Warped Textures for UV Mapping Encoding

Olga Sorkine and Daniel Cohen-Or

The School of Computer Science, Tel-Aviv University, Israel

Abstract

This paper introduces an implicit representation of the u,v texture mapping. Instead of using the traditional explicit u,v mapping coordinates, a non-distorted piecewise embedding of the triangular mesh is created, on which the original texture is remapped, yielding warped textures. This creates an effective atlas of the mapped triangles and provides a compact encoding of the texture mapping.

1. Introduction

In the past few years, much effort has been devoted to the development of mesh compression techniques, as the need to transfer 3D models over communication channels constantly grows with the evolvement of Web-based applications[3, 8, 9, 10]. The attention of the researchers has been focused on encoding polygonal meshes, in particular, triangular meshes. The representation of a mesh consists of set of vertex coordinates, known as the *geometry*, and the vertex/triangle adjacency list, known as the *topology*. While most of the attention has been devoted to the efficient encoding of the topology, the geometry is commonly encoded by some variation of delta-encoding, exploiting the intrinsic coherence among the vertex coordinates. Such delta-encoding can be applied to any attributes associated with the vertices as long as they have similar spatial coherence to that of the vertices. In particular, this is true for the u,v texture coordinates, which are associated with each vertex of a textured mesh.

Encoding texture coordinates is important, since textured models are widespread in the commercial and entertainment world, and their size is a significant portion of the mesh representation. In this paper, we investigate a unique approach for encoding the texture mapping. The new approach is not based on delta-encoding, and moreover, avoids the explicit encoding of texture coordinates altogether.

The u,v coordinates are an explicit representation of the texture mapping, defining the source location of each vertex in the texture domain. Let $T : R^3 \rightarrow R^2$ be the mapping function from the vertex coordinates space to the texture u,v space. Traditionally, T is described *explicitly*: for each vertex point in R^3, the corresponding mapping location in R^2 is stored. Given a triangular mesh $M^3 \subset R^3$, T embeds the triangles into a triangular mesh $M^2 \subset R^2$. We define an *implicit* representation of T by a composition of two mappings, $W \circ H$, where H is an *embedding* mapping from M^3 to $M'^2 \subset R^2$, and W is a warping function from M'^2 to M^2. The representation of T by $W \circ H$ is space efficient, since W is applied in a preprocess which warps the original texture into a *warped texture*, and H is defined by the geometry of M^3. This is illustrated in Figure 1, where a simple object (pyramid) consisting of four triangles is displayed in (a), and its surface is mapped onto the texture in (b) by the T mapping. The warped texture defined by the embedding of the pyramid onto the plane is shown in (c). Since the texture space is discrete, H and W should be chosen carefully to avoid sampling problems. In this paper, we construct an implicit H mapping which guarantees a proper sampling by the warping function W.

The paper is organized as follows. In the next section, we briefly discuss the embedding problem. In Section 3 we introduce a technique that constructs a proper embedding H. Preliminary results are discussed in Section 4.

2. Background

Embedding, or flattening, of a 3D mesh is a mapping of the 3D vertices onto the 2D plane, so that the topology of the mesh is preserved. Non-distorting embedding has been of much interest to the computer graphics community, since it provides a solution to different problems. In texture mapping applications, the flat texture is mapped onto the 3D surface by embedding the 3D surface triangles into the texture space.

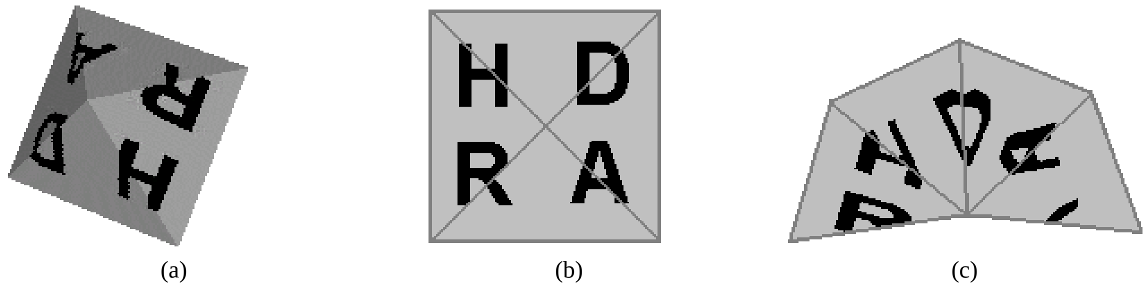

Figure 1: *A simple example of the remapping method: (a) A view of the textured 3D mesh, (b) The original texture mapping, (c) The warped texture*

In this context, various embedding techniques have been developed, which strive to flatten the surface while minimizing the deformation of the triangles using some global deformation metric.

