Eurographics Symposium on Point-Based Graphics (2004)
M. Alexa, S. Rusinkiewicz, (Editors)

Stratified Point Sampling of 3D Models

Diego Nehab and Philip Shilane

Computer Science Department, Princeton University

Abstract

Point sampling is an important intermediate step for a variety of computer graphics applications, and specialized sampling strategies have been developed to satisfy the requirements of each problem. In this article, we present a technique to generate a stratified sampling of 3D models that is applicable across many domains. The algorithm voxelizes the model and selects one sample per voxel, restricted to the original model's surface. Parameters allow control of the uniformity of the sample placement and the minimum distance between samples. We demonstrate the effectiveness of this technique in selecting stroke locations for painterly rendering models and for producing sampled geometry used as input to shape descriptors.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling

1. Introduction

Many algorithms that take 3D models as input sample the model's geometry. The sampling strategy is critical for point rendering systems [GD98, LR98] which rely exclusively on sampled geometry to represent models. Some painterly rendering systems, such as Meier's animation scheme [Mei96], sample the object surface in order to distribute strokes over it. Turk's re-tiling of polygonal surfaces [Tur92] and Hoppe's mesh optimization [HDD*93] algorithms are examples of mesh simplification methods that perform sampling in an intermediate step. Many shape descriptors, such as spin-images [Joh97], D2 [OFCD01], shell histograms [AKKS99], and EGIs [Smi79], are usually implemented to take sampled models as input. More recently, Turk's sampling technique has been used to choose surface points where the irradiance is evaluated in order to approximate the diffusion equation needed for the efficient computation of sub-surface scattering effects [JB02].

In this article, we present a stratified sampling strategy for 3D models. Stratified sampling is a technique that generates evenly spaced samples by subdividing the sampling domain into non-overlapping parts and sampling independently from each part. It has been shown to decrease the variance of the numerical estimation of integrals in several applications, including antialiasing of ray-traced images [Mit96]. Our technique behaves like voxelization, but generates samples on the surface of the model. The idea is to voxelize the model and output one sample for each voxel, chosing a position from the part of the model surface that is contained in the voxel's bounding box. The sample is selected accoording to a probability that decays as its distance to the center of the voxel increases. We allow the user to control the sampling resolution, the regularity of the sampling, and the minimum distance between samples.

Most previous sampling strategies consider only sample density over the surface area of the original model. Among them, uniform sampling is by far the most common strategy. Samples are spread such that the probability of a surface point being sampled is equal for all surface points. Uniform sampling is popular because it is simple, efficient, and unbiased. However, artifacts such as those seen in random dithered images also appear in uniformly sampled models, and for many applications, these artifacts are aesthetically undesirable. Other applications, such as point rendering systems, demand bounded maximum or minimum distance between samples. For these applications, uniform sampling is not an option.

Turk describes a sampling strategy that produces samples evenly distributed over the surface area of a model [Tur92]. His technique starts from a uniform sampling of the mesh and places a charged particle at each sampled position. While constrained to remain on the object's surface, these

particles are allowed to repel each other until equilibrium is reached. The output of the sampling is then retriangulated to produce a simplified version of the original mesh.

Johnson created a method that goes in the opposite direction [Joh97]. His goal was to create a sampling of a model at a given resolution for computing spin-images. For that task, he used a mesh simplification strategy based on vertex splits and edge collapses using a priority metric that favors edges of a target length. When no operations can be performed that maintain the new mesh within an envelope of the original mesh, the process halts. The final mesh has small variance in edge lengths, and therefore the distance (geodesic) between adjacent vertices (the samples) is uniform.

In more recent work, Alliez describes a remeshing strategy that takes advantage of halftoning techniques [AMD02]. By first splitting the input mesh into disk like patches which can be parametrized, his technique is able to generate samplings with different properties. A variety of geometry maps (based on curvature, projected area etc.) can be computed for each patch. These geometry maps can be dithered and the results triangulated. Mapping the triangulation back into the original patches and stitching them together generates the new mesh.

Point rendering systems require that the minimum distance between samples be small enough to produce a rendering with no holes [GD98]. Furthermore, for efficiency, the number of samples should be as small as possible. Samplings with such properties can be produced with three orthogonal layered depth images [LR98, PZvG00].

In shape matching applications, the sampling strategy must capture the shape of the object being sampled. Uniform sampling is often used when properties are to be evaluated uniformly over the surface area [OFCD01]. Another common technique involves rasterizing the model surface to a voxelized grid and using the occupied voxels as samples. Results show that the expressive power of descriptors operating on voxelized models is generally higher than those that take uniformly sampled models as input [SKMF04].

