Non-Distorted Texture Mapping For Sheared Triangulated Meshes

Bruno Lévy* Jean-Laurent Mallet†

GOCAD
ENSG, rue du doyen Marcel Roubeault, 54500 Vandoeuvre

Abstract

This article introduces new techniques for non-distorted texture mapping on complex triangulated meshes. Texture coordinates are assigned to the vertices of the triangulation by using an iterative optimization algorithm, honoring a set of constraints minimizing the distortions. As compared to other global optimization techniques, our method allows the user to specify the surface zones where distortions should be minimized in order of preference. The modular approach described in this paper results in a highly flexible method, facilitating a customized mapping construction. For instance, it is easy to align the texture on the surface with a set of user defined isoparametric curves. Moreover, the mapping can be made continuous through cuts, allowing to parametrize in one go complex cut surfaces. It is easy to specify other constraints to be honored by the so-constructed mappings, as soon as they can be expressed by linear (or linearizable) relations. This method has been integrated successfully within a widely used C.A.D. software dedicated to geosciences. In this context, applications of the method comprise numerical computations of physical properties stored in fine grids within texture space, unfolding geological layers and generating grids that are suitable for finite element analysis. The impact of the method could be also important for 3D paint systems.

CR Categories: I.3.3 [Computer Graphics] Picture/Image Generation ; I.3.5 [Computer Graphics]: Three-Dimenstional Graphics and Realism—Color, shading, shadowing and texture; I.4.3 [Image processing]: Enhancement—Geometric Correction, Texture

Keywords: Non Distorted Texture Mapping, Parametrization, Discrete Smooth Interpolation, Optimization

1 INTRODUCTION

Texture mapping is widely used to improve the visual richness of 3D surfaces in computer generated images. Each 3D surface is put in correspondence with a planar image through a function called a *mapping*. Such a *mapping* assigns a pair of coordinates (u, v) referring to a pixel of the planar image to each point of a surface. Thus, for instance, the latitude and longitude can define a trivial mapping of a sphere. This technique was introduced by Catmull in [Cat74], and first applied to bicubic patches using a recursive subdivision algorithm. Unfortunately, these methods often produce highly distorted textures in the resulting images.

First attempts to minimize these distortions were made in [Pea85] and in [BS86] by separating the process into two steps. The texture pattern is first applied to a simple intermediate surface such as a box or a cylinder for which texture mapping is trivial. Then, this intermediate surface is projected on the target object. The choice of the intermediate surface, its orientation together with the projection method dramatically affect the results, and great deal of user interaction is therefore required.

Another idea is to consider that assigning texture coordinates to any surface is equivalent to flattening it. Such a technique is described in [SMSW86]. The idea consists in unfolding a polygonal surface from a user selected seed. A similar idea has been developed in [BVI91]. This latter method means a parametric surface may be unfolded by allowing cuts to appear on the mapped texture when the discrepancy of the geodesic curvature goes beyond a given threshold.

Minimizing the distortions induced by texture mapping can be also realized using optimization techniques. In the method proposed in [ML88], a mapping of any surface is constructed by starting from a grid of points sampled on the surface. The grid is then iteratively optimized by minimizing a global distortion criterion. Krishnamurthy proposes in [KL96] a similar approach for converting a triangulated mesh into a set of B-Spline surfaces. It is also possible to construct a mapping by assigning (u, v) coordinates to the vertices of the mesh. This naturally leads to the use of harmonic maps, as described in [ERDH95]. This method consists in minimizing a *metric dispersion* criterion. Unfortunately, this does not always preserve angles accurately. Another approach is introduced in [Flo97], generalizing the *barycentric mapping* method introduced in [Tut60]. The (u, v) texture coordinates are found to be the solution of a linear system, where each (u, v) point is a convex combination of its neighbors. Floater[Flo97] proposes a way of choosing the coefficients of these convex combinations to mimic the chord length parametrization for curves. These global methods give good results for most surfaces, but suffer from several limitations when applied to complex surfaces. For these kinds of methods, since the criterion to be minimized is hardwired in the optimization algorithm, it is often difficult to take into account user defined information. For instance, as most surfaces are not developable, distortions will still remain, and the user may want to specify the distribution of these distortions.

This article proposes a new global optimization method. As compared to other similar techniques, the method is based on a modular approach enabling the way the mapping is constructed to be customized. For instance, it is possible to tune the perpendicularity and homogeneous spacing of isoparametric curves all over the surface, thus specifying the surface zones where distortions should be minimized in order of preference. It is also possible to make the mapping respect a set of user specified isoparametric

*levy@ensg.u-nancy.fr, GOCAD and INRIA Lorraine/CNRS
†mallet@ensg.u-nancy.fr, director of the GOCAD consortium

curves. Moreover, the mapping can be made continuous through cuts, hence allowing the mapping of a texture on a complex cut surface in one go. The method can be extended easily to honor other kind of linear constraints. All these constraints allow the method to take into account user specified information while beeing much more automatic than other interactive mapping methods [Ped95, LM94, MYV93], where the parametrization is partially or completely defined by the user.

The first section of the paper summarizes the notions involved in texture mapping on triangulated meshes, and shows how a mapping can be constructed using an iterative optimization algorithm. In Section 2, the criteria to be met in constructing a non-distorting mapping are introduced. These criteria are expressed as a set of linear constraints in Section 3, where the algorithm previously introduced is modified in order to honor them. This results in a general algorithm that can be extended easily to take into account user specified information, as shown in Section 4. In this latter section, we show how to respect user specified isoparametric curves, and how to make mappings continuous through cuts. Section 5 presents some applications and results. The paper concludes with some suggestions for future developments.

2 PARAMETRIZING A TRIANGULATION

In this section, the notion of mapping function defined on a triangulated mesh is recalled, and a new method for constructing such mapping functions is described, based on an iterative optimization algorithm. How these mappings can be optimized in order to minimize the distortions is then explained in Section 3.

2.1 Mapping Function Φ and Discrete Mapping φ

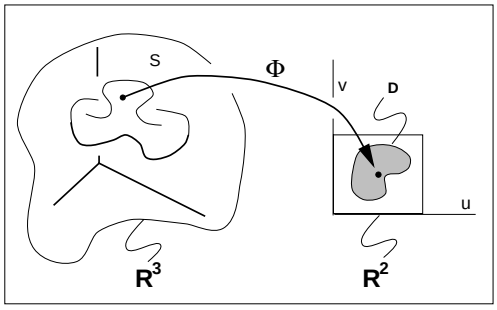


Figure 1: Mapping Φ from a surface $\mathbf{S}$ of $\mathbf{R}^3$ to $\mathcal{D} \subset \mathbf{R}^2$.

As shown in Figure 1, given an open surface $\mathbf{S}$ of $\mathbf{R}^3$, a *mapping* Φ is a one-to-one transform that maps the surface $\mathbf{S}$ to a subset $\mathcal{D}$ of $\mathbf{R}^2$.

$$(x, y, z) \in \mathbf{S} \quad \rightarrow \quad \Phi(x, y, z) = \begin{bmatrix} \Phi^u(x, y, z) \\ \\ \Phi^v(x, y, z) \end{bmatrix}$$

Regarding a mapping, the following definitions can be given:

- $\mathcal{D}$ is called the *parametric* (u, v) *domain*.
- As Φ is, by definition, a one-to-one function, it has an inverse function $\mathbf{x} = \Phi^{-1}$, called a *parametrization* of the surface:

$$(u, v) \in \mathcal{D} \quad \rightarrow \quad \mathbf{x}(u, v) = \Phi^{-1}(u, v) = \begin{bmatrix} x(u, v) \\ y(u, v) \\ z(u, v) \end{bmatrix}$$

If a surface has a parametrization $\mathbf{x}$ defined, the inverse $\Phi = \mathbf{x}^{-1}$ of this parametrization naturally provides a mapping function. Catmull applied this technique to cubic splines in [Cat74], in which a recursive subdivision scheme is described, making it possible to avoid inverting of the parametrization directly.

