INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Trimmed Textures

Fabrice Neyret

N° 2857

Avril 1996

THÈME 3

Rapport de recherche

ISSN 0249-6399

Trimmed Textures

Fabrice Neyret *

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projet Syntim

Rapport de recherche n° 2857 — Avril 1996 — 12 pages

Abstract: Textures play an important role in the design of synthetic scenes. However, setting textures boundaries independently of the surface mesh edges is still a problem. In this paper, we propose a simple and cheap solution to deal with this problem without modifying the mesh, which introduces an extra cost only for facets containing a texture boundary (by *facet*, we mean any kind of surface element (polygon, bicubic patch...), having any kind of parameterisation (barycentric, etc...)).

This is achieved by associating to the surface 2D closed curves defining the boundaries of the textures. A pre-processing is done before rendering, clipping and storing these 2D texture trimming curves in each facet, and associating each possible texture to a combination of the areas defined by the curves. At rendering time, the current point is checked relatively to these areas, thus determinating the texture to use on this location of the surface.

Key-words: textures, mapping, surface trimming

(Résumé : tsvp)

* Fabrice.Neyret@inria.fr http://www-rocq.inria.fr/syntim/research/neyret

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
Téléphone : (33 1) 39 63 55 11 – Télécopie : (33 1) 39 63 53 30

Découpe de Texture

Résumé : Les textures jouent un rôle important dans la conception des scènes de synthèse. Cependant, fixer les bords des zones texturées indépendamment des arêtes du maillage constitue toujours un problème. Dans ce papier, nous proposons une solution simple et économique pour traiter ce problème sans modifier le maillage, et qui n'introduit un surcoût que pour les facettes qui contiennent une limite de texture (par *facette*, nous entendons toute sorte d'élément de surface (polygone, patch bicubique...), ayant n'importe quel type de paramétrisation (barycentrique, etc...)).

Ceci est obtenu en associant à la surface des courbes fermées 2D définissant les bords des textures. Un prétraitement est effectué avant le rendu, pour clipper et stocker ces courbes de découpe 2D dans chaque facette, et associer chacune des textures à une combinaison des zones définies par les courbes. Lors du rendu, on teste si le point courant est dans l'une de ces zones, afin de déterminer la texture à utiliser en cet endroit de la surface.

Mots-clé : textures, plaquage, découpe de surface

1 Introduction

Texturing is an important feature of both realistic and non-realistic synthetic scenes. Many different kinds of textures and texture mapping tools are available to designers. An important open problem is the design of texture limits, when a texture does not cover the whole surface.

Two solutions are currently available to solve this problem:

- modifying the surface mesh to fit the texture boundary,
- using a large texture map whose pixels indicate the texture to use on each location on the surface.

The first solution modifies the mesh and increases the rendering cost (because of the mesh refinement). Checking for an attribute at a point of a surface is a 2D problem, while introducing additional facets turns it into a 3D problem.

The second solution needs a map whose resolution can be very high, and which has to be checked for every location, even when the texture limit occurs only in some polygons.

We propose a simple and efficient way of trimming textures on surfaces independently of the facets edges. The idea is to let the user specify closed 2D curves on the surface as for surface trimming [CR87], in order to design the area of each texture (some areas possibly contain several superposed textures). These curves are pre-processed before rendering (ray-tracing or scan-line), by storing clipped curves in the data structure of relevant facets. In this structure, textures are associated to areas delimited by the curves.

In section 2, we describe the new data structure added in the facets structures, and the pre-processing that is achieved in order to represent the texture boundaries. In section 3, we detail how the texture is determined at rendering time, for facets containing several textures.

2 Pre-processing

At modeling time, the user has to specify one or more closed 2D curves on the surface of an object to be textured. We assume that we have such 2D closed curves, their specification will not be addressed in this paper. This can easily be done interactively, e.g. by using the IRIS hardware ability to give the 3D coordinates on any pixel (or (u, v) values, as done in [HH90]). The curves may also be obtained by an implicit function or a procedural model like solid textures [Per85].

Before rendering, the curves are polygonalized, clipped and stored in the facets they cross. Numerous clipping algorithm are available [FvDFH90], we do not describe this task here. The coordinates of the clipped closed curves are converted into the parametric coordinate system of the facet, which is easy to do for triangle or quadrilateral (there is a linear 2×2 system to solve for triangles, and a 2×2 Q_1 system[1] for quadrilaterals).

The first data structure stored for each facet is a list of included areas. An area can be either simple (delimited by a curve) or composed (by combining simple areas). This allows

[1] A Q_1 polynomial is of degree one in each variable, i.e. the highest degree term is $u.v$.