Some techniques sample directly from implicit surface representations [WH94, ST92]. Naturally, implicit models can be triangulated before being sampled by methods that operate on triangle meshes.

In the following sections we describe the algorithm, analyze its running time complexity and the quality of the results, and show its application in painterly rendering and shape matching.

2. Algorithm description

Our algorithm is divided into three simple steps. The first step is the voxelization of the model. The next step produces one sample for each voxel. The final step constrains the minimum distance between samples by removing samples that are too close to each other. Figure 1 outlines the algorithm.

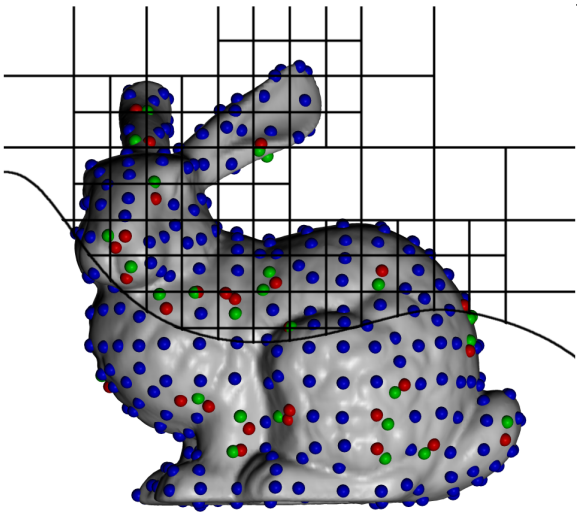

Figure 1: *Algorithm overview. The solid lines represent an octree voxelization of the model. Each dot represents a sample. Red and green samples are too close to each other and are considered for removal. Only the red are actually removed from the final sampling.*

The voxelization registers all triangles that touch the interior of each voxels's bounding box. One triangle is chosen from each voxel and from this triangle a new sample is produced as described later in this section. The resolution of the voxelization is specified by the user and controls the number of samples that the algorithm produces.

Rather than choosing the point on the surface of the object that is closest to the center of each voxel, we pick a position according to a probability distribution. Ideally, the distribution should favor points close to the center, but allow for a user controllable amount of variation in their placement. Naturally, many distributions satisfy these requirements. For the results shown in this paper, we use the exponential distribution function ($\lambda e^{-\lambda d}$) on the distance d (relative to the edge length of a voxel) between the sample and the center of the originating voxel.

We use a simple integration scheme to sample according to this distribution. Each triangle in a voxel is subdivided until the probability density function can be considered constant throughout its area. To ensure that integration happens only within voxel boundaries, subtriangles that cross boundaries can be subdivided further until no edge is bigger than a fixed length, relative to voxel edge length. The function value at the centroid of a terminal subtriangle is multiplied by its area and is defined as the subtriangle's priority. A roulette scheme is used to select a subtriangle according to these priorities. Once a terminal subtriangle is chosen, uniform sampling is used to produce a sample from it. Figure 2 gives an idea of the effect of λ.

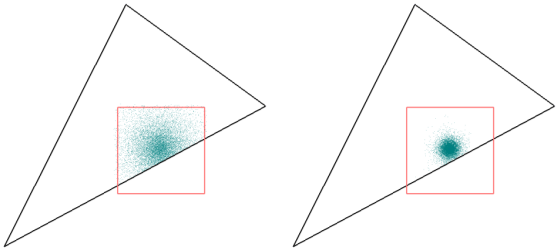

Figure 2: *The effect of the probability distribution function. The two figures show a comparison between the distributions of 10k samples in a large triangle around the center of a small voxel, for λ values of 5 and 20.*

model	triangles	height 5	height 6	height 7
bunny	69k	1.8/0.2 (497)	2.2/1.1 (2062)	2.7/5.2 (8401)
elephant	157k	3.6/0.36 (417)	4.3/1.2 (1731)	5.3/5.7 (7021)
dragon	871k	18.6/1.3 (549)	23.1/1.6 (2350)	27.7/4.8 (9806)

Table 1: *Running times. For tree heights 5, 6, and 7, times in seconds are given for the octree creation and sampling steps of the algorithm, respectively. The elimination of close samples runs in negligible time. The number of samples output is given in parenthesis. All experiments were run on a 1GHz PowerBook G4 laptop.*

The final step in the algorithm addresses the possibility that samples generated close to the boundary between two or more adjacent voxels can be too close to each other. For some applications, such as mesh simplification, this is undesirable. One solution is to enforce a minimum distance between samples, relative to the size of the voxel. For a given cluster of points that are too close to each other, we would like to keep those that better represent the cluster. We preserve original sample positions and simply eliminate samples one by one until the minimum distance constraint is satisfied.

