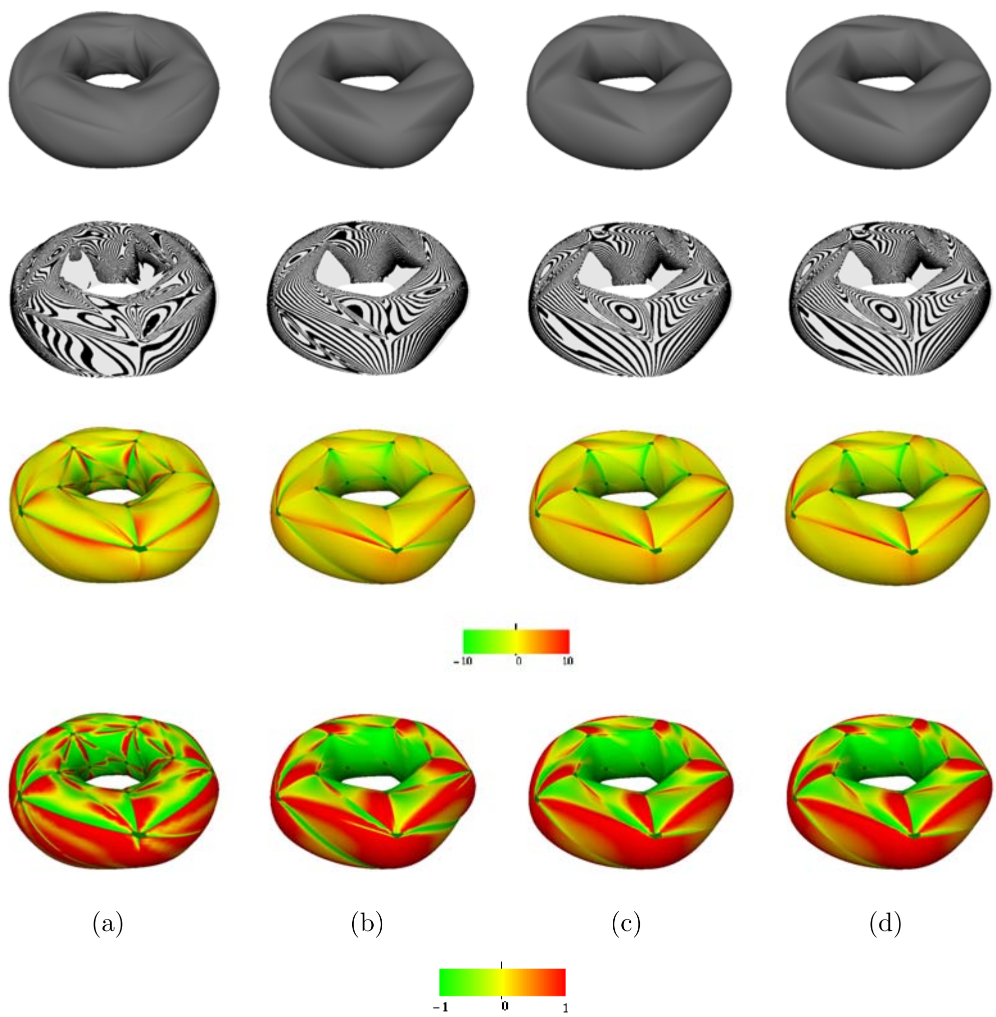


Figure 15: Torus $j = 5$: surfaces obtained from (a) hybrid parametric patch, (b) PNG1 triangles, (c) Walton and Meek's quartic patch and (d) cubicWM-B1 patch. First row: shaded surfaces; second row: highlight lines; third and fourth row: Gaussian curvature plots with different scale.

