MVA2000 IAPR Workshop on Machine Vision Applications, Nov. 28–30, 2000, The University of Tokyo, Japan

2—3 On the Precision of Textures*

Frank Nielsen† Nicolas de Mauroy‡

Abstract

In this paper, we first introduce the notion of texture precision given the 3d geometry of a scene. We then provide an algorithm to acquire a texture/color map of the scene within a given precision. The texture map is obtained using pinhole sensing devices from data either acquired in the real world or computer-synthesized. Finally, we describe a procedure to obtain level of precisions by combining a modified edge-collapse geometry technique with an appropriate remapping texture engine. We report on our experiments and give perspectives for further directions of research.

1 Precision of textures.

There is a considerable amount of literature on level of details (LODs) and texture map simplifications (see [2, 3, 4, 5, 6, 7]). Those methods generally proceed by dividing the object into patches and then, by simplifying the geometry by minimizing the texture distortion/displacement using energetic functions on each patch. As a matter of fact, the borders of the patches are not allowed to be touched in order to respect the boundaries of those patches. Roughly speaking, we adopt the following scenario: given the known geometry of an object, how many pictures (and the corresponding locations of the camera) do we need to take in order to build a texture/color map that has a guaranteed precision. Each picture defines a non-necessarily connected super patch. Potential applications of our method include range scanning where we may constrain the positions of the camera to be on a circle centered around the object (ie., the axis of revolution of a turntable): we then scan the object in two steps: (1) get the 3d coordinates of a triangle mesh, and (2) select positions of the camera in order to build a texture map within a prescribed quality. Figure 1 points out the pixel distortions obtained from the perspective projection of a checkboard on a 3d mesh.

In the sequel, objects $\mathcal{O}$ are considered to be tesselated manifolds with boundaries of $\mathbb{R}^3$. We define

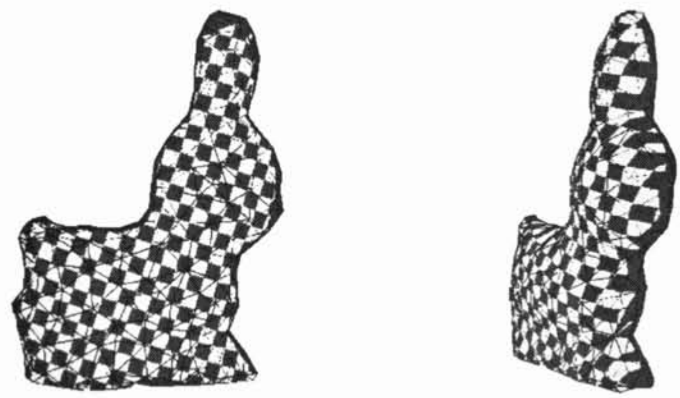

Figure 1: Projecting a checkboard on a 3d mesh; Observing the pixel projections from two different viewpoints.

a texture as an application $\mathcal{T}$ which associates to each point of $\mathcal{O}$ a scalar value (or attribute). Let $\mu(R)$ defines the measure on $\mathbb{R}^3$ defined by the diameter of R. A texture $\mathcal{T}$ has precision ϵ if we can find a *covering* $\mathcal{C} = \{c_1, ..., c_k\}$ of $\mathcal{O}$ (ie., $\bigcup_i c_i = \mathcal{O}$) such that:

- Each element $c_i \in \mathcal{C}$ of the cover has measure less than ϵ (i.e. $\mu(c_i) \leq \epsilon$), and
- For each element $c_i \in \mathcal{C}$, $\mathcal{T}(x)$ is constant for all $x \in c_i$.

For a given 3d location and orientation L of a camera and a point $p \in \mathcal{O}$, the precision of p is either infinite if no pixel of the camera projects onto p, or equals to the measure of the projected pixel. We denote by $\mu_L(p)$ the precision of p from camera location and orientation L. A triangle $\mathbf{t} \in \mathcal{O}$ has precision ϵ if $\forall p \in \mathbf{t}, \mu(p) \leq \epsilon$. Let p_1, p_2 and p_3 be the vertices of $\mathbf{t}$. If $\forall i \in \{1, 2, 3\}$, $\mu(p_i) \leq \epsilon$ then $\mu(\mathbf{t}) \leq \epsilon$. (See other notations[1] and proofs in [1].)

* Full version available as a technical report [1].

† Sony Computer Science Laboratories Inc, FRL. E-mail : nielsen@csl.sony.co.jp

‡ École Polytechnique, Palaiseau, Paris, France.

[1] Due to lack of space, we do not distinguish interior points from points on the boundaries here.