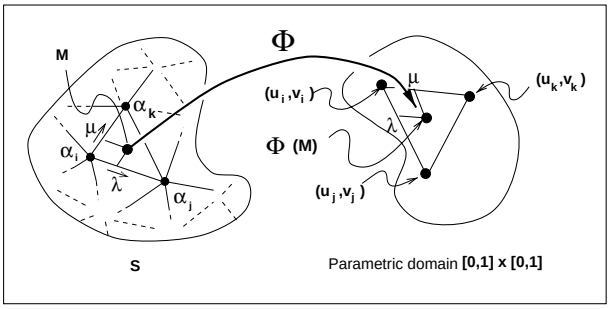


Figure 2: Mapping function Φ interpolated over a triangle.

In what follows, the surface $\mathbf{S}$ is provided with a triangulation $\mathcal{G} = \{\Omega, \mathcal{T}\}$, where Ω is the set of the vertices of the triangulation, and $\mathcal{T}$ the set of the triangles of $\mathcal{G}$, defined as vertex triplets. For the sake of simplicity, Ω will be identified with the interval $[1 \ldots M]$ of integers, where $M = |\Omega|$ denotes the number of vertices of the triangulation. The geometric location at a vertex $\alpha \in \Omega$ is denoted $\mathbf{p}(\alpha)$ in what follows.

For this kind of surface, it is natural to define the value of Φ at the vertices Ω of the triangulation $\mathcal{G}$. This information can be stored as a set of (u_i, v_i) values, where $1 \leq i \leq M$. How to choose these (u_i, v_i) values is explained in Section 4. This defines a discrete function $\varphi : \Omega \rightarrow \mathbf{R}^2$ such that $\forall \alpha_i \in \Omega, \varphi(\alpha_i) = \{\varphi^u(\alpha_i), \varphi^v(\alpha_i)\} = (u_i, v_i)$. As shown in Figure 2, a mapping function Φ can be then defined as the linear interpolation of φ over each triangle $\mathbf{T} = (\alpha_i, \alpha_j, \alpha_k)$ of $\mathcal{T}$. For each point $\mathbf{p}$ in $\mathbf{T}$, Φ is given by:

$$\left\{ \begin{array}{rcrcl} \Phi(\mathbf{p}) & = & & (1 - \lambda - \mu) & . \quad \varphi(\alpha_i) \\ & & + & \lambda & . \quad \varphi(\alpha_j) \\ & & + & \mu & . \quad \varphi(\alpha_k) \end{array} \right.$$

where:

- λ and μ are the local barycentric coordinates at the point $\mathbf{p}$ in $\mathbf{T}$
- $\varphi(\alpha_i) = (u_i, v_i) \quad , \quad \varphi(\alpha_j) = (u_j, v_j) \quad , \quad \varphi(\alpha_k) = (u_k, v_k)$

2.2 Discrete Smooth Interpolation

Given a triangulation $\mathcal{G} = \{\Omega, \mathcal{T}\}$, we want to assign (u, v) coordinates to each vertex $\alpha \in \Omega$. Floater has shown in [Flo97] that verifying the following two sufficient conditions constructs a mapping:

1. The image of the border of the surface through φ in the parametric (u, v) domain is a convex polygon.
2. Each internal node is a convex combination of its neighbours.

One must keep in mind that these two conditions are **sufficient** and not necessary to define mappings. We show in Section 4 how the first one can be replaced by a less restrictive condition.

More formally, the second condition can be written as follows (see Equation 1):

$$\forall k \in \Omega, \sum_{\alpha \in N(k)} v^{\alpha}(k).\varphi(\alpha) = 0 \quad (1)$$

where:

- $N(k)$ denotes the set of nodes directly connected to k, including k.
- the $v^{\alpha}(k)$ are **given** coefficients such that:

$$\begin{cases} v^{\alpha}(k) > 0 & \forall \alpha \in N(k) - \{k\} \\ v^{k}(k) = -\sum_{\substack{\alpha \in N(k) \\ \alpha \neq k}} v^{\alpha}(k) \neq 0 & \forall k \in \Omega \end{cases} \quad (2)$$

Once boundary nodes have been mapped to a convex polygon in parametric domain space, (u, v) coordinates must be assigned to the internal nodes of the triangulation. Instead of finding φ by directly solving Equation 1 as done in more classical approaches, the method described in this article consists of minimizing a global criterion *in a least square sense*, honoring at the same time a set of linear constraints, as will be shown in the next section. The algorithm is based on the *Discrete Smooth Interpolation* (D.S.I.), that we describe in [Mal89, Mal92]. The reader is referred to these two articles where the notions of *generalized roughness*, *linear constraints*, and the iterative D.S.I. algorithm introduced further on in this document are described in depth. The criterion minimized by the D.S.I. method is called the *roughness* R, and is defined in Equation 3 below:

$$R(\varphi) = \sum_{k \in \Omega} \sum_{\nu \in \{u,v\}} \left\{ \sum_{\alpha \in N(k)} v^{\alpha}(k).\varphi^{\nu}(k) \right\}^2 \quad (3)$$

The minimum of this functionnal is reached if $\partial R(\varphi)/\partial \varphi^{\nu}(\alpha) = 0$ for each $\alpha \in \Omega$ and for each $\nu \in \{u, v\}$, where ν denotes one of the two components of φ. This yields the following equation:

$$\varphi^{\nu}(\alpha) = -\frac{G^{\nu}(\alpha|\varphi)}{g^{\nu}(\alpha)}$$

where:

$$\begin{cases} G^{\nu}(\alpha|\varphi) = \sum_{k \in N(\alpha)} \left\{ v^{\alpha}(k). \sum_{\substack{\beta \in N(k) \\ \beta \neq \alpha}} v^{\beta}(k).\varphi^{\nu}(\beta) \right\} \\ g^{\nu}(\alpha) = \sum_{\alpha \in N(\alpha)} \{v^{\alpha}(k)\}^2 \end{cases} \quad (4)$$

The following algorithm computes iteratively the assignments of (u, v) coefficients minimizing the roughness given in Equation 3. We have proven in Mallet[Mal89] that it does converge to a unique solution, as soon as at least one node α has its value $\varphi(\alpha)$ fixed, and provided that the chosen $v^{\alpha}(k)$ coefficients honor Equation 2. Later in this document, we show how this method can be enhanced using D.S.I. constraints.

```
let I be the set of nodes where φ is unknown
let φ[0] be a given initial approximated solution
while (more iterations are needed) {
    for_all( α ∈ I ) {
        for_all( ν ∈ {u, v}) {
            φ^ν(α) := − G^ν(α) / g^ν(α)
        }
    }
}
```

Where the $v^{\alpha}(k)$ coefficients are concerned, several choices are available. One possible choice described by Floater[Flo97] is referred to as the *shape preserving* weighting, and ensures that the location of a vertex in parametric space relative to its neighbors mimics the local geometry around the vertex being considered. The approach described in this article is quite different, as by separating the criteria minimizing the distortions from those which ensure that a valid mapping is constructed, we can obtain a finer control on the way the surface is parametrized. For this reason, the simple harmonic weighting defined as follows is used for the $\{v^{\alpha}(k)\}$:

$$v^{\alpha}(k) = \begin{cases} 1 & \text{if} \quad \alpha \in N(k) - \{k\} \\ -degree(k) & \text{if} \quad \alpha = k \end{cases}$$

where $degree(k)$ denotes the number of neighbors of k. Clearly, one of the previously mentioned more sophisticated weightings such as the *shape preserving* or *gaussian* weightings could have been used instead, since they both satisfy Equation 2, but it is shown in the next section that by using linear constraints, the same effect can be obtained with higher flexibility.