Eck et al.[4] treat the mesh as a system of springs, lying on the edges of the mesh, which obey some physical laws. The embedding is computed using harmonic maps, i.e. the 2D positions of boundary vertices are set along the unit disk and inner vertex positions are calculated while minimizing the total energy of the system. The resulting embedding minimizes the metric dispersion of the mesh. Maillot et al.[7] partition the mesh into several areas using curvature information, and apply harmonic maps to each part, thus making a compromise between discontinuities of the mapping and image distortion. Other works minimize the distortion by preserving geodesic distances between vertices. Bennis et al.[2] first flatten an isoparametric curve on the surface with respect to the curvature and chord length and then unfold the surface around it, until a certain distortion threshold is reached. Floater[5] embeds a 3D mesh by first setting the positions of its boundary vertices along the boundary of a flat convex polygon (such as a rectangle or a circle), and then forcing the inner vertex positions to be placed in a convex combination of their neighbors, while preserving the edge lengths as much as possible. These constraints define a linear system of equations for the flat vertex positions. Levy and Mallet[6] attempt to preserve perpendicularity and distance between isoparametric curves on the surface by posing another set of linear constraints which minimizes a *roughness* criterion. Azariadis and Aspragathos[1] choose to decompose the mesh into parallel strips of triangles. Each triangle strip can be flattened by an isometric mapping to the 2D plane, i.e. no distortion is caused to the triangles. The resulting texture mapping, however, is not continuous along the strip borders. To solve this problem, the gaps between neighboring strips are closed, which distorts the flattened triangles.

All the above techniques focused on the local behavior of the mesh in order to prevent stretching of the texture. Therefore, the relations between neighboring triangles were important. In contrast to those techniques, in our application, we are concerned with the behavior of each individual triangle and thus, our embedding procedure guarantees that none of the triangles is distorted over some predefined threshold.

3. Non-distorting Embedding

Most known methods flatten the surface in one piece to avoid discontinuities in the texture mapping along the seam lines. In our case, flattening is used to embed an existing texture mapping, therefore we can allow the embedding to be broken into several patches. However, we would like to minimize the number of patches by generating patches as large as possible, so that their overall shape is compact. That is, the collection of textured patches should be contained in rectangular images so that their size is as small as possible.

Recall that our primary requirement is that none of the triangles is distorted above some threshold. One way to flatten a mesh without distorting the triangles is by simply recursively unfolding the triangles onto the plane. This, in general, yields overlapping of unfolded branches on the one hand and gaps among the branches on the other hand.

We use an iterative algorithm, which flattens each triangle with distortion below a certain predefined threshold. The process is as follows: an initial seed triangle is mapped to the plane as is. Then, we proceed iteratively by flattening its topological neighbors in bread-first-search order and growing a patch P around the seed triangle. At each step, we find the set T of k triangles that share an edge with the triangles of P and attempt to flatten each of these candidate triangles. This implies a set of new candidate vertices to be added to P. Given a candidate vertex q and its associated triangles $\{t_j\} \in T$, we compute a set of positions $\{q_j\}$ in the plane, such that q_j is located by unfolding t_j to the plane. The initial position q' of q in the plane is the weighted average of q_j, where the weights are defined by the distortion of the supporting edges of t_j in P. The final position of q' is determined by applying a relaxation, which minimizes the distortions of the triangles t'_j. (See Figure 2) If the position of q' causes one of its associated triangles to deform above the threshold or to overlap with P, the vertex is discarded from P, and will