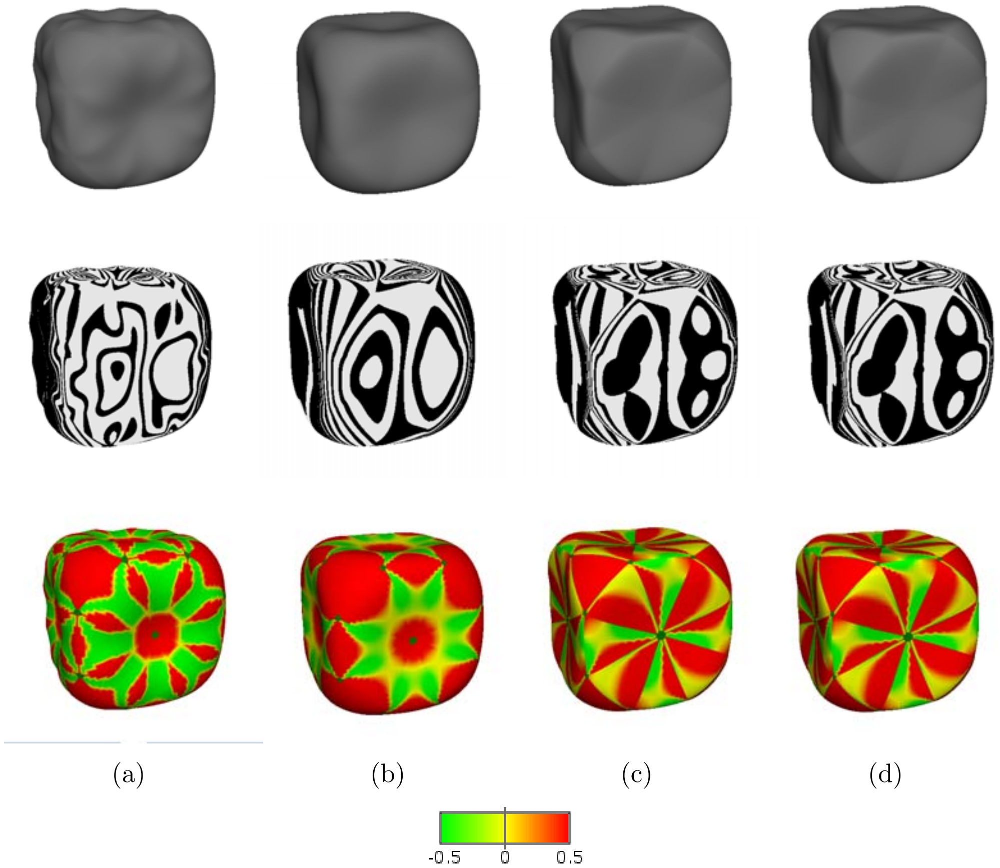
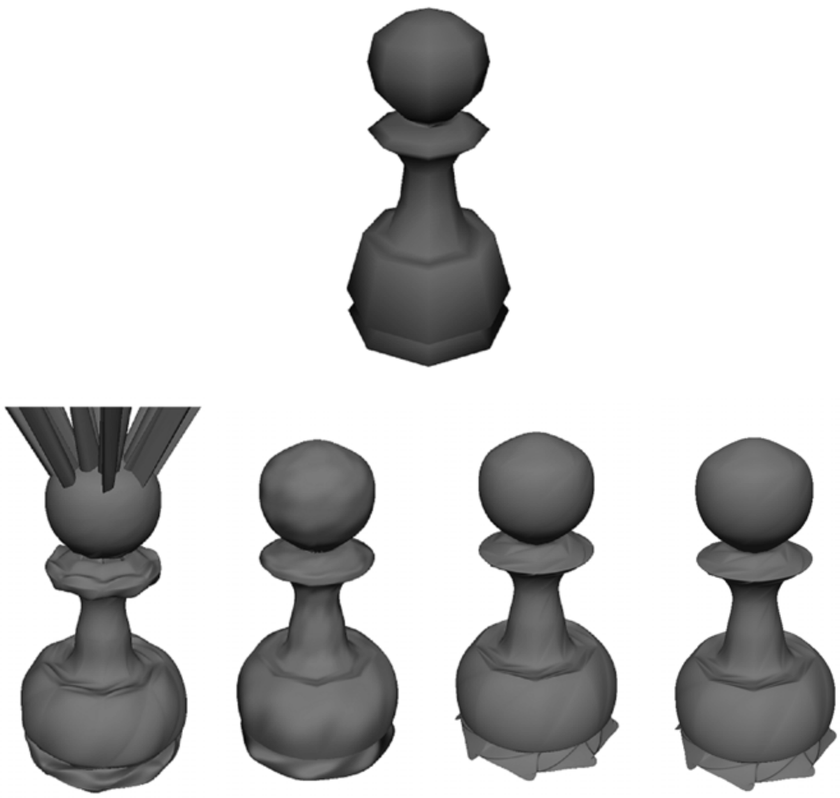


Figure 16: RoundedCube: surfaces obtained from (a) hybrid parametric patch, (b) PNG1 triangles, (c) Walton and Meek's quartic patch and (d) cubicWM-B1 patch. First row: shaded surfaces; second row: highlight lines; third row: Gaussian curvature.

