Direction Fields over Point-Sampled Geometry

Marc Alexa Tobias Klug Carsten Stoll

Interactive Graphics Systems Group
Department of Computer Science
Technische Universität Darmstadt
Fraunhoferstr. 5
64283 Darmstadt, Germany
{alexa,tklug,cstoll}@gris.informatik.tu-darmstadt.de

ABSTRACT

We describe techniques to establish local frames over point-sampled manifold surfaces. The tangential alignment of local frames is determined using a wave front algorithm starting from a set of pre-defined directions and conquering the remaining points. The repeated application of this algorithm is a relaxation procedure that maximizes coherence along the surface and minimizes the discrepancy to user defined or surface features. The resulting direction field serves as a parameterization for subsequent processes.

Keywords
Point-sampled geometry, surface processing, local frames, surface parameterization.

1. INTRODUCTION

Representing surfaces using point samples without additional topology becomes increasingly popular [Gro01]. Point-based methods seem to originate from the idea of using points as rendering primitives [LeW85, GrD98, PZ+00, RuL00, KaV01]. In addition, real-world geometry is often scanned [LP+01, RHL02] and, naturally, represented as a set of points (possibly together with other information). Consequently, surface processing methods that do not require an explicit topology are appealing (e.g. [PaG01, ZP+02]) because they avoid the conversion steps to and from other representations such as meshes.

A particular problem in surface processing is to find a suitable parameterization in order to apply techniques from the Euclidean domain to surfaces (e.g., texture mapping, fairing, filtering). Lots of work has been devoted to parameterize meshes [ED+95, Tau95, ZSS97, KC+98, LS+98, GSS99, GV+00]. Generating parameterizations for point-sampled geometry seems even more difficult than for meshes because of the missing neighborhood information among the points (see e.g. [PaG01, ZP+02, FC+02]).

Journal of WSCG, Vol.11, No.1., ISSN 1213-6972
WSCG'2003, February 3-7, 2003, Plzen, Czech Republic.
Copyright UNION Agency – Science Press

We propose to solve the parameterization and neighborhood ambiguities in one step by computing smooth direction fields over the surface. As a primitive operation to separate tangential from normal direction (i.e. if normals are not given as part of the data) we use the tangent plane fitting technique from Levin [Lev01,AB+01]. We smooth an arbitrary initial set of directions in tangential direction using relaxation techniques. A relaxation approach allows optimizing additional measures, for example, the discrepancy with principal curvature directions.

Involving the user in the specification process allows the specification of a mapping from one surface to another. In addition, user specification of discontinuities in the direction field makes the relaxation procedure more robust, as it avoids the problem of automatically generating the necessary discontinuities on shapes different from a torus.

If the direction field is smooth it can serve as a parameterization. Around a discontinuity, it essentially provides a polar coordinate system. If discontinuities are used to specify point-to-point correspondences, the direction field may completely specify a bijective mapping.

We demonstrate the use of direction fields with some applications, i.e. generating a radial parameter domain and texture synthesis on surfaces.

2. RELATED WORK

The idea of this work is to use direction fields as local parameterizations of point sampled geometry. This idea has been used for texture synthesis on manifold meshes.

Praun et al. [PFH00] let the user define the local frames of a few vertices. The remaining directions are computed using an RBF scattered data interpolation approach, where distances are determined using Dijkstra's algorithm on the edge graph.

Wei and Levoy [WeL01] use a relaxation technique to generate a smooth vector field. The energy functional takes into account different types of symmetries.

Turk [Tur01] exploits a multi resolution representation of the mesh and employs a push-pull technique to interpolate sparse user information about the vector field.

Local frames for point sample geometry have been established in the context of processing operations and scale-space representations.

Pauly and Gross [PaG00] grow patches of points that allow a projection onto a planar domain. Starting from a set of seeds, points are merged if their normal cone stays bounded. The set of planar parameter domains is used to define spectral transforms on the point-sampled geometry.

Zwicker et al. [ZP+02] define a discrete version of the surface's gradient using k-nearest neighbors. This gradient is used to define the parametrization as an optimization problem. The solution minimizes a measure of distortion of the gradient along the surface. The local parameterizations are used for local operations on the geometry or texture mapping operations.

Fleishman et al. [FC+02] build local frames that minimize a weighted least squares problem. The union of origins of the frames form a manifold surface, while each frame locally approximates the surface. The generation process of local frames defines a subdivision operation on the point set, which is used to build a multi scale representation of the surface.