3 NON-DISTORTED MAPPING

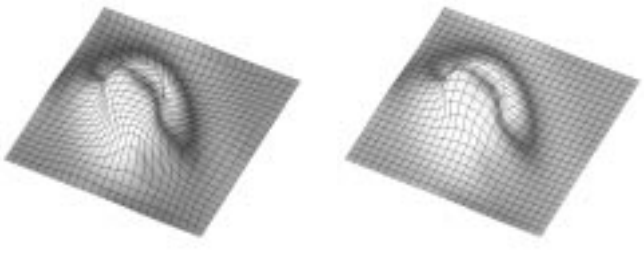


Figure 3: Isoparametric curves obtained without (A) and with (B) non-distortion constraints.

In this section, we define the criterion to be minimized in order to construct a non-distorted texture mapping. In a nutshell, this criterion preserves the perpendicularity and constant spacing of the isoparametric curves traced on the surface, as shown in Figure 3. In other words, the gradients of u and v should be perpendicular one to another and constant all over the surface (see [Car76]). This requires defining the gradient of a function interpolated over a triangulated mesh from the vertices of the triangulation, definition given below. The way the algorithm presented in the previous section can be modified to take into account this criterion is then explained.

Data in Figure 3 shows the effect of the constraints described in this section as applied while parametrizing a triangulated mesh.

The isoparametric curves obtained when applying the algorithm described in Section 2 are shown in Figure 3-A, whereas the constraints described further on give the result shown in Figure 3-B.

3.1 Gradient of a Discrete Function φ Interpolated over a Triangulation $\mathcal{G}$

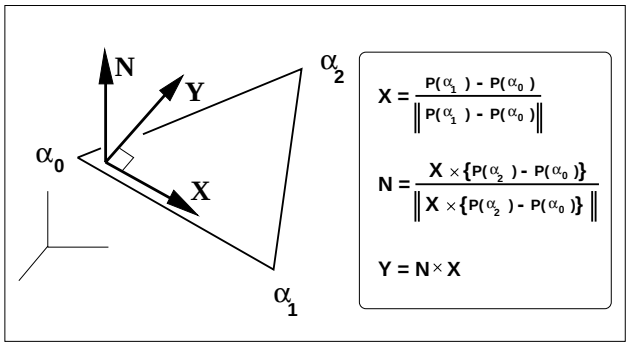


Figure 4: Local orthonormal basis (**X**, **Y**) of a triangle $\mathbf{T} = (\alpha_0, \alpha_1, \alpha_2)$.

As shown in Figure 4, each triangle $\mathbf{T} = (\alpha_0, \alpha_1, \alpha_2)$ of $\mathcal{T}$ can be provided with a local orthonormal basis $(\mathbf{p}(\alpha_0), \mathbf{X}, \mathbf{Y})$. The function $\varphi^\nu_T(X, Y)$ denotes the linear interpolation of φ^ν over the triangle $\mathbf{T}$, where $\nu \in \{u, v\}$ represents one of the two components of φ and where (X, Y) are the local coordinates in the orthonormal basis $(\mathbf{p}(\alpha_0), \mathbf{X}, \mathbf{Y})$ of $\mathbf{T}$.

In this basis, one can check that the gradient of φ^ν_T is constant over $\mathbf{T}$ and is a linear combination of the values of φ^ν_T at the three vertices of the triangle $\mathbf{T}$. The six coefficients $D_X(\alpha_j)$ and $D_Y(\alpha_j)$ given in Equation 5 below are solely dependent on the geometry of the triangle $\mathbf{T}$.

$$\left| \begin{array}{rcl} \frac{\partial \varphi^\nu_T}{\partial X} & = & \sum_{j=0}^{2} D_X(\alpha_j).\varphi^\nu(\alpha_j) \\ \\ \frac{\partial \varphi^\nu_T}{\partial Y} & = & \sum_{j=0}^{2} D_Y(\alpha_j).\varphi^\nu(\alpha_j) \end{array} \right.$$

$$\text{where:} \left\{ \begin{array}{rcl} D_X(\alpha_0) & = & (y_1 - y_2)/d \\ D_X(\alpha_1) & = & (y_2 - y_0)/d \\ D_X(\alpha_2) & = & (y_0 - y_1)/d \\ D_Y(\alpha_0) & = & (x_2 - x_1)/d \\ D_Y(\alpha_1) & = & (x_0 - x_2)/d \\ D_Y(\alpha_2) & = & (x_1 - x_0)/d \end{array} \right.$$

$$d = (x_1 - x_0).(y_2 - y_0) - (x_2 - x_0).(y_1 - y_0)$$

$$\left| \begin{array}{rcl} x_j & = & (\mathbf{p}(\alpha_j) - \mathbf{p}(\alpha_0)).\mathbf{X} \\ y_j & = & (\mathbf{p}(\alpha_j) - \mathbf{p}(\alpha_0)).\mathbf{Y} \end{array} \right| \quad \forall j \in \{0, 1, 2\} \tag{5}$$

Using this definition of the gradient of φ, it is possible to write the equations corresponding to the orthogonality and homogeneous spacing of the isoparametric curves. The orthogonality of the iso-u and iso-v curves in a triangle $\mathbf{T}$ is given by:

$$\left[\frac{\partial \varphi^u_T}{\partial X} \quad \frac{\partial \varphi^u_T}{\partial Y} \right] . \left[\begin{array}{c} \frac{\partial \varphi^v_T}{\partial X} \\ \\ \frac{\partial \varphi^v_T}{\partial Y} \end{array} \right] = 0 \tag{6}$$

If we consider that φ^u is fixed and that φ^v is to be determined, replacing in Equation 6 the gradient of φ^v with its expression given in Equation 5 yields the following equation, which **linearly** combines the values of φ^v at the three vertices $(\alpha_0, \alpha_1, \alpha 2)$ of $\mathbf{T}$. The equation to be used when φ^u is interpolated can be obtained by exchanging u and v in Equation 7.

$$\sum_{j \in \{0,1,2\}} \left\{ \varphi^v(\alpha_j). \left(\frac{\partial \varphi^u_T}{\partial X}.D_X(\alpha_j) + \frac{\partial \varphi^u_T}{\partial Y}.D_Y(\alpha_j) \right) \right\} = 0 \tag{7}$$

The remaining condition on φ concerns the homogeneous spacing of the isoparametric curves. In other words, the gradient must not vary from one triangle to another. This requires that a common basis for two adjacent triangles $\mathbf{T}$ and $\tilde{\mathbf{T}}$ be defined, as shown in Figure 5. The same expressions as introduced in Figure 4 are used. The vector $\mathbf{X}$ is shared by the two bases, and $\tilde{\mathbf{Y}}$ is such that $\mathbf{Y}$ and $\tilde{\mathbf{Y}}$ would become colinear if the pair of triangles $(\mathbf{T}, \tilde{\mathbf{T}})$ was unfolded along their common edge $[\alpha_0, \alpha_1]$.

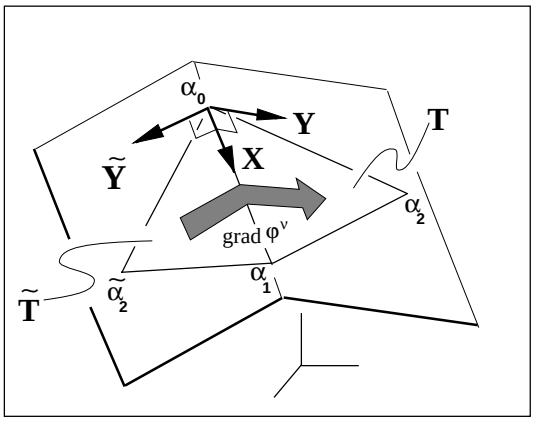


Figure 5: Constant gradient across the common edge of two triangles $\mathbf{T}$ and $\tilde{\mathbf{T}}$.