When we consider meshes with a higher number of faces, as for example the Pawn mesh, the stability problem of the parametric hybrid patch related to the choice of the plane on which the patch pairs are projected, becomes evident (Figure 17). In this mesh, in particular, the chosen projection plane is unstable because the normal given in a vertex is perpendicular to the normal defined by the triangle plane and this yields a zero denominator in the definition of the control points. For a more detailed discussion on the possible choices of the projection plane and their consequences we refer the reader to [10]. By comparing the resulting surfaces of the other three methods in Figure 18, we notice that they also present artifacts, even if minor when compared to the parametric hybrid surface.

Figure 17: The parametric hybrid patch reveals stability problems when applied to the Pawn mesh.

Finally, when the four methods are applied to meshes with a higher triangle count, as, for example, Bunny and Dinosaur (respectively 1000 and 1850 faces), they behave differently. On one hand, in Bunny (Figure 19) PNG1 triangles seems to visually produce the smoothest surface and this is confirmed by the highlight lines, where those of cubicWM-B1, Walton and Meek and parametric hybrid surfaces appear more discontinuous and fragmented, and by the statistics on the curvature. Surprisingly, the statistics on the curvature of parametric hybrid patch are not heavily affected from the stability problem that can be easily remarked on the ear of the Bunny. The curvature value is, in fact, on average lower than the values of all the other



(a) (b) (c) (d)

Figure 18: Pawn in columns from left to right: (a) parametric hybrid patch, (b) PNG1 triangles, (c) Walton and Meek's Gregory patch and (d) cubicWM-B1 patch. First row: shaded planar mesh; second row: shaded surfaces.

tested meshes, although minimum and maximum values for all the methods, except PNG1 triangles, are high. This means that the curvature strongly deviates from the mean value. In particular, for Walton and Meek's and cubicWM-B1 patches standard deviations are extremely large. On the other hand, Dinosaur's curvature values are surprisingly much lower than those of all the other meshes and the standard deviation values are more acceptable, except for Walton and Meek's patch. A reasonable explanation could be that the parametric hybrid patch on this mesh exhibits less triangles with stability problems than the previous meshes, resulting in better curvature statistics. No more differences can be seen from the shaded surfaces, the highlight lines and the curvature plots, even with a close up on the details.

4. Conclusions

In this article we made a comparison of local parametric G^1 interpolatory schemes that use rational blends to bypass the vertex consistency problem in the construction of the surface. The main emphasis of this comparison is on the computational costs of the different schemes available, as well as on the surface quality, investigated by using well known methods of surface interrogation as highlight lines and Gaussian curvature plots. The comparison includes four different schemes based on triangular Bézier patches: hybrid parametric patch and PNG1 triangles of degree 3, Walton and Meek's Gregory patch of degree 4. The fourth cubicWM-B1 scheme is a cubic Gregory patch that we proposed inspired by Walton and Meek's construction.

The study on the number of operations required to evaluate the control points reveals that Walton and Meek's patch, and consequently cubicWM-B1 patch, have the important advantage that only the interior control points are blended. Furthermore, PNG1 triangles are also penalised by the more complicated blending function for the interior control point. In practice, we verified this assumption by measurements on the time required for the tessellation of the patches on the CPU and on the GPU. Both on the CPU and on the GPU, cubicWM-B1 and Walton and Meek's patches perform best, where Walton and Meek's patch is faster on the CPU and cubicWM patch is faster on the GPU.

When analysing the surfaces constructed by the four schemes with respect to a sphere and a torus, the statistics show that Walton and Meek's and cubicWM-B1 patches have the best approximation behaviour, followed by parametric hybrid patches and PNG1 triangles.