3. COMPUTING DIRECTION FIELDS

The goal of the following procedure is to assign each point in a set P a direction that conforms with user-defined or automatically detected directions so that the discrepancy of close points is minimized. We assume that each point p_i has a corresponding normal n_i. If the normals are not given we compute them using Levin's projection operation [Lev01].

Any point might contain information about its (initial) direction d_i. Points may be constrained or

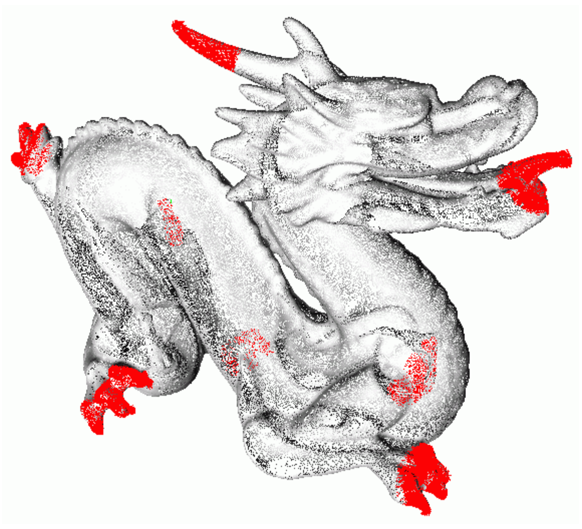


Figure 1. Dragon model with several user defined discontinuities while calculating the directions.

unconstrained., i.e. their direction is changeable during the process, or not.

The first phase of the process assignes each point p_i a direction d_i using a Dijkstra-like approach. In subsequent processing steps each point has a direction.

Specifying initial directions

The specification of initial directions can be done in different ways. In most application we expect the user to position the discontinuities like sources and sinks explicitly within the point set (see, e.g., Figure 1). A source or sink is defined by a position and a radius r. Any point within a sphere of radius r gets a direction that points either to the center of the sphere (sink) or away (source) from it. This direction is then projected onto the tangent plane of each point.

Initial directions could be also assigned by computing surface features such as principal curvature directions.

To start our process we need to assign at least one point a direction. If no preferred directions are specified, this could be done randomly.

Calculating initial directions

Let $I_0 \subset P$ be the set of points with an initial direction (constrained or unconstrained) and $O_0 \subset P$ be the set of points without information about the direction. In general, O_k and I_k are the sets after k iterations of the following procedure.

In each iteration we select a point p_j from O_k, calculate a direction for this point and move it from O_k to I_k.

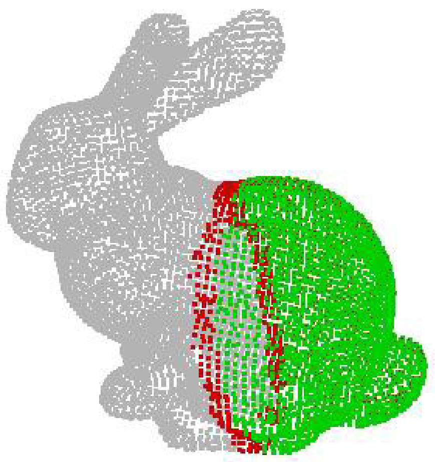


Figure 2. Directions are computed for each point iteratively. The sets O_k (grey) and B_k (red) contain points that need to be processed. The set I_k (green) represents points with direction information. The border B_k is used to speed up the determination of the next point to be treated.

Selecting the next point

The selection of the point for the next iteration is essential for the success of the algorithm and the quality of its output. It is necessary to select a point near points of I_k so that it is possible to determine a direction for the new point based on the directional information of the points in I_k.

The easiest way to ensure a sufficient neighbourhood of directions is to select the point from O_k that is closest to any point from I_k.

Our approach to this problem introduces another set of points B_k - the border set (see Figure 2). It contains all points in O_k that are near points from I_k. For each of these points a distance to the next point in I_k is stored. We select the point with the smallest distance.

Every point in the set I_k and B_k contains information about the distance *dist* to the closest point with an initial direction. The distance *dist* for points in B_k is calculated as

$$dist_k(p_j) = \min\{dist(p_i) + \|p_i - p_j\| \mid p_i \in I_k\} \quad (1)$$

The distances of points from I_k are final while those of the points in B_k may change if we find a shorter path to that point at any given time (resembling a Dijkstra-like strategy of computing shortest paths).