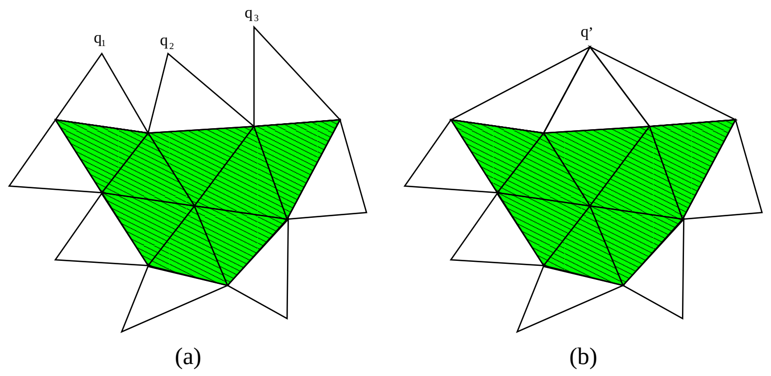

Figure 2: *Growing a patch: the dark triangles were flattened in previous iterations. The white triangles are the candidates for embedding in the current iteration. In (a), q_1, q_2, q_3 are the unfolded 2D positions of the candidate vertex q. In (b), q' is the average 2D position of q.*

be embedded in another patch. When the algorithm cannot add any vertex to P, it stops, and a new patch starts.

The distortion caused by embedding a triangle, is measured by the following expression. Let e_1, e_2, e_3 be the edges of the original triangle t and e'_1, e'_2, e'_3 the corresponding edges of the embedded triangle t'. Denote by L_i and l_i, $i = 1,2,3$, the maximal and the minimal edge lengths, respectively. That is, $L_i = \max(len(e_i), len(e'_i))$ and $l_i = \min(len(e_i), len(e'_i))$. We define the distortion as the maximum ratio between the edge lengths:

$$distortion(t,t') = \max_i(\frac{L_i}{l_i}),\ i = 1,2,3.$$

Note that $distortion(t,t') \geq 1$, and $distortion(t,t') = 1$ if and only if t and t' are isometric, since two triangles are isometric iff their edge lengths are equal.

4. Preliminary Results

The piecewise non-distorting embedding algorithm has been developed in C++. We have tested it on several meshes, in particular, non-trivial ones which are not topologically equivalent to a disk or to a sphere. For example, the twisted loop in Figure 3 has a complex surface which cannot be flattened into one piece, and moreover, the surface curvature is relatively high. By applying our algorithm, the surface is decomposed into several patches, two of which are shown in Figure 3(b-c).

The twisted cone (see Figure 4) is an example of another non-trivial surface, since the size of its triangles is non-uniform and the curvature varies over different regions. This surface can be embedded to the plane in one piece, but with significant distortion of the triangles. By applying a distortion threshold of 1.4, the surface is flattened into two pieces, shown in Figure 4(c). Figures 4(a),(d) show the twisted cone with texture and the two pieces of the warped texture, respectively.

We have tested our algorithm on several textured meshes with different distortion thresholds. The lower the maximum distortion is, the more patches are produced, which enlarges the total size of the images. We have found that a maximum distortion of 1.3 provides a good balance.

To generate a compact representation of the warped texture, the patches are rotated to fit into a minimal bounding rectangle. The size of the warped textures is not necessarily larger than the original texture, since in many cases, the original texture has redundant parts and is not fully mapped. In fact, our algorithm creates an effective atlas of the mapped triangles. For example, the original JPEG size of the texture used in Figure 4 is 17.6KB, while the two warped textures' size is 17.9KB. The overhead of the warped textures' size is relatively small with respect to the u,v coordinates' size. For example, the twisted cone model consists of around 6000 triangles and the original u,v coordinates of the VRML representation can be encoded with binary representation into 7.4KB.

We are currently working on the optimization of the embedding algorithm, in the sense that the algorithm will yield a minimal number of patches. We are also considering developing an algorithm for compact packing of the patches.

Acknowledgments

We would like to thank Reuven Bik for his early work on this project, and Roman Manevich and Ronen Gvili for their contribution to the implementation of the algorithm.