The homogeneous spacing of the isoparametric curves is verified if, and only if, for each edge of $\mathcal{T}$ the equation below is verified:

$$\left\{ \begin{array}{rcl} \frac{\partial \varphi^u_T}{\partial X} & = & \frac{\partial \varphi^u_{\tilde{T}}}{\partial X} \quad ; \quad \frac{\partial \varphi^u_T}{\partial Y} = -\frac{\partial \varphi^u_{\tilde{T}}}{\partial \tilde{Y}} \\ \\ \frac{\partial \varphi^v_T}{\partial X} & = & \frac{\partial \varphi^v_{\tilde{T}}}{\partial X} \quad ; \quad \frac{\partial \varphi^v_T}{\partial Y} = -\frac{\partial \varphi^v_{\tilde{T}}}{\partial \tilde{Y}} \end{array} \right. \tag{8}$$

By replacing in Equation 8 the gradients of φ^u and φ^v by their expressions in $\mathbf{T}$ and $\tilde{\mathbf{T}}$, the following four **linear** equations are obtained (see Equation 9), concerning the two components X and Y of the gradients of φ^u and φ^v. The term δ_W takes into account the fact that $\mathbf{Y}$ and $\tilde{\mathbf{Y}}$ point in an opposite direction.

$$\begin{array}{l} \\ \forall \nu \in \{u, v\}, \\ \forall W \in \{X, Y\}, \end{array} \left| \begin{array}{rcl} \varphi^\nu(\alpha_0). \left\{ D_W(\alpha_0) + \delta_W . \tilde{D_W}(\alpha_0) \right\} & + \\ \varphi^\nu(\alpha_1). \left\{ D_W(\alpha_1) + \delta_W . \tilde{D_W}(\alpha_1) \right\} & + \\ \varphi^\nu(\alpha_2). \{ D_W(\alpha_2) \} & + \\ \varphi^\nu(\tilde{\alpha_2}). \left\{ \delta_W . \tilde{D_W}(\tilde{\alpha_2}) \right\} & = & 0 \end{array} \right.$$

$$\text{where} \quad \delta_W = \left\{ \begin{array}{ll} -1 & \text{if} \quad W = X \\ +1 & \text{if} \quad W = Y \end{array} \right. \tag{9}$$

3.2 Honoring Linear Constraints

We have shown in Section 2 how D.S.I. can be used to construct a mapping of a triangulated mesh. What we want to do now is to take into account the two criteria minimizing the distortions of the mapping, namely the perpendicularity and homogeneity criteria previously introduced. These two criteria can be written as a set of linear equations. As it is not possible to honor these constraints for a non-developable surface, they will be respected *in a least square sense*, thus *minimizing* the distortions. The general form of such a constraint is given in Equation 10 below:

$$\sum_{\alpha \in \Omega} \{A_{c^\nu}(\alpha).\varphi^\nu(\alpha)\} \quad = \quad b_{c^\nu} \tag{10}$$

where the values $A_{c^\nu}(\alpha)$ and the scalar b_{c^ν} are constant **given** coefficients defining the constraint c.

Equation 7, corresponding to the perpendicularity of the isoparametric curves in the triangle $\mathbf{T} = (\alpha_0, \alpha_1, \alpha_2)$, yields two constraints $c^u_{\mathbf{T}}$ and $c^v_{\mathbf{T}}$ to be honored when interpolating φ^u and φ^v respectively. The expression of $c^v_{\mathbf{T}}$ is given below in Equation 11. The expression of the twin constraint $c^u_{\mathbf{T}}$ can be obtained by permuting u and v in this equation.

$$\left|\begin{array}{lcl} \forall j \in \{0,1,2\}, & & \\ \qquad A_{c^v_{\mathbf{T}}}(\alpha_j) & = & \frac{\partial \varphi^u_T}{\partial X}.D_X(\alpha_j) + \frac{\partial \varphi^u_T}{\partial Y}.D_Y(\alpha_j) \\ \forall \alpha \notin \{\alpha_0,\alpha_1,\alpha_2\}, & & \\ \qquad A_{c^v_{\mathbf{T}}}(\alpha) & = & 0 \\ \qquad b_{c^v_{\mathbf{T}}} & = & 0 \end{array}\right. \tag{11}$$

The homogeneity criterion specified by Equation 9 can be expressed by the following four constraints $c^{u_X}_{\mathbf{E}}$, $c^{u_Y}_{\mathbf{E}}$, $c^{v_X}_{\mathbf{E}}$ and $c^{v_Y}_{\mathbf{E}}$ yielded by Equation 12 below, to be taken into account at each edge $\mathbf{E} = (\alpha_0, \alpha_1)$ of the triangulation $\mathcal{G}$. The vertices α_2 and $\tilde{\alpha_2}$ denote the two remaining vertices of the two triangles $\mathbf{T}$ and $\tilde{\mathbf{T}}$ sharing the edge $\mathbf{E}$.

$$\left|\begin{array}{lcl} A_{c^\nu_{\mathbf{E}W}}(\alpha_0) & = & \left\{D_W(\alpha_0) + \delta_W.\tilde{D_W}(\alpha_0)\right\} \\ A_{c^\nu_{\mathbf{E}W}}(\alpha_1) & = & \left\{D_W(\alpha_1) + \delta_W.\tilde{D_W}(\alpha_1)\right\} \\ A_{c^\nu_{\mathbf{E}W}}(\alpha_2) & = & D_W(\alpha_2) \\ A_{c^\nu_{\mathbf{E}W}}(\tilde{\alpha_2}) & = & \delta_W.\tilde{D_W}(\tilde{\alpha_2}) \\ A_{c^\nu_{\mathbf{E}W}}(\alpha) & = & 0 \quad \forall \alpha \notin \{\alpha_0,\alpha_1,\alpha_2,\tilde{\alpha_2}\} \\ b_{c^\nu_{\mathbf{E}W}} & = & 0 \end{array}\right.$$

where:

$$\nu \in \{u,v\} \quad ; \quad W \in \{X,Y\} \quad ; \quad \delta_W = \left\{\begin{array}{ll} -1 & \text{if} \quad W = X \\ +1 & \text{if} \quad W = Y \end{array}\right. \tag{12}$$

The *roughness* criterion which D.S.I. minimizes can be generalized in order to honor a set $\mathcal{C}$ of linear constraints in a least square sense. In our case, the set $\mathcal{C}$ of constraints is given by Equation 13, where $\mathcal{E}$ denotes the set of the edges of the triangulation $\mathcal{G}(\Omega, \mathcal{T})$.

$$\mathcal{C} = \left(\bigcup_{\mathbf{T} \in \mathcal{T}} \{c^u_{\mathbf{T}}, c^v_{\mathbf{T}}\}\right) \cup \left(\bigcup_{\mathbf{E} \in \mathcal{E}} \{c^{u_X}_{\mathbf{E}}, c^{u_Y}_{\mathbf{E}}, c^{v_X}_{\mathbf{E}}, c^{v_Y}_{\mathbf{E}}\}\right) \tag{13}$$

The *generalized roughness* $R^*(\varphi)$, taking into account the degree of violation of the constraints $\mathcal{C}$, is given by Equation 14 below. In addition to the equation of the roughness given in Section 3 (Equation 3), several terms correspond to the linear constraints, as described further on:

$$R^*(\varphi) = R(\varphi) \quad + \quad \phi. \sum_{c \in \mathcal{C}} \varpi_c. \left\{ \left(\sum_{\nu \in \{u,v\}} \sum_{\alpha \in \Omega} A^\nu_c(\alpha).\varphi^\nu(\alpha) \right) - b_c \right\}^2 \tag{14}$$

In Equation 14, the term $R(\varphi)$ is the *roughness* (see Equation 3), and the second term represents the degree of violation of the linear constraints. Each constraint c is ponderated by a given $\varpi_c > 0$ coefficient, allowing to tune the relative importance of the constraints. For instance, it is possible to make the mapping respect the perpendicularity rather than the homogeneity. Moreover, since each triangle $\mathbf{T}$ and edge $\mathbf{E}$ has an individual constraint defined, as well as an individual associated ϖ_c coefficient, it is possible to select the surface zones where the distortions are to be minimized in order of preference. The remaining coefficient $\phi \in]0, +\infty[$ is a given parameter called the *fitting factor* and representing the importance of the constraints relative to the roughness.