Naturally, we are faced with the question of which samples to remove first. Given the minimum distance m between samples, we define the set $N(p) = \{q \,|\, \mathrm{d}(p,q) < m\}$ for each sample p. The candidates for removal are all points with a non-empty $N(p)$. We choose for removal the candidate r for which the centroid of $N(r)$ is furthest from r. After removing r, we update the neighborhoods of all points in $N(r)$ and repeat the process while candidates remain.

In the following two sections, we analyze the time complexity of each step in our algorithm. We present examples that illustrate the effect of the parameters m and λ in the samplings. Finally, we employ techniques developed by the halftoning community to analyze the quality of our samplings and compare it to the point distributions generated by other methods.

2.1. Time complexity

In our implementation, the voxelization step is performed with an octree. Assume a mesh with t triangles and area A, measured in multiples of the voxel face area. Define the *ceil area* $\lceil\!\lceil A \rceil\!\rceil$ as the sum of $\lceil A_i \rceil$ for all triangles i. Let h be the height of the octree. Then, the worst case voxelization step takes time $O(\lceil\!\lceil A \rceil\!\rceil + ht)$. An $O(\lceil\!\lceil A \rceil\!\rceil)$ algorithm exists that rasterizes all triangles directly into a regular voxel grid, but table 1 shows that for most applications the $O(\lceil\!\lceil A \rceil\!\rceil + ht)$ approach is fast enough.

The computation of sample positions requires the integration of the probability density function over all area of the model. In our implementation, we limit the area of the smallest generated subtriangle to be $1/r$ times the voxel size area (we use $r = 25$). Therefore, the running time for the integration is $O(Ar + t)$. This is confirmed in table 1, which shows that the time for the sampling step is multiplied approximately by 4 every time the octree is made deeper, causing the model area (relative to voxel face area) to quadruple.

Consider that the minimum distance constraint m at the final step is smaller than the side of a voxel (or almost no samples will be left). In addition, there is only one sample per voxel. Therefore, using an octree and a priority queue, the rejection of samples can be implemented in $O(n + c \log c)$, where n is the number of samples before rejection and c is the original number of candidates for removal. The low complexity makes this step run in negligible time when compared to the other steps.

2.2. Quality analysis

In the 2D case, we can compare our results with those produced by halftoning methods and employ techniques developed to analyze them. Halftoning methods are usually evaluated by their radially averaged power spectrum distributions (RAPSD) and their radial anisotropy. Intuitively, these measure the frequency content of the spacial distribution of points produced by a halftoning algorithm. The RAPSD measures the power per radial frequency, and the radian anisotropy gives the variance of that power per radial frequency (see [Uli88] for formal definitions and examples). These are computed for dithered constant gray level images, considering the minority pixel positions as a point process. We use our method to produce a sampling of a 1×1 square in order to compare it to halftoning methods over images. We match our voxelization resolution with the gray level of the input images to the halftoning methods, producing samplings with the same expected density. Figures 6 and 7 were produced from periodograms computed

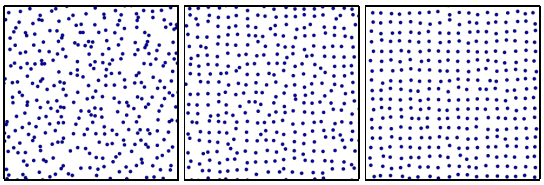

Figure 3: *Varying λ. From left to right, values 0, 10, and 30 produce samplings progressively more aligned with the voxelization grid.*

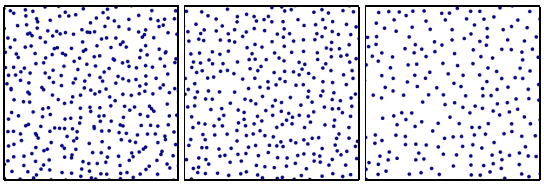

Figure 4: *The minimum distance constraint. From left to right, values of 0, 0.5, and 0.75 of m produce samplings with progressively more rejected samples. All examples use $\lambda = 2$.*