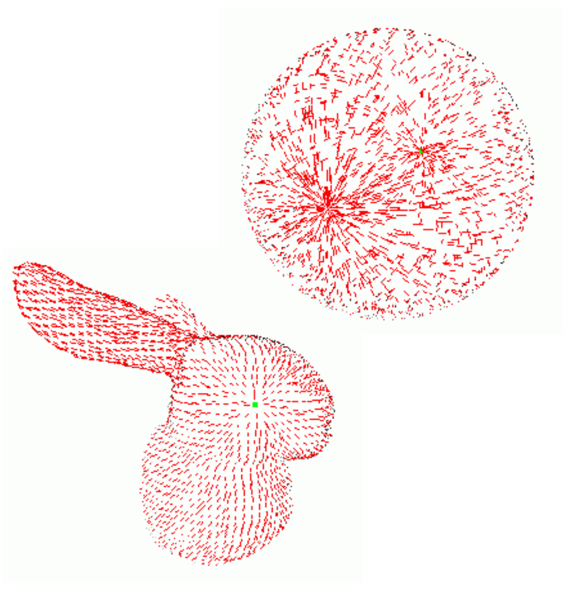


Figure 3. The direction field resulting from the first pass of the algorithm starting in one, respectively two, user specified discontinuities.

In every iteration we select that point from B_k that has the smallest *dist*. After adding the selected point to I_k we find the points with maximal distance h to the point in O_k and add them to B_k. This way the set of calculated points grows with constant speed into all directions.

Computing the new direction

In order to calculate the new direction we define a new set $T_{k,i,h}$ containing all points in I_k that are within a distance h from p_j. In practice this set is calculated using an octree structure for storing the positions of the points. The new direction d_j is calculated as the sum of the directions of all points in $T_{k,i,h}$ weighted using the Gaussians w_i.

$$T_{k,i,h} = \{p \in I_k \mid \|p - p_i\| < h\}$$

$$w_i(p) = e^{\frac{-\|p_i - p\|^2}{h^2}} \quad (2)$$

$$d'_k = \sum_{p_j \in D_{i,2h}} d_j * w_i(p_j)$$

The resulting direction is projected onto the plane defined by the normal of p_k and then normalized:

$$d_k = d'_k - \langle d, n_k \rangle * n_k$$

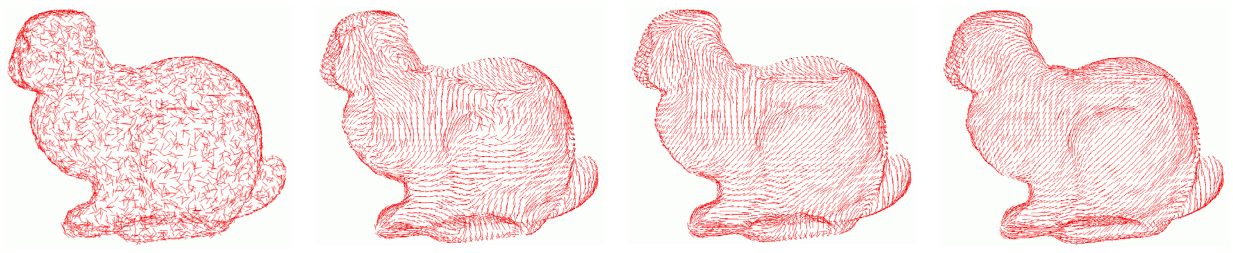


Figure 4. Starting from a set of random directions this point set is smoothed iteratively by repeatedly averaging directions of the points.

If either the sum of directions or the projected vector results to be zero, the result needs to be dropped and the point has to be processed again later.

These steps are repeated and the direction field grows uniformly from all specified initial directions (see Figure 1) until all directions are calculated. Results are depicted in Figure 3.

The choice of *h* is a factor for global smoothness of the directions generated. If *h* is small only few local points will be used to calculate the new direction and thus the directional field will only be locally smooth while a big *h* will lead to a global smoothness but has the deficit of slower processing.

Smoothing the directions

Since it may happen that the initial generation of directions does not result in a satisfactory smooth directional field it may be necessary to perform successive relaxation passes to smooth the directions to a certain degree.

These passes are calculated like the first, with the difference that all points are already contained in I_k. Because of this, all points have enough points in their environment which already have a direction, so the order of calculation is not as important as in the first pass.