The functionnal $R^*(\varphi)$ is a quadratic form, whose minimum is reached if $\partial R^*(\varphi)/\partial \varphi^\nu(\alpha) = 0$ for each $\nu \in \{u,v\}$ and for each $\alpha \in \Omega$. This yields the following equation, which solution minimizes $R^*(\varphi)$:

$$\varphi^\nu(\alpha) = -\frac{G^\nu(\alpha|\phi) + (\phi.\varpi).\Gamma^\nu(\alpha|\varphi)}{g^\nu(\alpha) + (\phi.\varpi).\gamma^\nu(\alpha)} \tag{15}$$

$$\left|\begin{array}{lcl} \Gamma^\nu(\alpha|\varphi) & = & \sum_{c \in \mathcal{C}} \varpi_c.\Gamma^\nu_c(\alpha|\varphi) \\ \gamma^\nu(\alpha) & = & \sum_{c \in \mathcal{C}} \varpi_c.\gamma^\nu_c(\alpha) \end{array}\right. \tag{16}$$

with:

$$\left\{\begin{array}{lcl} \gamma^\nu_c(\alpha) & = & \left(A^\nu_c(\alpha)\right)^2 \\ \Gamma^\nu_c(\alpha|\varphi) & = & A^\nu_c(\alpha).\left\{\sum_{\beta \neq \alpha} A^\nu_c(\beta).\varphi^\nu(\beta) - b_c\right\} \end{array}\right. \tag{17}$$

The orthogonality constraint suggests a modification in the iterative D.S.I. algorithm. The two internal loops iterating on the components of φ and on the nodes of Ω respectively have been inverted. At each iteration, φ^u is interpolated while φ^v is considered to be constant, then the roles of φ^u and φ^v are permuted. The resulting algorithm given below assigns (u, v) coordinates to the vertices of the triangulation while respecting the specified set of constraints.

let I be the set of nodes where φ is unknown
let $\varphi_{[0]}$ be a given initial approximated solution
while (more iterations are needed) {
  for_all($\nu \in \{u, v\}$) {
    for_all($\alpha \in I$) {
      $\varphi^{\nu}(\alpha) := -\frac{G^{\nu}(\alpha)+\Gamma^{\nu}(\alpha|\phi)}{g^{\nu}(\alpha)+\gamma^{\nu}(\alpha)}$
    }
  }
}

4 LOCALLY CONSTRAINING A MAPPING

The constraints defined so far in this paper provide the user with a *global* control on the mapping function. Even if the orthogonality and perpendicularity constraints can be weighted locally to specify the zones where distortions are preferably to be minimized, this may be not sufficient for some applications, where a more precise set of local constraints is required. For instance, it may be necessary to align some details of textures with details of models, which can be achieved by specifying isoparametric curves. Moreover, the model to be texture mapped can present cuts, and the user may want do define a single mapping function for a cut model instead of sewing together several patches. This requirement can be fulfilled by making the mapping continuous through cuts as described further on.

4.1 Specifying an Isoparametric Curve

As shown in Figure 6, we consider that we have a given polygonal curve $L = \{\mathbf{p}_0, \ldots, \mathbf{p}_m\}$ associated with a given value u_0 of the parameter u. We describe here the constraints to be honored for making the isoparametric of the mapping defined by $(u = u_0)$ correspond to the projection of L on the surface $\mathbf{S}$. Each point $\mathbf{p}_i$ of L yields a constraint $c_{\mathbf{p}_i}$ ensuring that the isoparametric curve $u = u_0$ of the mapping φ passes near the projection $\mathbf{p}'_i$ of $\mathbf{p}_i$ on $\mathbf{S}$.

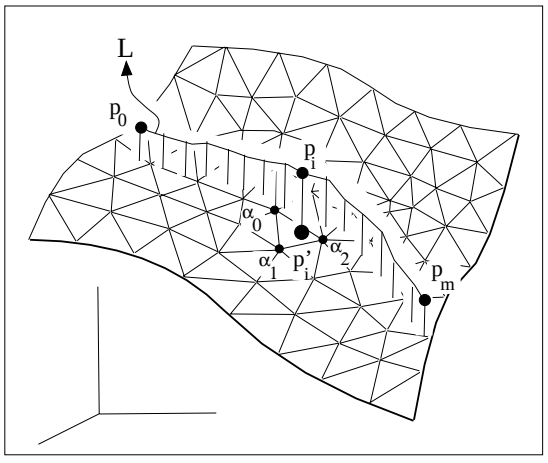


Figure 6: Aligning details of the texture to details of the model by specifying an isoparametric curve.

The triangle $\mathbf{T} = (\alpha_0, \alpha_1, \alpha_2)$ is the triangle of $\mathbf{S}$ that contains $\mathbf{p}'_i$, and $(\lambda_0, \lambda_1, \lambda_2)$ are the barycentric coordinates of $\mathbf{p}'_i$ in $\mathbf{T}$. The linear relation to be honored is given in Equation 18 below.

$$\left| \begin{array}{l} \sum\limits_{j\in\{0,1,2\}} \lambda_j . \varphi^u(\alpha_j) = u_0 \\ \\ \text{where:} \quad \left\{ \begin{array}{l} \sum\limits_{j\in\{0,1,2\}} \lambda_j . \mathbf{p}(\alpha_j) = \mathbf{p}'_i \\ \\ \sum\limits_{j\in\{0,1,2\}} \lambda_j = 1 \end{array} \right. \end{array} \right. \tag{18}$$

Equation 19 below gives the expression of the constraint $c_{\mathbf{p}_i}$ in the form of Equation 10 in Section 2. Such a constraint per point $\mathbf{p}_i$ is added to the set $\mathcal{C}$ to be honored by D.S.I., introduced in the previous section.

$$\left\{ \begin{array}{lcll} A_{c_{\mathbf{p}_i}}(\alpha_j) & = & \lambda_j & \forall j \in \{0,1,2\} \\ A_{c_{\mathbf{p}_i}}(\alpha) & = & 0 & \forall \alpha \notin \{\alpha_0, \alpha_1, \alpha_2\} \\ b_{c_{\mathbf{p}_i}} & = & u_0 & \end{array} \right. \tag{19}$$

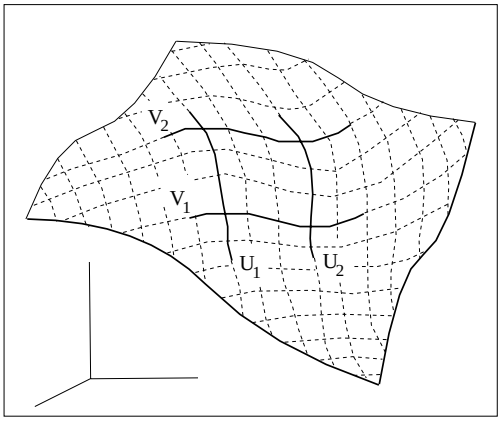


Figure 7: Extrapolating a mapping from four user specified isoparametric curves.