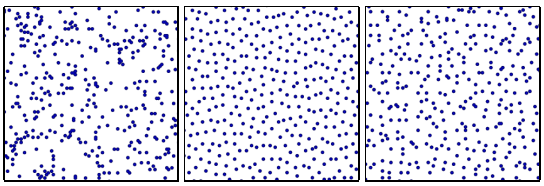

Figure 5: *Dithered constant $\frac{1}{32}$ gray level images (not to scale). From left to right, random dithering, blue noise dithering, and stratified sampling ($\lambda = 5$, $m = 0.5$).*

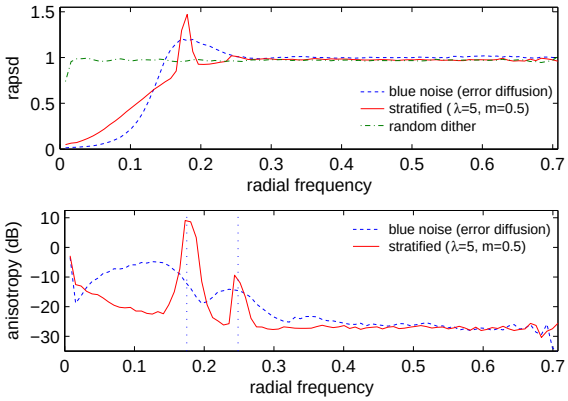

Figure 6: *RAPSD and radial anisotropy plots corresponding to figure 5. Although the radial power spectrum for the stratified sampling is similar to blue noise, the spectrum is anisotropic.*

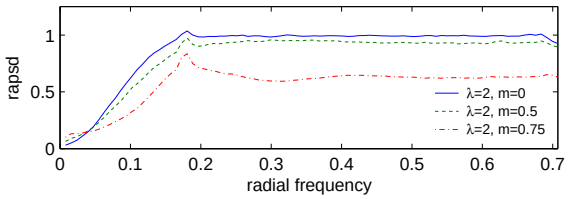

Figure 7: *RAPSD for samplings of figure 4. Increasing values of m produce patterns with better blue noise properties. The decrease in power is due to the increase in the number of rejected samples.*

by Welch's method [Wel67], and averaged over 20 sampled 512×512 images. We also use these plots to discuss the effects of m and λ.

The parameter λ provides control over how far away the samples are likely to move from the center of their originating voxels (restricted to the surface of the model). Small values of λ turn the algorithm into a jittered sampling. Large values turn it into regular voxelization. In between, there is a continuum of options, shown in figure 3.

Figure 5 shows $\frac{1}{32}$ gray level images produced with random dithering, Floyd-Steinberg [FS76] error diffusion with 50% random weights processed on a serpentine raster (this method displays blue noise properties [Uli88]), and with our method. Figure 6 shows the corresponding RAPSD and radial anisotropy plots. As expected, the random dithering (which is closely related to uniform sampling) displays a flat power spectrum. Both the blue noise dithering and our method have peaks at the target frequency, and most of the remaining power is moved to higher frequencies. This produces a visually pleasant pattern, since higher frequencies are naturally filtered by our visual system. Notice, however, that the sampling produced by our method has a second peak. This is an artifact of the alignment to the rectangle grid, which enforces frequencies corresponding to horizontal/vertical as well as diagonal alignments. These peaks are more evident in the radial anisotropy plot, which shows that our method performs worse than the blue noise dithering exactly in the two main modes.

Figure 4 shows the samples produced with different values of m, which controls the minimum distance between samples. Figure 7 shows that higher values of m produce RAPSDs with increasing blue noise properties. However, since the number of samples drop, the plots show a corresponding decrease in the total power of the distribution. The RAPSD for the stratified sampling of figure 6 does not show a significant power loss because the higher λ causes fewer samples to be rejected.

Figure 8 shows samplings of the model of a lion. The highly detailed mane takes 81% of the surface area of the 104k triangles in the model. Models of this class are a problem for most sampling strategies, and were one of the main

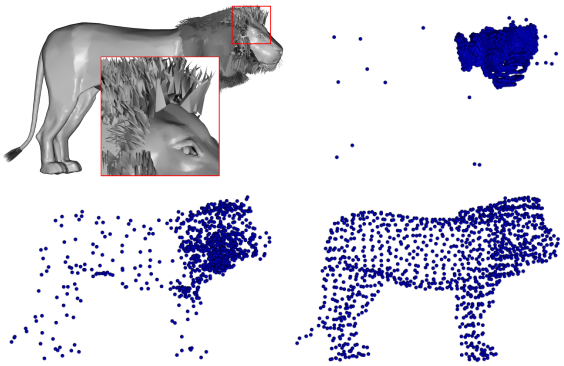

Figure 8: *The top right lion shows vertex positions of a the lion model after simplification by Garland's quadric error metric. The bottom left lion was sampled uniformly, while the bottom right lion was sampled with our algorithm (1k samples for both).*

motivations behind this work. The complex mane unduly influences methods guided by area or by vertices. As far as our method is concerned, the mane is just a region of space with a high triangle count.