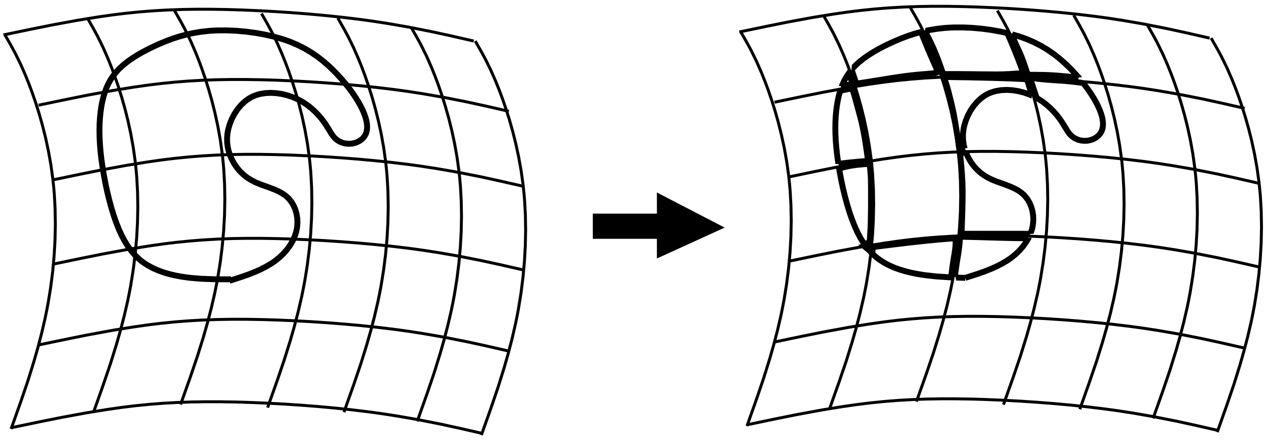


Figure 1: Clipping of the 2D boundary in each facet containing the curve.

the designer to use union and intersection operators to specify complex areas. In order to save computations, an IF operator is also provided, which uses an area as selector to combine either a list of areas or another (the selector is an area bounding the first list, thus separating the elements in the same spirit as BSP-trees). Each entry of this list contains:

- a state, that will indicate at rendering time if the point to be rendered is inside or outside the area (thus valid), or not yet determined (thus invalid),
- a category, indicating if the area is inside or outside the curve, or if this area is composed,
- the parameters defining the area, which are a vertex list for a simple area corresponding to a curve, or a list of areas and operators like NOT, AND, OR, IF for a composed area. The areas in the list can either be simple or composed. (AND or OR are placed on first position, indicating how to combine the list. NOT can be placed before either area number. IF is followed by the selector area number, and an offset pointing to the areas list to use when the state of the selector is 'outside'.)

The second data structure is a list of the textures availables for this facet. Each entry contains:

- the texture number for this entry (note that several entries may release the same texture),
- two areas numbers a and b,
- the operation to achieved on these areas to obtain the area to fill with the texture (the operator can be 1, a, !a, a&b, !a&b, a&!b, !a&!b, a|b, !a|b, a|!b, !a|!b, where '!' means NOT),
- the coefficient to apply to this texture if several textures are superposed on the same place.

Most of the facets do not contain the texture boundary. Their area list is empty, and their texture list contains a single entry containing the texture number and the operator '1'.

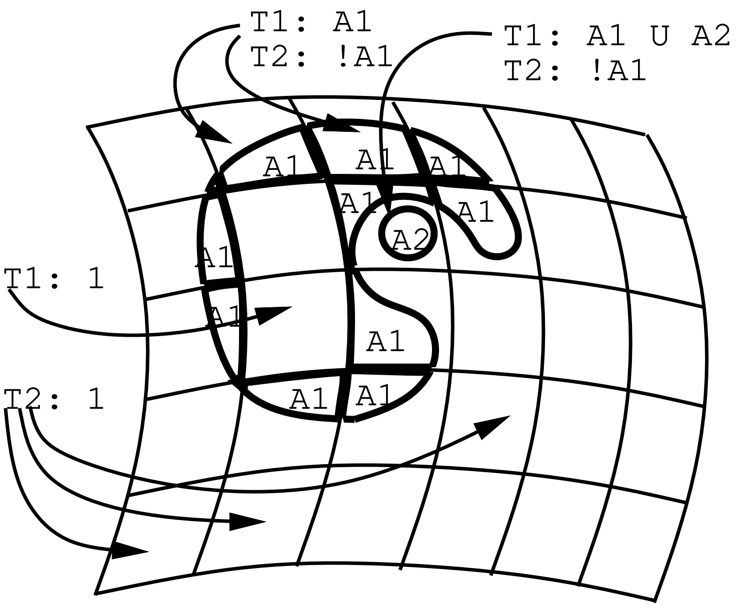


Figure 2: Area list and texture entries associated to each facet ('!' means NOT).