Remark:

As mentioned in Section 2, the two following **sufficient** conditions ensure that a discrete function φ defines a mapping:

1. The image of the border of the surface through φ in the parametric (u, v) domain is a convex polygon.
2. Each internal node is a convex combination of its neighbors.

Introducing the constraints to ensure that the isoparametric curves are orthogonal, with homogeneous spacing means the first condition can be replaced by a less restrictive one. As shown in Figure 7, it is then sufficient to specify four arcs of isoparametric curves $\{u_1, u_2, v_1, v_2\}$ using the constraint previously introduced. Thus, by enabling us to use the algorithm not only as an *interpolator*, but also as an *extrapolator*, it is possible to construct mappings for surfaces having complex shaped borders by leaving φ unspecified on the border.

4.2 Constructing a Mapping for a Cut Surface

Let us now consider that the surface has cuts, and that we want the mapping function Φ to be continuous through these cuts. Instead of using several distinct patches and making the edges of the patches match as described in [Blo85], the surface is considered here as a single patch (as it was before being cut), as suggested in [CEM97]. The set of constraints described below allows us to assign (u, v) coordinates to the vertices of the triangulation in such a way that the two borders of a cut are mapped to the same curve by the interpolated Φ mapping function. In other words, the cuts are sewn in (u, v) domain space.

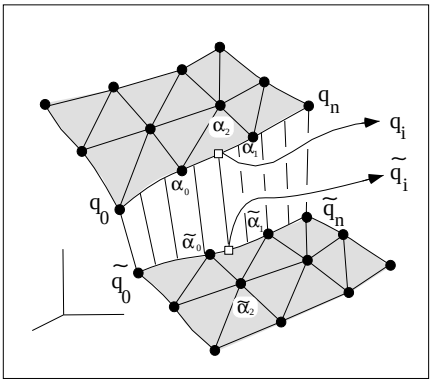


Figure 8: Connecting two borders of a cut in texture space.

As shown in Figure 8, twin set of points $\{\mathbf{q}_i, i = 0 \dots n\}$ and $\{\tilde{\mathbf{q}}_i, i = 0 \dots n\}$ are sampled on the twin borders of the cut. We describe now how to make the mapping match at each pair $(\mathbf{q}_i, \tilde{\mathbf{q}}_i)$ of points. More precisely, we want to respect the following conditions:

$$\forall \nu \in \{u, v\} \left| \begin{array}{lrcl} (1) & \varphi_T^\nu(\mathbf{q}_i) & = & \varphi_{\tilde{T}}^\nu(\tilde{\mathbf{q}}_i) \\ (2) & \mathbf{grad}\varphi_T^\nu & = & \mathbf{grad}\varphi_{\tilde{T}}^\nu \end{array} \right. \quad (20)$$

where T and $\tilde{T}$ denote the triangles containing $\mathbf{q}_i$ and $\tilde{\mathbf{q}}_i$ respectively. The gradient $\mathbf{grad}\varphi_T^\nu$ is computed as described in Section 3 (see Equation 5), using the basis shown in Figure 4.

Using the methods introduced in the previous two sections, it is easy to translate these two conditions into the D.S.I. constraints $c^\nu_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}$ and $c^{\nu_W}_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}$, given below in the equations 21 and 22 respectively.

$$\left| \begin{array}{lcll} A_{c^\nu_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}}(\alpha_j) & = & \lambda_j & \forall j \in \{0,1\} \\ A_{c^\nu_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}}(\tilde{\alpha_j}) & = & -\tilde{\lambda_j} & \forall j \in \{0,1\} \\ A_{c^\nu_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}}(\alpha_j) & = & 0 & \forall \alpha \notin \{\alpha_0, \alpha_1, \tilde{\alpha_0}, \tilde{\alpha_1}\} \\ b_{c^\nu_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}}(\alpha_j) & = & 0 & \end{array} \right. \quad (21)$$

where $\lambda_{j,j\in\{0,1\}}$ and $\tilde{\lambda}_{j,j\in\{0,1\}}$ denote the barycentric coordinates of $\mathbf{q}_i$ in $[\mathbf{p}(\alpha_0), \mathbf{p}(\alpha_1)]$ and $\tilde{\mathbf{q}}_i$ in $[\mathbf{p}(\tilde{\alpha_0}), \mathbf{p}(\tilde{\alpha_1})]$ respectively.

The four constraints $c^{u_X}_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}$, $c^{u_Y}_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}$, $c^{v_X}_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}$, and $c^{v_Y}_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}$ yielded by Equation 22 below ensure a constant gradient of the mapping through the cut. In other words, an isoparametric curve points in the same direction in the two corresponding triangles $\mathbf{T}$ and $\tilde{\mathbf{T}}$.

$$\left| \begin{array}{lcll} A_{c^{\nu_W}_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}}(\alpha_j) & = & D_j & \forall j \in \{0,1,2\} \\ A_{c^{\nu_W}_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}}(\tilde{\alpha_j}) & = & \delta_W . \tilde{D_j} & \forall j \in \{0,1,2\} \\ A_{c^{\nu_W}_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}}(\alpha) & = & 0 & \forall \alpha \notin \{\alpha_0, \alpha_1, \alpha_2, \tilde{\alpha_0}, \tilde{\alpha_1}, \tilde{\alpha_2}\} \\ b_{c^{\nu_W}_{\mathbf{q}_i,\tilde{\mathbf{q}}_i}}(\alpha) & = & 0 & \end{array} \right.$$

where:

$$\nu \in \{u, v\} \quad ; \quad W \in \{X, Y\} \quad ; \quad \delta_W = \left\{ \begin{array}{lll} -1 & \text{if} & W = X \\ +1 & \text{if} & W = Y \end{array} \right. \quad (22)$$

5 RESULTS AND APPLICATIONS

One can see in Figure 9 the results of the method applied to a triangulated mesh representing a face (see Figure 9-A). The effect of the orthogonality and homogeneity constraints can be brought to the fore by comparing Figure 9-B (no constraint used) and Figure 9-E (orthogonality and homogeneity enforced), where a checker pattern is mapped to the mesh. The isoparametric curves corresponding to this latter image are displayed in Figure 9-D, where one can check that the iso-u curves shown in red are perpendicular to the iso-v shown in blue. In Figure 9-C, the same non-distorting mapping function is used with a fancier texture. For all these pictures, the constraints ensuring the continuity of the mapping through cuts have been specified at the mouth and the eyes of the model. This model has 3000 triangles, and has been parametrized after 100 iterations in approximatively one minute using an R4000 machine.

As with any other texture mapping method, or more precisely as with any parametrization algorithm, our techniques may be applied to problems other than those associated with texture mapping. In the realm of geosciences, several different methods based on our technique have been implemented into a widely used geology oriented C.A.D. software. Among all the possible applications, to name but a few:

- Unfolding surfaces representing the boundaries of geological layers while preserving the volume of the layers;
- Generating grids suitable for finite elements analysis;
- Beautifying triangulated meshes by remeshing in (u, v) domain space.
- Constructing Spline surfaces from triangulated meshes;
- Performing computations such as *geostatistical simulations* in (u, v) domain space.

Not only do these applications require that mappings present non-distorting properties, which is fulfilled by our method, but in addition, these applications will benefit from the ability of our method to take into account additional information expressed in the form of linear constraints.