When the input mesh is smooth (like the bunny mesh), simplification algorithms such as Garland's quadric error metric [GH97] generally produce high quality simplified models. Models like the lion, however, present a harder problem. We can create a simplified mesh from our sampling, using either the ball-pivoting algorithm [BMR*99] or the topology preserving method described by Turk [Tur92]. For smooth meshes, Garland's method outperforms our approach by an order of magnitude in Hausdorff distance to the original mesh, as computed by Metro [CRS98]. For models like the lion, however, the situation is reversed. Figure 9 shows the result of triangulating a stratified sampling of the lion, and the same model simplified by Garland's method, both with the same triangle count. The output of the ball-pivoted lion could be simplified even further by traditional methods.

3. Results and applications

As mentioned in the introduction, the stratified point sampling is useful across a variety of applications. For point-rendering systems, it provides a sampling with a constrained maximum distance between samples, allowing for a rendering with no holes. For painterly rendering, it provides evenly spread samples that can be used to stroke the model. For shape matching, it avoids oversampling high frequency details and captures the overall shape of the model. We present results for painterly rendering and shape matching.

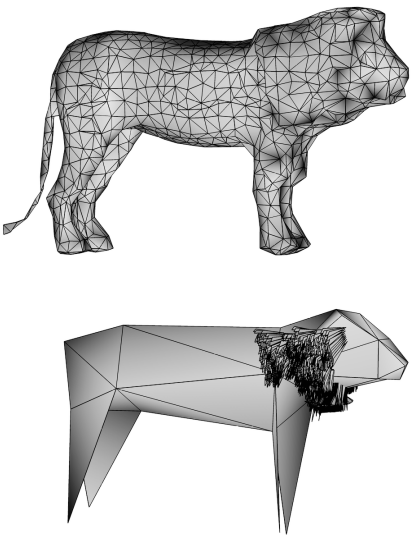

Figure 9: *Simplified lion. The top figure was simplified from 104k faces down to 2.6k starting from a stratified sampling with 1.4k samples. The bottom model was simplified using the quadric error metric to the same number of faces.*

3.1. Painterly rendering

A simple technique that produces interesting painterly rendered models is described by Meier [Mei96]. The main idea is to sample the geometry of a model and use the sampled positions to stroke textured images on it. Different choices of stroke images can produce renderings similar to impressionist paintings or animal fur, among others.

Taking advantage of the graphics hardware widely available in current personal computers, it is possible to implement a similar technique that runs in real-time. The mesh is initially rendered to the depth buffer with a small depth offset. For each visible sample point, a rectangle is drawn into the color buffer, with opacity determined by a stroke texture. The viewing direction and surface normal can be used to control different stroke properties, such as color, opacity and which of several stroke textures to use.

Strokes are alpha-blended on top of each other, creating the illusion of seamless integration between adjacent strokes. For this step, stroke size and sampling density need to agree so that strokes are evenly distributed over the object surface. Too many samples might produce overly saturated regions whereas too few samples may leave regions without strokes.

Although Meier obtained good results with uniform sampling, care must be taken or the inherent lack of local control on the sample distances will produce poor results. As an example, the two pictures in figure 10 were produced from samplings of the bunny model, painted with fur-like strokes. The strokes in the first image were placed by uni-