3 Processing at rendering time

The rendering algorithm can be a ray-tracer or a scan-liner (for the latter, our algorithm can be further optimized). The color has to be evaluated for the current point. Before the local illumination computation, the material parameters have to be evaluated if they are defined by a texture (either a map or a procedure). When the current facet contains several textures, we have to check for the one to use (or possibly the set of textures to superpose).

The state of each area list entry is first invalidated, then the texture list of the facet is parsed. For each entry, the texture has to be applyied if the associated area covers the current point. To determine this, the state of the two areas parameters is checked and the indicated operation is evaluated. Checking the state of an area consists in reading the state written in the entry corresponding to this area in the area list, if it has still be validated, otherwise it has to be evaluated. For a composed area, the states of the areas parameters are checked and combined as indicated in the definition. For a simple area, a classical test[FvDFH90] is achieved to determine if the current point is inside or outside the curve (by testing the parity of the number of intersections between the curve and an horizontal half-line on the right of the point[2]).

Note that areas are evaluated once, and only if they are used. Using the IF operator with a selector area thus allows to evaluate only the curves that are on a side of the selector.

[2]The point is inside if the parity is odd. The intersection is a very simple computation that has to be done only for segments of the curve which ordinates bound the point ordinate.

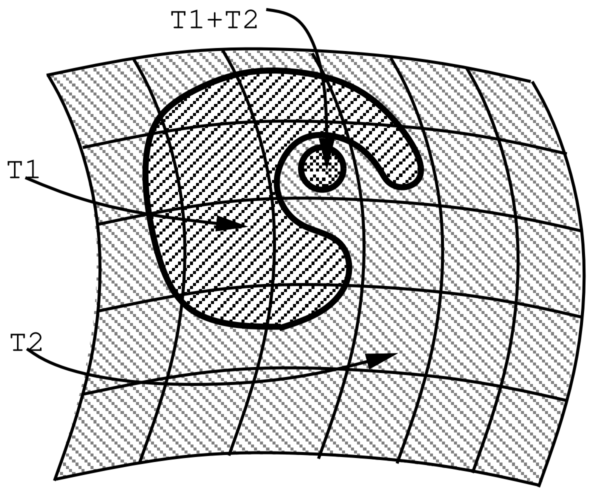
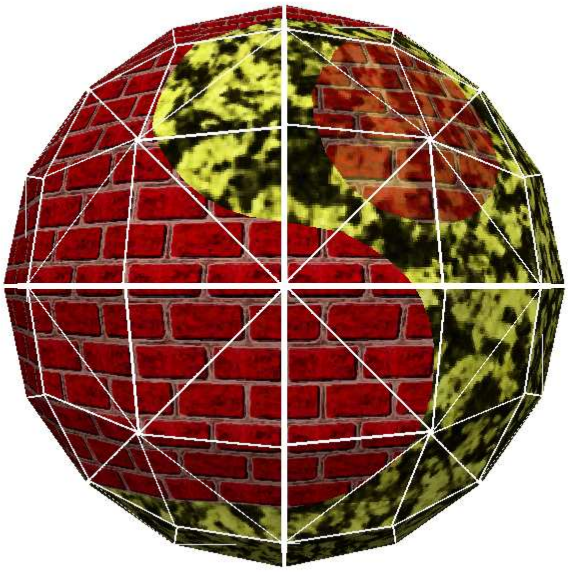


Figure 3: Resulting texture determination (as defined in the previous draft). The first texture is inside the curves, and the second one is outside the long curve, so that the texture are superposed inside the small area.

4 Result

The following image is computed in ray-tracing. The sphere is triangulated, each facet being shrinked in order to make the mesh visible. Two textures are used, and two curves are defined. Note that the two textures are superposed in the small circular area.

The trimming texture method we have described is very easy to implement, does not modify the mesh, and introduces an extra cost only for facets containing a texture boundary. Moreover, it is compatible with standard geometric and textural representations. The basic idea is to solve in 2D the texture boundary determination which is a 2D problem. We may also define a transition, in order to switch smoothly from one texture to another. Note that some other 2D problems may be addressed in the same way, e.g. for the smooth shadow boundaries encoding on the surfaces in radiosity algorithms. Mapping optimisation techniques, such as *atlas* [MYV93], may also become more precise, by relaxing the constraint of having the texture pieces corners on the vertice of the mesh.



References

[CR87] Gary A. Crocker and William F. Reinke. Boundary evaluation of non-convex primitives to produce parametric trimmed surfaces. In Maureen C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 129–136, July 1987.

[FvDFH90] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practices (2nd Edition)*. Addison Wesley, 1990.

[HH90] Pat Hanrahan and Paul E. Haeberli. Direct WYSIWYG painting and texturing on 3D shapes. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 215–223, August 1990.

[MYV93] Jérôme Maillot, Hussein Yahia, and Anne Verroust. Interactive texture mapping. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 27–34, August 1993.

[Per85] Ken Perlin. An image synthesizer. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19(3), pages 287–296, July 1985.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399