We recalculate the direction for every point using Equation (2) but limiting the calculation to points that are unconstrained. Figure 4 shows the result of this procedure starting from a random set of directions.

Filtering a directional field several times in this way can lead to much better results in prospect of smoothness, especially if the factor *h* is increased for the passes, though even subsequent relaxation steps cannot resolve some problems arising through poorly positioned discontinuities.

4. APPLICATIONS

We demonstrate how to compute user-defined parameterizations for mapping one geometry to another. In addition we show how to use the coherent directions to generate globally smooth textures on point sampled geometry.

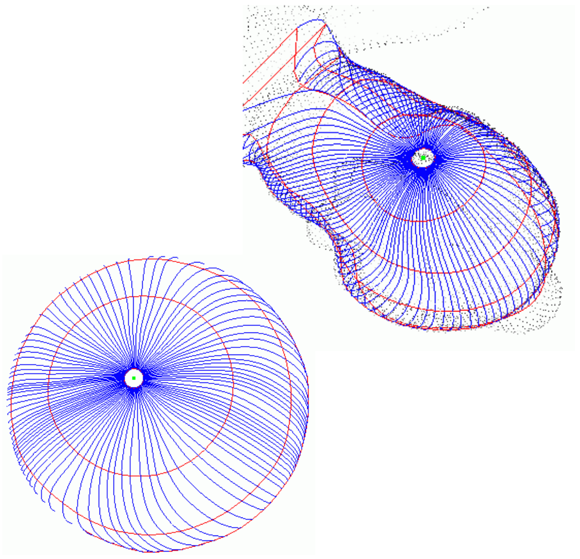
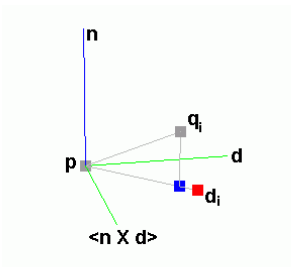


Figure 5. Specifying sinks and sources allows defining polar coordinate systems, which could be used to map information from one surface to another. If the same number of discontinuities is defined, a global mapping might be defined.

User-defined parameterizations

By defining the same number of sinks and sources in two models the resulting direction fields are topological equivalent (assuming the shapes have the same genus) and define a bijective mapping from surface to the other. This mapping is most easily exploited as local polar coordinate systems around sinks and sources. Figure 5 illustrates this idea for the Stanford bunny and a sphere.

By selecting a starting point *p* on the surface of the object we can travel along this surface by calculating the surface direction of this point, moving in this direction a fixed distance *e* and projecting this new point back onto the surface using Levin's projection operation [Lev01] (blue lines in Figure 5).



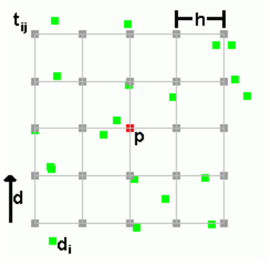


Figure 6a. Projection of the points onto the tangent Plane of p

Figure 6b. Schematic of the grid tij used for texture synthesis.

By starting in a sink or source of the direction field we can now parametrize the local surface around this discontinuity with polar coordinates as any point can be identified by an angle and a distance (found by backtracing the direction field from the point to the discontinuity).

The iso-contours (red lines in Figure 5) can be obtained by connecting points of the same distance to a discontinuity along several angles.

Texture synthesis

The calculated directions can be used to generate texture information for the point set surface by using an approach similar to [Tur01,WeL01].

We start in an initial point by assigning a random color from the texture. Selecting the point with the shortest distance to our initial point each step (similar to our method used in the directional field generation) a new color for that point is calculated by generating a NxN texture neighbourhood of the colors already generated and finding the point in the original texture that has the smallest error compared to the neighbourhood.