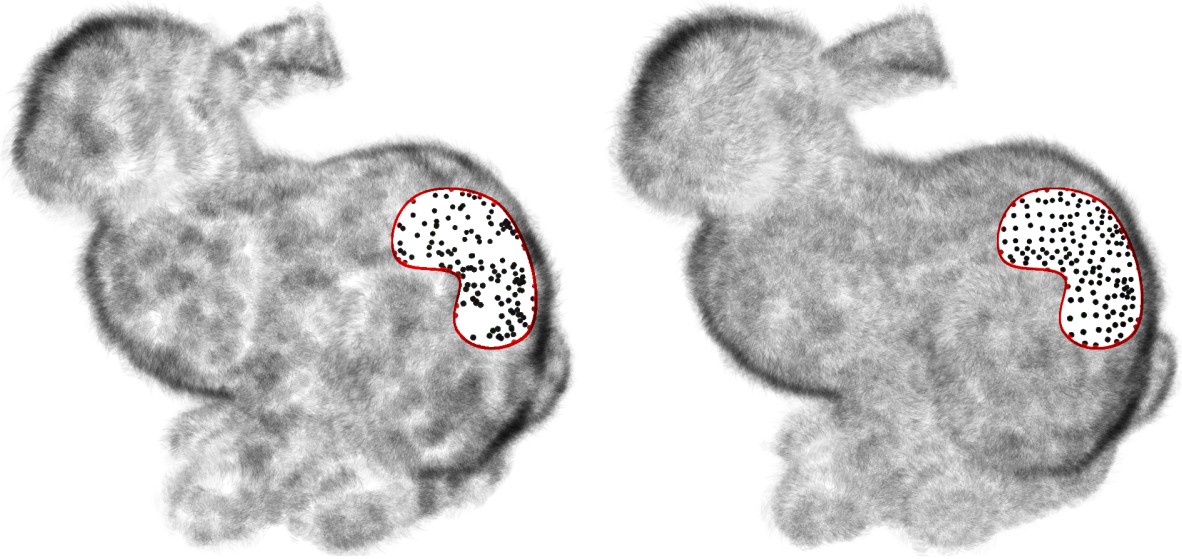

Figure 10: *Two furry bunnies at 7.5k strokes. The left figure uses uniform sampling, the right uses stratified sampling. Both figures use the same number of samples. The stroke positions are shown in the cut away region.*

form sampling. The second bunny was produced with the same number of strokes and the same parameters, but using the stratified stroke positions. Unless the goal of the artist was to depict a bunny with uneven fur, it is clear that the stratified sampling produces better results. The irregular spacing created by the uniform sampling causes some regions to receive too many strokes (too dark) and other regions to receive too few strokes (not painted). Stratified sampling greatly reduces these artifacts. For a model like the bunny, techniques that spread samples evenly over the model's surface area [Tur92, Joh97] would show similar improvements. For models like the lion in figure 8, our method or LDIs [LR98] would perform better.

3.2. Shape Matching

In the field of shape matching, there has been a great deal of recent research on shape descriptors as a basis for measuring the similarity of two models. Comparing models directly is an ill posed problem, so a common technique is to create a description of the shape and then compare descriptors directly. A shape descriptor is generated by analyzing properties of a model and creating a feature vector of property values, and the difference between two models is described as the difference between their respective feature vectors. Shape descriptors are favored over other shape matching techniques such as graph matching when computational time must be minimized for an interactive search application. Shape matching research is surveyed in [TV04].

Many shape descriptors sample properties of a model on selected points of its surface [Joh97, OFCD01, BMP00]. Choosing samples uniformly on the surface area has been shown to help make shape descriptors robust to small errors commonly found in models downloaded from the Internet [OFCD01]. As shown in figure 8, the 3D model of a lion has an enormous number of triangles representing the mane (81% of the surface area). Uniform sampling on the surface leads to a sparse sampling of the rest of the model. Arguably, the legs and tail are important features of the lion that should be better represented in the shape descriptor for matching against other quadrupeds. In order to show fine detail in the model, though, a large percentage of the surface area may be used on what is a small portion of the volume of the model.

We chose to investigate shape retrieval using stratified sampling versus uniform sampling on the 907 models of the test set of the Princeton Shape Benchmark (PSB) [SKMF04]. This set of models is partitioned into 131 human-generated classes representing a variety of common 3D graphics models. We evaluated the following eight shape descriptors, representing a variety of techniques in the literature, using both stratified and uniform samples: D2 Shape Distribution (D2) [OFCD01], Shape Histogram Shells [AKKS99], Shape Histogram Sectors [AKKS99], Shape Histogram Sectors and Shells (SecShells) [AKKS99], Spherical Extent Function (Ext) [SV01], Radialized Spherical Extent Function (RExt) [Vra03], Gaussian Euclidean Distance Transform (GEDT) [KFR03], Spherical Harmonic Descriptor (SHD) [KFR03], and Voxel. The last is a uniform, axis-aligned, $64 \times 64 \times 64$ voxel grid, representing the binary condition of whether surface area intersects each voxel.