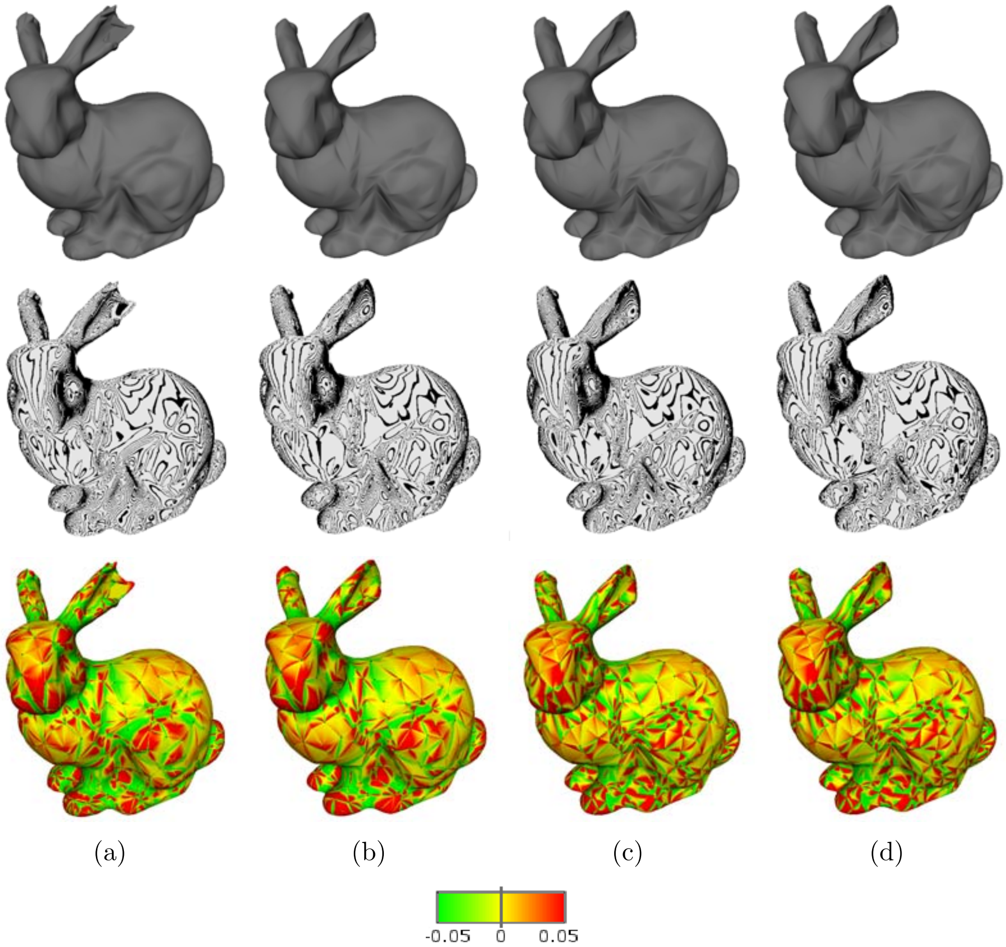


Figure 19: Bunny: surfaces obtained from (a) hybrid parametric patch, (b) PNG1 triangles, (c) Walton and Meek's quartic patch and (d) cubicWM-B1 patch. First row: shaded surfaces; second row: highlight lines; third row: Gaussian curvature.

On the contrary, on arbitrary triangle meshes PNG1 triangles give in general the surfaces with the best appearance. Their statistics, indeed, show that their curvature values vary more regularly. Besides, when we increase the number of faces the stability problem of the parametric hybrid patch related to the choice of the plane on which the patch pairs are projected becomes evident. Unfortunately, this fact makes this method practically unusable on meshes with completely arbitrary normals. In many arbitrary triangle meshes analysed, Walton and Meek's and cubicWM-B1 surfaces seem to suffer in a certain sense from flatness of their boundary curves, as the high standard deviation values in the statistics on the curvature confirm.

From all our tests we can assert that our cubicWM-B1 patch attains almost identical surfaces to the quartic original Walton and Meek's patch, with lower computational costs on the GPU. In particular, the behaviour of our cubic version is slightly better in sphere and torus approximation, as described in section 3.2.

We additionally remark the following important property of our cubicWM-B1 patch and Walton and Meek's Gregory patch. Differently from the other two methods, they do not directly use the triangle neighbour in their construction, since the interior control points are constructed by means of tangent ribbons that depend only on the boundary curves. This is important in some applications, as, for example, in computer games, where usually stored neighbourhood information is not available.

All the results of these tests gave us several subjects and suggestions for future work. First, we want to improve our cubicWM-B1 patch by investigating if the use of different cubic boundary curves can yield surfaces that do not suffer from flatness on arbitrary meshes. Second, we believe that also other choices of the function $\mathbf{h}_i(t)$ used to define the plane for the tangent ribbons could be of interest for further investigation.