2 Acquiring a triangle mesh within a given precision.

We present below a heuristic for capturing a triangle mesh within precision ϵ:

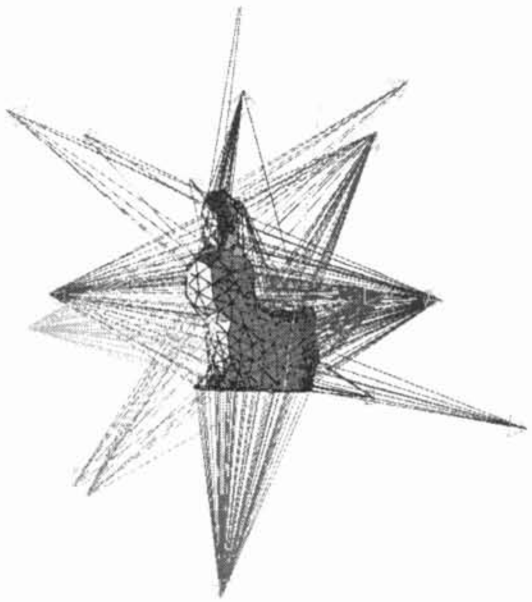

Figure 2: A set of 18 cameras covering 98% of a 800-triangle bunny model. Each position of the camera defines a set of triangles having precision at least ϵ.

1. Find a set of camera positions (i.e. camera locations and attitudes) $\mathcal{L}$ so that each triangle can be "seen" at least once within precision ϵ. (See Figure 2)
2. Select a subset $\mathcal{L}' \subseteq \mathcal{L}$ so that all triangles can be seen within precision ϵ. (This amounts to a set cover problem [8].)
3. Take the "pictures" from $\mathcal{L}'$ and create the corresponding texture map (see Figure 3).

Step 1 is a *preselection* step. A naive approach consists of regularly sampling (grid-like) a bounding box centered around the object. Then at each point of the grid we sample the orientations of the camera. Sampling each parameter p times yield a set $|\mathcal{L}| = O(p^6)$ of camera positions that are not necessarily covering all the object. We investigated several heuristics (see [1]) that cover usually 99% of the object with linear order of camera positions. (We handle the few not-yet-covered triangles one by one.) Note that the combinatorial complexity of the visibility graph of a n-vertex nonconvex polyhedron can be as large as $\Theta(n^9)$ [9].

Step 2 is a *set cover problem* in disguise [10]. Indeed, let $L(\mathcal{O}) = \mathbf{t}(L)$ be the set of triangles of $\mathcal{O}$ having precision at most ϵ for a given camera position and attitude L. We want to minimize $|\mathcal{L}'|$ such that $\bigcup_{L\in\mathcal{L}'} \mathbf{t}(L) = \mathcal{O}$. Note that we can associate a cost (penalty) $w(L)$ to each camera position and ask for minimizing $\sum_{L\in\mathcal{L}'} w(L)$ as well. Step 1 is the most delicate part since we need to find for each triangle t a position L such that t is fully visible from L (ie., not occluded by the object itself) and $\mu_L(t) \leq \epsilon$. In most cases, where we sampled the combinatorial space of camera positions, we prefer to solve a *partial set cover*, where we first ask to cover at least a fraction of the triangles (using configuration space, say $\mathcal{L}_1$). We then compute for the not-yet-covered triangles corresponding camera positions and attitudes (configuration space, say $\mathcal{L}_2$), and finally solve the *set cover problem* on $\mathcal{L}_1 \cup \mathcal{L}_2$. Step 3 acquires the pictures (either in the real world or by computer simulations, eg. raytracing). In the latter case, we can choose appropriate projection models like orthographic projection, etc. The portions of each picture combine in the overall texture map using 2d bin packing [11] as depicted in Figure 3.

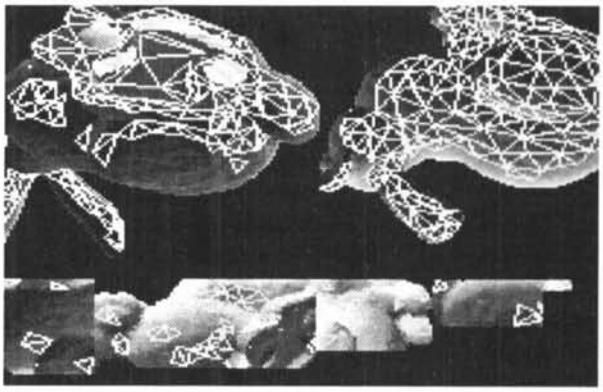

Figure 3: Acquired pictures that combine into a texture map (we only show here a cropped texture image). Bounded triangles indicate the ones that have the desired precision.

The preselection step is important in practice since it samples the configuration space which otherwise will be too costly to compute. Let f denotes the focal length of a pinhole camera, and d the size of the pixels. Given a precision ϵ, we define the α-distance as $\frac{\alpha\epsilon f}{d}$.