The calculation of the texture neighbourhood is done by projecting the direction between *p* and all surrounding points q_i onto *p*'s tangent plane. The resulting vectors d_i are normalized and multiplied by the distance between *p* and q_i. In this way we preserve a certain degree of distance information between the points on the surface, which would be lost by simply projecting them onto the tangent plane. (Figure 6a)

Now a regular grid *G* of NxN points t_{ij} with a distance of *h* (centered on *p*) is generated on the plane in alignment with the direction of the point and the vector orthogonal to the direction and the normal (in effect defining the up and right direction for the texture). (Figure 6b)

Each of these points t_{ij} is assigned the color of the closest point d_i. If no point d_i is closer than a certain

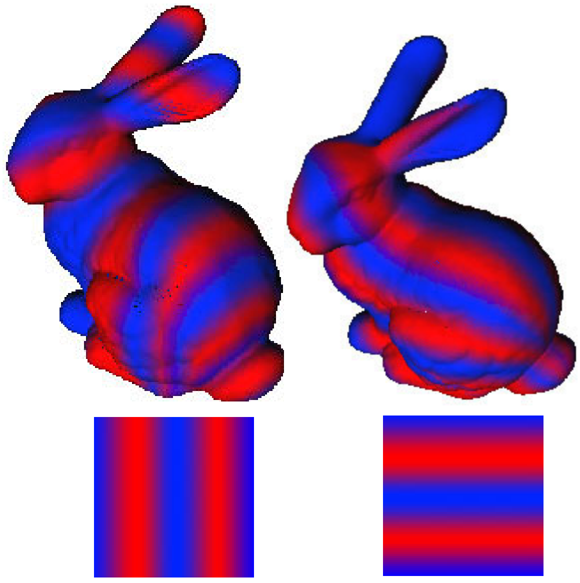


Figure 7. An anisotropic texture synthesized on the Stanford bunny. Note that the pattern are orthogonal and the textured bunnies together define local orthogonal grids on the surface.

multiple of *h* (usually 2) the point is marked as unassigned. The central point of the grid (being the one we want to calculate a color for) is also marked as unassigned.

The grid *G* is now compared to the source texture at all possible positions and the Gaussian of the error defines the likelihood for the position to be chosen. The color at the center of the grid is selected as the new color for our point. Figure 7 shows the Stanford bunny textured with anisotropic textures, several different isotropic textures synthesized on different point-sampled geometries are depicted in Figure 8.

5. CONCLUSIONS

We have presented an efficient technique to establish smooth direction fields over point-sampled geometry. Starting from a few user defined directions our approach typically establishes a smooth direction field in one pass over the surface. This process takes only a few seconds for all models depicted in this paper, which enables an interactive specification process.

Smooth direction fields serve as parameter domains for local synthesis operations. If discontinuities are placed adequately, direction fields might specify a bijective mapping from one surface to another.

In our future work we will exploit mappings among surfaces further. Together with hierarchical representations of point sets [FC+02] they would allow to apply linear techniques for the analysis and synthesis of shapes [PSS01]. With point sets, however, topological changes are possible, so that the linear space of shapes would no longer be fixed to a certain homotopy group.

6. REFERENCES

[AB+01] Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., Silva, T. Point Set Surfaces, *IEEE Visualization 01*, pp. 21-28, 2001

[ABK98] Amenta, N., Bern, M., Kamvysselis, M. A new voronoi-based surface reconstruction algo-rithm. *Proceedings of SIGGRAPH 9*8, 415–422, 1998

[ED+95] Eck, M., DeRose, T.D., Duchamp, T., Hoppe, H., Lounsberry, M., Stuetzle, W. Multiresolution analysis of arbitrary meshes. *Proceedings of SIGGRAPH 9*5, 1995

[FC+02] Fleishman, S., Cohen-Or, D., Alexa, M., Silva, C. Progressive Point Set Surfaces. *ACM Transactions on Graphics*, to appear

[Gro01] M. Gross. Are points the better graphics primitives? *Computer Graphics Foru*m, 20(3), 2001. ISSN 1067-7055.

[GrD98] Grossman, J.P., Dally, W.J. Point sample rendering. *Eurographics Rendering Workshop 199*8, 181–192, 1998

[GSS99] Guskov, I., Sweldens, W., Schröder, P. Multiresolution signal processing for meshes. *Proceedings of SIGGRAPH 9*9, 325–334, 1999

[GV+00] Guskov, I., Vidimce, K., Sweldens, W., Schröder, P. Normal meshes. *Proceedings of SIGGRAPH 200*0, 95–102, 2000

[KaV01] Kalaiah, A., Varshney, A. Differential point rendering. In *Rendering Technique*s, S. J. Gortler and K. Myszkowski, Eds. Springer-Verlag.