5. Acknowledgements

Maria Boschiroli acknowledges the financial support in form of travel allowances from French-Italian University within the Da Vinci program (http://www.universite-franco-italienne.org/). Christoph Fünfzig acknowledges support of the Conseil Régional de Bourgogne in a postdoc fellowship for the 2009/2010 academic year as well as a one month invitation by the LAMAV laboratory at University of Valenciennes, and a three month invitation of the University of Valenciennes (IUT) during the 2010/2011 academic year. Lucia Romani acknowledges a one month invitation from University of Valenciennes during the 2010/2011 academic year.

References

[1] M. Boschiroli, C. Fünfzig, L. Romani, G. Albrecht, A comparison of local parametric C^0 Bézier interpolants for triangular meshes, Computers&Graphics 35 (2011) 20–34.

[2] G. Farin, Curves and surfaces for CAGD: a practical guide, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition, 2002.

[3] J. Peters, Smooth interpolation of a mesh of curves, Constr. Approx. 7 (1991) 221–246.

[4] M. A. Watkins, Problems in geometric continuity, Comput. Aided Geom. Des. 20 (1988) 499–502.

[5] S. Mann, C. Loop, M. Lounsbery, D. Meyers, J. Painter, T. DeRose, K. Sloan, A survey of parametric scattered data fitting using triangular interpolants, in: H. Hagen (Ed.), Curve and surface design, SIAM, 1992, pp. 145–172.

[6] L. A. Shirman, C. H. Séquin, Local surface interpolation with Bézier patches, Computer Aided Geometric Design (1987) 279–295.

[7] G. M. Nielson, A transfinite, visually continuous, triangular interpolant, in: G. Farin (Ed.), Geometric modeling, SIAM, Philadelphia, PA, 1987, pp. 235–246.

[8] J. A. Gregory, Smooth interpolation without twist constraints, in: R. Barnhill, R. Riesenfeld (Eds.), Computer aided geometric design, Academic Press, New York, 1974, pp. 71–87.

[9] S. Mann, M. Davidchuk, A parametric hybrid triangular Bézier patch, in: Proceedings of the international conference on Mathematical methods for curves and surfaces II Lillehammer, 1997, Vanderbilt University, Nashville, TN, USA, 1998, pp. 335–342.

[10] M. F. Davidchuk, A parametric hybrid triangular Bézier patch, Master's thesis, University of Waterloo, 1997.

[11] T. A. Foley, K. Opitz, Hybrid cubic Bézier triangle patches, in: T. Lyche, L. L. Schumaker (Eds.), Mathematical methods in computer aided geometric design II, Academic Press Professional, Inc., San Diego, CA, USA, 1992, pp. 275–286.

[12] C. Fünfzig, K. Müller, D. Hansford, G. Farin, PNG1 triangles for tangent plane continuous surfaces on the GPU, in: GI '08: Proceedings of graphics interface 2008, Canadian Information Processing Society, Toronto, Ont., Canada, 2008, pp. 219–226.

[13] D. J. Walton, D. S. Meek, A triangular G^1 patch from boundary curves, Computer-Aided Design 28 (1996) 113–123.

[14] D. J. Walton, D. S. Meek, Point normal interpolation for stereolithography modelling, Computers & Graphics 19 (1995) 345–353.

[15] D. J. Walton, M. Yeung, Geometric modelling from CT scans for stereolithography apparatus, in: Z.Tang (Ed.), New Advances in CAD&Computer Graphics (Proc. CAD/Graphics '93), pp. 417–422.

[16] A. Vlachos, J. Peters, C. Boyd, J. L. Mitchell, Curved PN triangles, in: I3D '01: Proceedings of the 2001 symposium on Interactive 3D graphics, ACM, New York, NY, USA, 2001, pp. 159–166.

[17] M. Boschiroli, Local Parametric Bézier Interpolants for Triangular Meshes: from Polynomial to Rational Schemes, Ph.D. thesis, 2011. In preparation.

[18] S. Hahmann, A. Belyaev, L. Buse, G. Elber, B. Mourrain, C. Roessl, Shape interrogation - a state of the art, in: L. De Floriani, M. Spagnuolo (Eds.), Shape Analysis and Structuring, Springer, 2008, pp. 1–52.