The method applied to geological data is demonstrated in Figure 10. In Figure 10-A, one can see a mapping of a complex cut surface, corresponding to a boundary of a geological layer presenting faults. In Figure 10-B, the isoparametric curves of the mapping are displayed, and one can see that the mapping is continuous through the

cuts of the surface. In Figure 10-(C,D,E), a surface representing a dome of salt is parametrized. For this kind of surfaces which are far from developable, distortions will still remain, and one can choose a compromise between the orthogonality and the homogeneity of the mapping by tuning the weightings ϖ_c of the two constraints. In Figure 10-C, the orthogonality is respected, but the sizes of the squares differ in a great deal, whereas in Figure 10-E the squares have approximatively the same size while the isoparametric curves are far from orthogonal. An average solution is shown in Figure 10-D, where the same weighting has been used for the two constraints. One can see in Figure 10-F a mapped surface with an isoparametric curve specified. As shown in Figure 10-G, the texture has been aligned to this curve.

CONCLUSIONS

We have presented in this paper new techniques for non-distorted mapping. In addition to the other methods based on global minimization of distortions, our method can easily take into account various additional information. It is thus possible to specify the zones where distortions should be minimized in order of preference, to make a set of isoparametric passes through user specified curves, and to sew the cuts of a surface in texture space. Moreover, it is very easy to extend the method by defining new constraints, once these constraints can be expressed as linear (or linearizable) relations.

The method can be easily implemented, since it does only require an efficient representation of triangulated meshes, which is provided by most C.A.D. packages. Thus, the algorithm has been integrated as a basic algorithm into a widely used C.A.D. software dedicated to geology, and several methods other than these associated with texture mapping have been developped based on this algorithm, such as unfolding geological layers and performing computations in texture space.

One of the limitations of the technique is that it can be applied to planar graphs only, i.e. to surfaces topologically equivalent to a disk. A generalization of the method working on arbitrary topology could be realized, by dividing the surface into (topological) disks using a Voronoi based approach, as proposed in [ERDH95]. A method such as the one descrined in [Tur91] could be also used to choose the sites of the Voronoi diagram. A constraint ensuring the continuity of the gradient from one domain to another could be added (see Equation 22), thus blurring the limits of the base triangles that appear when directly applying the method described in [ERDH95]. The interactivity of the tool could also be improved by speeding up the algorithm, using a conjugate gradient method. This latter improvement together with a large set of possible local constraints could have an important impact on 3D paint systems. Future research also comprise the extension of the method to tetrahedralized meshes, enabling to assign (u, v, w) coordinates to the vertices of tetrahedralizations.

AKNOWLEDGEMENTS

This research has been performed in the frame of the G◯CAD project, and the authors want to thank here the sponsors of the consortium, especially *Gaz de France* who supports this work. Thanks also to the reviewers for their interresting comments.

References

[Blo85] J. Bloomenthal. Modeling the Mighty Maple. In *SIGGRAPH 85 Conference Proceedings*, volume 19, pages 305–311. ACM, July 1985.

[BS86] E. Bier and K. Sloan. Two-Part Texture Mapping. *IEEE Computer Graphics and Applications*, pages 40–53, September 1986.

[BVI91] C. Bennis, J.M. Vézien, and G. Iglésias. Piecewise Surface Flattening for Non-Distorted Texture Mapping. In *SIGGRAPH 91 Conference Proceedings*, volume 25, pages 237–246. ACM, July 1991.

[Car76] M.F. Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice Hall, Englewood Cliffs, Inc., 1976.

[Cat74] E. Catmull. A Subdivision Algorithm for Computer Display of Curved Surfaces. *PhD thesis*, Dept. of Computer Sciences, University of Utah, December 1974.

[CEM97] R. Cognot, T. Aït Ettajer and J.L. Mallet. Modeling Discontinuities on Faulted Geological Surfaces. In *SEG Technical Program*, pages 1711–1718, November 1997.

[ERDH95] M. Eck, T. DeRose, T. Duchamp, H. Hoppes, M. Lounsbery and W. Stuetzle. Multiresolution Analysis of Arbitrary Meshes. In *SIGGRAPH 95 Conference Proceedings*, pages 173–182. ACM, August 1995.

[Flo97] M.S. Floater. Parametrization and Smooth Approximation of Surface Triangulations. *Computer Aided Geometric Design*, 14(3):231–250, April 1997.

[KL96] V. Krishnamurthy and M. Levoy. Fitting Smooth Surfaces to Dense Polygon Meshes. *SIGGRAPH 96 Conference Proceedings*, pages 313–324. ACM, August 1996.

[LM94] P. Litwinowicz and G. Miller. Efficient Techniques for Interactive Texture Placement. In *SIGGRAPH 94 Conference Proceedings*, pages 119–122. ACM, July 1994.

[Mal89] J.L. Mallet. Discrete Smooth Interpolation in Geometric Modeling. *ACM-Transactions on Graphics*, 8(2):121–144, 1989.

[Mal92] J.L. Mallet. Discrete Smooth Interpolation. *Computer Aided Design Journal*, 24(4):263–270, 1992.

[ML88] S.D. Ma and H. Lin. Optimal Texture Mapping. In *EUROGRAPHICS'88*, pages 421–428, September 1988.

[MYV93] J. Maillot, H. Yahia, and A. Verroust. Interactive Texture Mapping. In *SIGGRAPH 93 Conference Proceedings*, volume 27. ACM, 1993.

[Pea85] Peachey, R. Darwyn. Solid Texturing of Complex Surfaces. In *SIGGRAPH 85 Conference Proceedings*, volume 19, pages 287–296. ACM, July 1985.

[Ped95] H.K. Pedersen. Decorating Implicit Surfaces. In *SIGGRAPH 95 Conference Proceedings*, pages 291–300. ACM, 1995.

[SMSW86] Samek, Marcel, C. Slean, and H. Weghorst. Texture Mapping and Distortions in Digital Graphics. *The Visual Computer*, 2(5):313–320, September 1986.

[Tur91] G. Turk. Generating Textures on Arbitrary Surfaces Using Reaction-Diffusion. In *SIGGRAPH 91 Conference Proceedings*, pages 289–298. ACM, 1991.

[Tut60] W.T. Tutte. Convex Representation of Graphs. In *Proc. London Math. Soc.*, volume 10, 1960.