Shape Descriptors	Uniform DCG	Increase (%)	Stratified DCG	Increase (%)	Voxel DCG
Shells	.378	7.7	.407	5.4	.386
Sectors	.521	3.8	.541	2.3	.529
SecShells	.538	4.3	.561	2.9	.545
D2	.445	2.7	.457	2.7	.445
Ext	.556	1.3	.563	0.2	.562
RExt	.583	2.7	.599	-0.3	.601
GEDT	.582	1.4	.590	1.0	.584
SHD	.588	1.2	.595	1.9	.584
Voxel	.538	2.0	.549	1.1	.543

Table 2: *Comparing 9 shape descriptors on the PSB base classification. For nearly all descriptors, stratified sampling outperforms uniform sampling and voxelization of the model.*

Models were normalized for translation and rotation and shape descriptor parameters were configured as described in [SKMF04]. This selected set of shape descriptors is not intended as a full evaluation of shape descriptor research but provides a breadth of techniques for comparison.

We performed a leave-one-out experiment, where every model in the database is used as a query model, and the resulting models are ordered by similarity of their descriptors. To measure the performance, we used the discounted cumulative gain (DCG) metric defined in [JK00], which is logarithmically weighted by matches appearing towards the front of the retrieval list. Scores range from $[0, 1]$, and values closer to one indicate better performance.

Table 2 demonstrates that across all shape descriptors, stratified sampling has better performance than uniform sampling. As indicated in figure 8 with the lion, uniform sampling can underrepresent important features of a model when the surface area is not evenly distributed over the model. Since many shape descriptors are based on sampling the surface, stratified sampling can offer improvement that seems to be generalizable across descriptors with a variety of properties.

Stratified sampling is clearly useful when a portion of the algorithm involves selecting data points, but most of these descriptors (except D2) can incorporate full polygons from the mesh as opposed to point samples. The polygons are rasterized and occupied voxels are determined. Table 2 shows a comparison of the same set of descriptors using both stratified samples and the voxelized models. D2 is defined based on point samples, and the uniform sampling results were repeated for completeness. Across nearly all descriptors, stratified sampling has better performance than using the voxelized models for shape matching. Both experiments indicate that using that stratified sampling is a valuable technique for shape matching.

4. Conclusion

We described a new technique for stratified sampling of 3D polygonal meshes. A continuum of samplings can be generated, from highly jittered to highly regular. The samplings can be made to satisfy maximum and minimum distance constraints between adjacent samples and display blue noise properties. The method runs sufficiently quickly to be used as a preprocessing step to a variety of algorithms. We have shown that stratified sampling is useful in several domains, presenting results in shape matching and painterly rendering.

For shape matching, stratified sampling outperforms uniform sampling, possibly because of uneven sampling over the model, and is competitive with previously published results that rasterize polygons to a voxel grid. For painterly rendering, we have shown that the aesthetic results are highly dependent on the sampling and that stratified sampling does a good job of evenly covering the surface a model.

An area of future research involves the use of non-cubic grids in an attempt to generate samplings with better radial isotropy. Another useful extention would be the computation of adaptive samplings that concentrate on features that the user might wish to preserve, such as high curvature regions. It would be also interesting to investigate multi-resolution applications of this technique.

5. Acknowledgements

Special thanks to Greg Prisament for creating the painterly renderings of the bunny. We would also like to thank Michael Garland for making QSlim available and Paolo Cignoni for making Metro available. Finaly, we would like to thank the anonymous reviewers for their constructive comments.

References

[AKKS99] ANKERST M., KASTENMLLER G., KRIEGEL H.-P., SEIDL T.: 3d shape histograms for similarity search and classification in spatial databases. In *Proceedings of the 6th International Symposium on Spatial Databases* (July 1999), Springer Verlag, pp. 207–226. 1, 6

[AMD02] ALLIEZ P., MEYER M., DESBRUN M.: Interactive geometry remeshing. In *Proceedings of SIGGRAPH'02* (2002), ACM Press, pp. 347–354. 2

[BMP00] BELONGIE S., MALIK J., PUZICHA J.: Shape context: A new descriptor for shape matching and object recognition. In *NIPS* (2000), pp. 831–837. 6

[BMR*99] BERNARDINI F., MITTLEMAN J., RUSHMEIER H., SILVA C., TAUBIN G.: The ball-pivoting algorithm for surface reconstruction. *IEEE Trans-*

actions on Visualization and Computer Graphics 5, 4 (Oct. 1999), 349–359. 5

[CRS98] CIGNONI P., ROCCHINI C., SCOPIGNO R.: Metro: Measuring error on simplified surfaces. In *Computer Graphics Forum* (1998), vol. 17(2), Blackwell Publishers, pp. 167–174. 5

[FS76] FLOYD R., STEINBERG L.: An adaptive algorithm for spatial gray-scale. *Proceedings Society Information Display 17*, 2 (1976), 75–78. 4