[KC+98] Kobbelt, L., Campagna, S., Vorsatz, J., Seidel, H.-P. Interactive multi-resolution modeling on arbitrary meshes. *Proceedings of SIGGRAPH 9*8, 105–114, 1998

[LS+98] Lee, A., Sweldens, W., Schröder, P., Cowsar, L., Dobkin, D. Maps: Multiresolution adaptive parameterization of surfaces. *Proceedings of SIGGRAPH 9*8, 95–104, 1998

[Lev00] Levin, D. Mesh-independent surface interpolation. Tech. rep., Tel-Aviv University. http://www.math.tau.ac.il/~levin, 2000

[LeW85] M. Levoy and T. Whitted. The use of points as a display primitive. Tr 85-022, University of North Carolina at Chapel Hill, 1985

[LP+01] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital Michelangelo project: 3d scanning of large statues. *Proceedings of SIGGRAPH 200*0, 131–144, July 2000

[Lin01] L. Linsen. Point cloud representation. Technical report, Fakultät fuer Informatik, Universität Karlsruhe, 2001

[PaG01] Pauly, M., Gross, M. Spectral processing of point-sampled geometry. *Proceedings of SIGGRAPH 200*1, 379–386, 2001

[PZ+00] Pfister, H., Zwicker, M., van Baar, J., Gross, M. Surfels: Surface elements as rendering primitives. *Proceedings of SIGGRAPH 200*0, 335–342, 2000

[PFH00] Praun, E., Finkelstein, A., Hoppe, H. Lapped Textures. *Proceedings of ACM SIGGRAPH*, 465-470, 2000.

[PSS01] Praun, E., Sweldens, W., Schröder, P. Consistent mesh parameterizations. *Proceedings of SIGGRAPH 200*1, 179–184, 2001

[RHL02] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy. Real-time 3d model acquisition. *ACM Transactions on Graphic*s, 21(3):438–446, July 2002 (Proceedings of ACM SIGGRAPH 2002)

[RuL00] Rusinkiewicz, S., Levoy, M. Qsplat: A multi-resolution point rendering system for large meshes. *Proceedings of SIGGRAPH 200*0, 343–352, 2000

[Tau95] Taubin, G. A signal processing approach to fair surface design. *Proceedings of SIGGRAPH 9*5, 351–358, 1995

[Tur01] Turk, G. Texture synthesis on surfaces. *Proceedings of SIGGRAPH 200*1, 347–354, 2001

[WeL01] Wei, L.-Y., Levoy, M. Texture synthesis over arbitrary manifold surfaces. *Proceedings of SIGGRAPH 200*1, 355–360, 2001

[ZSS97] Zorin, D., Schröder, P. Sweldens, W. Interactive multiresolution mesh editing. *Proceedings of SIGGRAPH 9*7, 259–268, 1997

[ZP+02] M. Zwicker, M. Pauly, O. Knoll, and M. Gross. Pointshop 3d: An interactive system for point-based surface editing. *ACM Transactions on Graphic*s, 21(3):322–329, July 2000

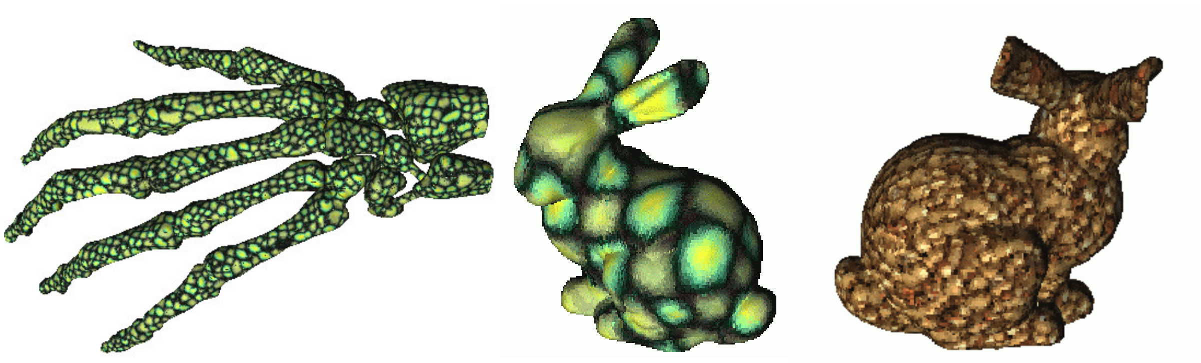


Figure 6. An isotropic texture synthesized on the points of the hand model, the same texture on the stanford bunny in larger scale and a texture of bird seed on the bunny.