References

1. Phillip Azariadis and Nikos Aspragathos. Design of plane developments of doubly curved surfaces. *Computer-aided Design*, 29(10):675–685, 1997. 2
2. Chakib Bennis, Jean-Marc Vézien, Gérard Iglésias, and

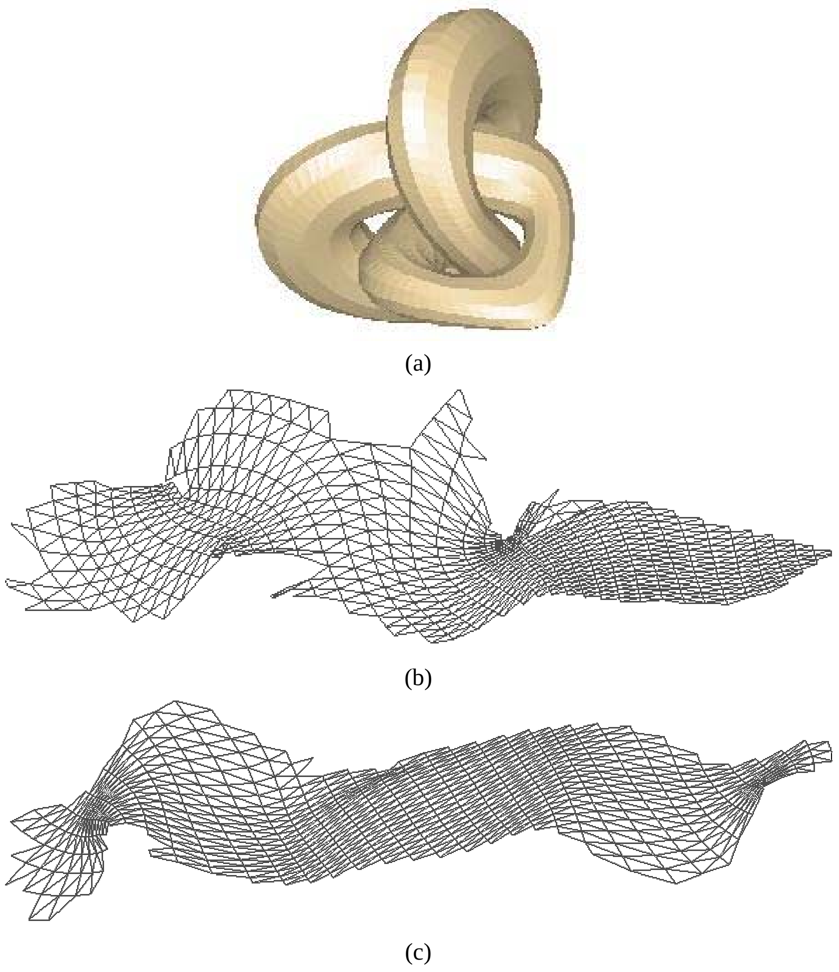

Figure 3: *Embedding of the twisted loop model. The 3D mesh and two of the patches created by the embedding algorithm.*

André Gagalowicz. Piecewise surface flattening for non-distorted texture mapping. *Computer Graphics (Proceedings of SIGGRAPH 91)*, 25(4):237–246, July 1991. 2

3. Daniel Cohen-Or, David Levin, and Offir Remez. Progressive compression of arbitrary triangular meshes. *IEEE Visualization '99*, pages 67–72, October 1999. 1

4. Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Michael Lounsbery, and Werner Stuetzle. Multiresolution analysis of arbitrary meshes. *Proceedings of SIGGRAPH 95*, pages 173–182, August 1995. 2

5. Michael S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(3):231–250, 1997. 2

6. Bruno Lévy and Jean-Laurent Mallet. Non-distorted texture mapping for sheared triangulated meshes. *Proceedings of SIGGRAPH 98*, pages 343–352, July 1998. 2

7. Jérôme Maillot, Hussein Yahia, and Anne Verroust. Interactive texture mapping. *Proceedings of SIGGRAPH 93*, pages 27–34, August 1993. 2

8. Renato Pajarola and Jarek Rossignac. Compressed progressive meshes. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):79–93, January - March 2000. 1

9. Gabriel Taubin, André Gueziec, William Horn, and Francis Lazarus. Progressive forest split compression. *Proceedings of SIGGRAPH 98*, pages 123–132, July 1998. 1