[GD98] GROSSMAN J. P., DALLY W. J.: Point sample rendering. In *Rendering Techniques'98, Proceedings of the 9th Eurographics Workshop on Rendering* (Aug. 1998), Drettakis G., Max N., (Eds.), Springer-Verlag, pp. 181–192. 1, 2

[GH97] GARLAND M., HECKBERT P. S.: Surface simplification using quadric error metrics. *Computer Graphics 31*, Annual Conference Series (1997), 209–216. 5

[HDD*93] HOPPE H., DEROSE T., DUCHAMP T., MCDONALD J., STUETZLE W.: Mesh optimization. In *Proceedings of SIGGRAPH'93* (1993), ACM Press, pp. 19–26. 1

[JB02] JENSEN H. W., BUHLER J.: A rapid hierarchical rendering technique for translucent materials. In *Proceedings of SIGGRAPH'02* (2002), ACM Press, pp. 576–581. 1

[JK00] JARVELIN K., KEKALAINEN J.: IR evaluation methods for retrieving highly relevant documents. In *23rd Annual International ACMSIGIR Conference on Research and Development in Information Retrieval* (2000). 7

[Joh97] JOHNSON A.: *Spin-Images: A Representation for 3-D Surface Matching*. PhD thesis, Robotics Institute, Carnegie Mellon University, Aug. 1997. 1, 2, 6

[KFR03] KAZHDAN M., FUNKHOUSER T., RUSINKIEWICZ S.: Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Symposium on Geometry Processing* (June 2003). 6

[LR98] LISCHINSKI D., RAPPOPORT A.: Image-based rendering for non-diffuse synthetic scenes. In *Rendering Techniques'98, Proceedings of the 9th Eurographics Workshop on Rendering* (Aug. 1998), Drettakis G., Max N., (Eds.), Springer-Verlag, pp. 301–314. 1, 2, 6

[Mei96] MEIER B. J.: Painterly rendering for animation. In *Proceedings of SIGGRAPH'96* (1996), ACM Press, pp. 477–484. 1, 5

[Mit96] MITCHELL D. P.: Consequences of stratified sampling in graphics. In *Proceedings of SIGGRAPH'96* (1996), ACM Press, pp. 277–280. 1

[OFCD01] OSADA R., FUNKHOUSER T., CHAZELLE B., DOBKIN D.: Matching 3d models with shape distributions. In *SMI 2001 International Conference on Shape Modeling and Applications* (May 2001), IEEE, pp. 154–166. 1, 2, 6

[PZvG00] PFISTER H., ZWICKER M., VAN BAAR J., GROSS M.: Surfels: Surface elements as rendering primitives. In *Proceedings of SIGGRAPH'00* (2000), ACM Press/Addison-Wesley, pp. 335–342. 2

[SKMF04] SHILANE P., KAZHDAN M., MIN P., FUNKHOUSER T.: The Princeton Shape Benchmark. In *SMI 2004 International Conference on Shape Modeling and Applications* (June 2004). 2, 6, 7

[Smi79] SMITH D.: *Using Enhanced Spherical Images for Object Representation*. Memo, MIT, 1979. 1

[ST92] SZELISKI R., TONNESEN D.: Surface modeling with oriented particle systems. In *Proceedings of SIGGRAPH'92* (1992), ACM Press, pp. 185–194. 2

[SV01] SAUPE D., VRANIC D. V.: 3d model retrieval with spherical harmonics and moments. In *DAGM 2001* (Sept. 2001), pp. 392–397. 6

[Tur92] TURK G.: Re-tiling polygonal surfaces. In *Proceedings of SIGGRAPH'92* (1992), ACM Press, pp. 55–64. 1, 5, 6

[TV04] TANGELDER J. W., VELTKAMP R. C.: A survey of content based 3d shape retrieval methods. In *SMI 2004 International Conference on Shape Modeling and Applications* (June 2004). 6

[Uli88] ULICHNEY R. A.: Dithering with blue noise. In *Proceedings of the IEEE* (Jan. 1988), vol. 76(1), pp. 56–79. 3, 4

[Vra03] VRANIC D. V.: An improvement of rotation invariant 3d shape descriptor based on functions on concentric spheres. In *IEEE International Conference on Image Processing 2003* (Sept. 2003), vol. 3, pp. 757–760. 6

[Wel67] WELCH P.: The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Trans. Audio Electroacoustic 15*, 2 (June 1967), 70–73. 4

[WH94] WITKIN A. P., HECKBERT P. S.: Using particles to sample and control implicit surfaces. In *Proceedings of SIGGRAPH'94* (1994), ACM Press, pp. 269–277. 2