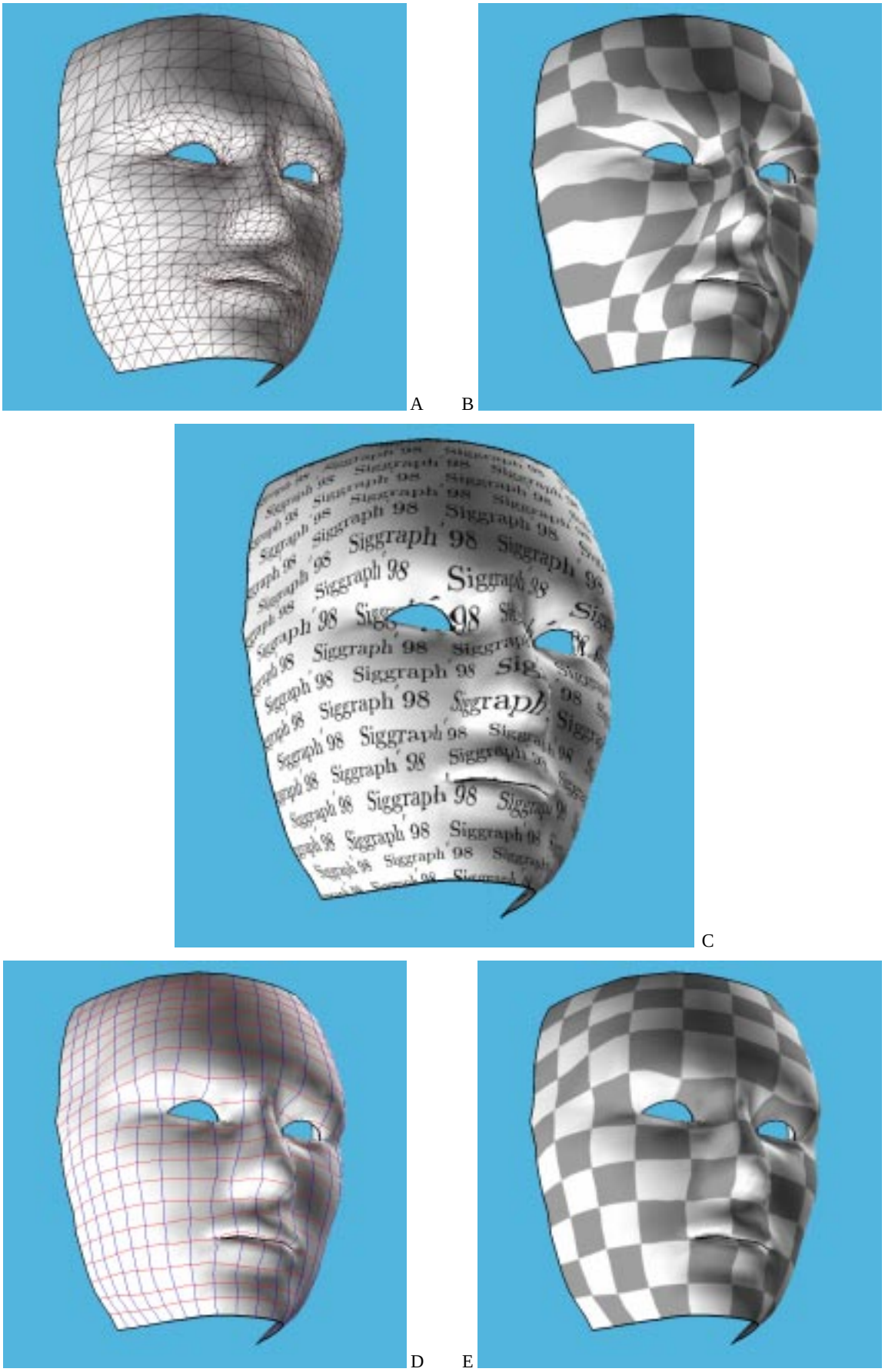
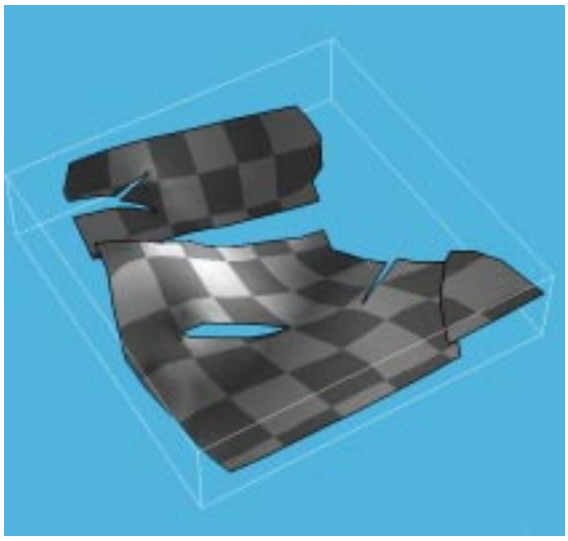
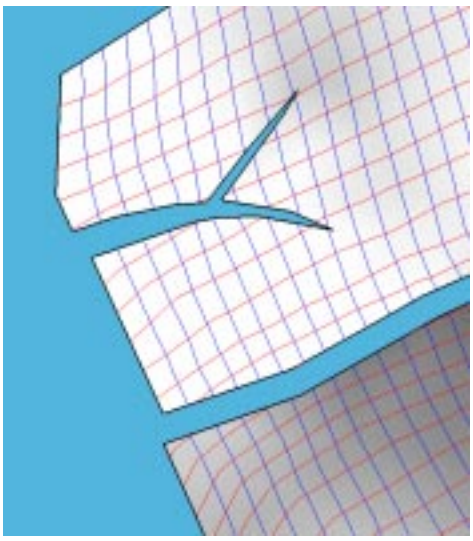
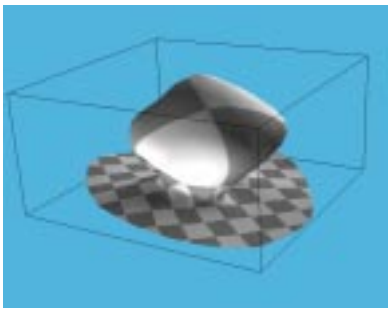
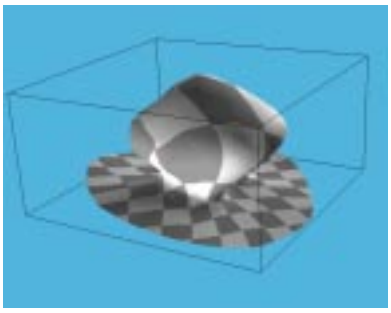


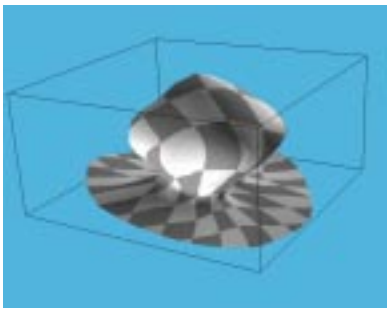
Figure 9: Texture mapping on a face.

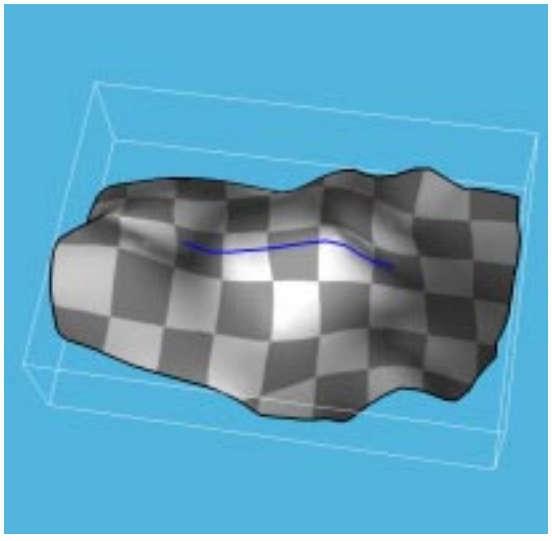

A


B


C


D


E


F

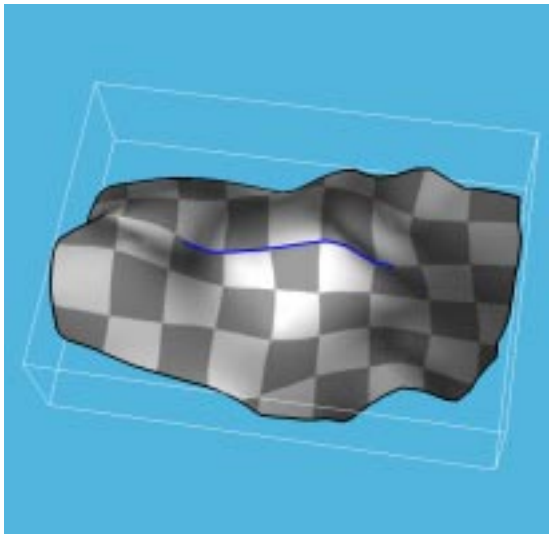

G

Figure 10: Applications to geology.