Below, we report on several heuristics:

Spherical discretization: This heuristic selects cameras on a sphere centered at the object with ra-

Object	Standard sphere	Triangle normal	Simplified triangle normal
A	1470 (99%)	1476 (99%)	1474 (99%)
B	1493 (97%)	1518 (99%)	1526 (100%)
C	1587 (97%)	1634 (100%)	1634 (100%)

Figure 4: Efficiency of the preselection heuristics. **A** is a bunny model (1477 triangles), **B** is a Champagne cup (1526 triangles) and **C** is a compund scene of 4 animals (1634 triangles).

dius the α-distance and camera orientations pointing to the center. The surface of the sphere is then discretized both in lattitude and longitude. Although this heuristic is very simple, we obtain good results in practice for values of α around 0.6 and a hundred points sampled on the sphere.

Triangle normal method: For each triangle of the object, we put a camera in the point c such as, if g is the centroid of the triangle, cg is a normal to the plane of the triangle, c is in the outside side of the triangle, and $|cg|$ is the α-distance. This algorithm works slightly better than the "spherical discretization", but for large objects, such as 100000 polygon objects, the size of the preselection set is prohibitive.

Simplified triangle normal method: This method is roughly speaking the same as the previous one, except that we use a simplified version of the triangle mesh. So we only have one camera for every triangle of the simplified object which can have 10 or 100 times less triangles than the original model.

3 Building levels of precisions.

We extend the notion of precision ϵ of the texture to both the *geometry* and the *texture* (in order to reduce the number of vertices and henceforth triangles). Informally speaking, we want to approximate the object so that one cannot distinguish the simplified from the original one by taking pictures having resolution at most ϵ.

Our algorithm works as follows:

1. Simplify the triangle mesh within some precision ϵ'. (We use a modified edge-collapse algorithm that do not necessarily preserve the topology as we wished.)

2. Compute, for the simplified triangle mesh, the positions of the camera in order to acquire the whole object within some precision ϵ''.

3. Take pictures of the original object at the positions computed in Step 2 and build the overall texture map.

Theorem.
A simplified object satisfying the following conditions is an ϵ-approximation:

- The symmetric Haussdorf distance between the simplified and original model is at most $\epsilon' = \frac{\epsilon}{2}$.

- The texture is acquired within precision $\epsilon'' = \frac{\epsilon}{2}$.

- The angle from any ray emanating from a camera to the normal of a selected triangle is at most $\frac{\pi}{4}$.

We add the constraints on incidence angle in Step 2 of the simplification algorithm: that is in finding initially a set of positions and attitudes of the camera (We leave the proofs in [1].) Figure 5 shows a hierarchy of levels of precisions.

4 Concluding remarks.

Our methods extend naturally to the simplification of already captured color map objects: Given a 3d textured model, we first compute its precision and then build a hierarchy of levels of precisions. Figure 6 shows such an example. Our algorithms extend among others to scene illuminations and positions of cameras for telesurveillance planning. However, our model does not take into account the reflectance of the objects since we assume that ray emanating from a same surface point have the same color. Future research may focus on the reflectance acquisition and progressive coding of LOPs using this notion of precision.

References

[1] N. de Mauroy and F. Nielsen. On the precision of geometric objects and some of its algorithmics. Technical Report CSL-99, Sony Computer Science Laboratories, FRL, 1999.

[2] Makoto Maruya. Generating a texture map from object-surface texture data. *Computer Graphics Forum*, 14(3):397–406, August 1995. Proceedings of Eurographics '95. ISSN 1067-7055.

[3] Marc Rioux Marc Soucy, Guy Godin. A texture-mapping approach for the compression of colored 3D triangulations. *The Visual Computer*, 12(10):503–514, 1996. ISSN 0178-2789.

[4] A. Certain, J. Popovic, T. DeRose, and T. Duchamp. Interactive multiresolution surface viewing. *Computer Graphics*, 30(Annual Conference Series):91–98, 1996.