10. Costa Touma and Craig Gotsman. Triangle mesh compression. *Graphics Interface '98*, pages 26–34, June 1998. 1

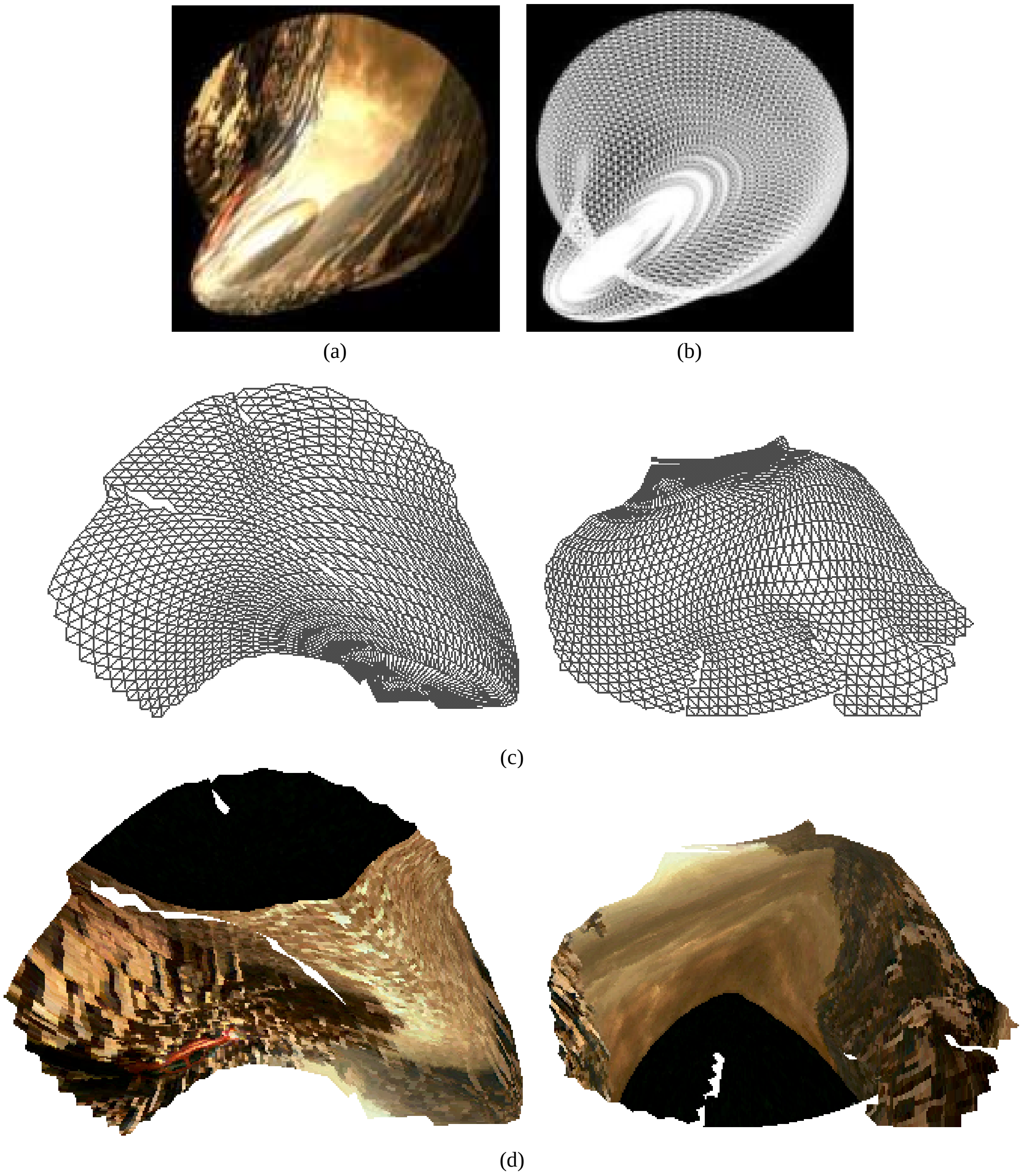

Figure 4: *The results of the twisted cone model embedding. (a) The textured 3D mesh; (b) 3D mesh in wireframe; (c) The two patches created by the embedding algorithm; (d) The warped textures.*