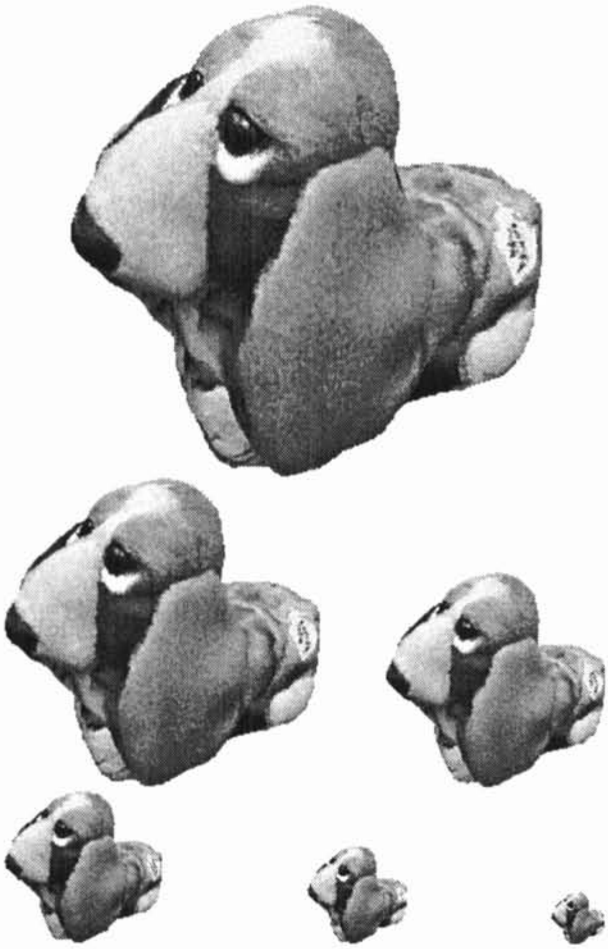

Figure 5: Level of precision (LOP) hierarchy: 8251 triangles (200K color map), 2925 triangles (120K), 2238 triangles (71K), 1567 triangles (33K), 1034 triangles (18K), 425 triangles (6K). Objects are displayed at scales where there are *perceived* with same quality.

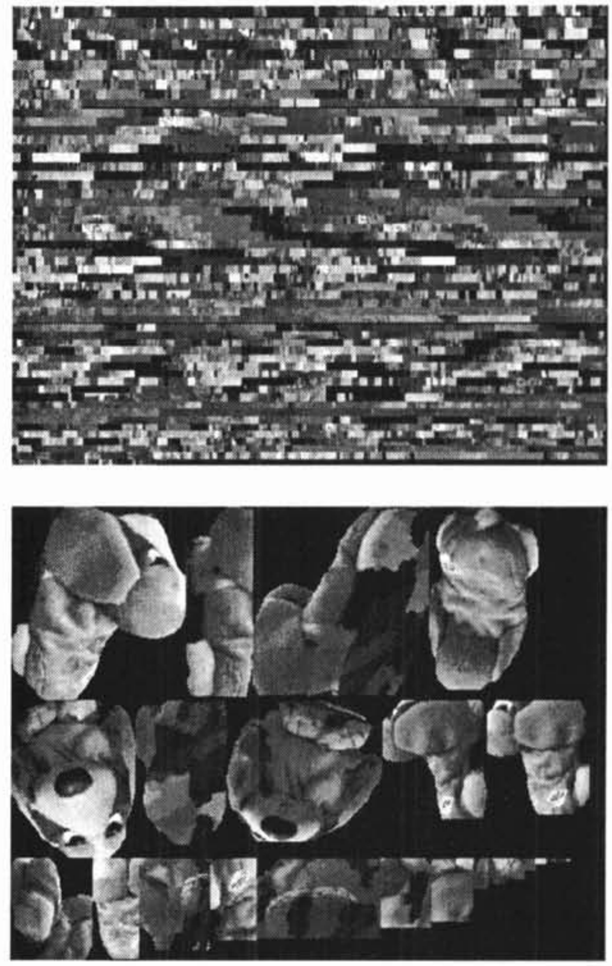

Figure 6: Top: Texture map of a 3d textured object. Bottom: Synthesized texture map with guaranteed precision.

[5] J. Cohen, M. Olano, and D. Manocha. Appearance-preserving simplification. In *Proc. SIGGRAPH '98*, Computer Graphics Proceedings, Annual Conference Series, pages 115–122, July 1998.

[6] Paolo Cignoni, Claudio Montani, Claudio Rocchini, and Roberto Scopigno. A general method for recovering attribute values on simplified meshes (color plate p. 518). In *Proceedings of the 9th Annual IEEE Conference on Visualization (VIS-98)*, pages 59–66, New York, October 18–23 1998. ACM Press.

[7] Hugues H. Hoppe. New quadric metric for simplifying meshes with appearance attributes. In David Ebert, Markus Gross, and Bernd Hamann, editors, *Proceedings of the 1999 IEEE Conference on Visualization (VIS-99)*, pages 59–66, N.Y., October 25–29 1999. ACM Press.

[8] Dorit S. Hochbaum. Efficient bounds for the stable set, vertex cover and set packing problems. *Discrete Appl. Math.*, 6:243–254, 1983.

[9] Z. Gigus, J. Canny, and R. Seidel. Efficiently computing and representing aspect graphs of polyhedral objects. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(6):542–551, June 1991.

[10] H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete Comput. Geom.*, 14:263–279, 1995.

[11] J. Csirik, J. B. G. Frenk, M. Labbe, and S. Zhang. On the multi-dimensional bin packing. *Acta Cybern.*, 1990.