SCIENCE CHINA
Information Sciences

• **RESEARCH PAPER** •

September 2013, Vol. 56 092117:1–092117:12
doi: 10.1007/s11432-011-4322-8

Approximate Poisson disk sampling on mesh

GENG Bo[1], ZHANG HuiJuan[1], WANG Heng[1] & WANG GuoPing[1,2]*

[1]*Graphics and Interactive Technology Lab of Dept. of Computer Science, Peking University, Beijing* 100871*, China,*
[2]*The Key Lab of Machine perception and intelligent, MOE, Beijing* 100871*, China*

Received July 10, 2012; accepted August 15, 2012; published online September 30, 2012

Abstract Poisson disk sampling has been widely used in many applications such as remeshing, procedural texturing, object distribution, illumination, etc. While 2D Poisson disk sampling is intensively studied in recent years, direct Poisson disk sampling on 2-manifold surface is rarely covered. In this paper, we present a novel framework which generates approximate Poisson disk distribution directly on mesh, a discrete representation of 2-manifold surfaces. Our framework is easy to implement and provides extra flexibility to specified sampling issues like feature-preserving sampling and adaptive sampling. We integrate the tensor voting method into feature detection and adaptive sample radius calculation. Remeshing as a special downstream application is also addressed. According to our experiment results, our framework is efficient, robust, and widely applicable.

Keywords Poisson disk sampling, feature preserving sampling, adaptive sampling, remeshing, tensor voting

Citation Geng B, Zhang H J, Wang H, et al. Approximate Poisson disk sampling on mesh. Sci China Inf Sci, 2013, 56: 092117(12), doi: 10.1007/s11432-011-4322-8

1 Introduction

Recently, Poisson disk distributions are used frequently in computer graphics. One of the most important applications is sampling. Since Poisson disk sampling was first introduced by Dippé and Wold into computer graphics to solve problems of image anti-aliasing [1], it is now accepted as one of the best sampling patterns for a wide range of applications due to its blue noise properties, such as image anti-aliasing [1–3], primitive distribution for illumination [2,4–6], object distribution [7,8], etc.

While there are intensive studies of 2D plane Poisson disk sampling in recent years [9], direct Poisson disk sampling on 2-manifold surface is rarely covered [10]. Mesh as a discrete surface representation is widely used in many fields such as CAD, FEA, computer graphics, etc. Therefore, it becomes significant to perform Poisson disk sampling on mesh.

Efficiency and robustness are the major concerns of direct sampling algorithm on mesh. This paper presents a robust and efficient algorithm to generate Poisson disk distribution on mesh directly. It avoids distortions, such introduced by the parameterization because it manipulates manifold surface directly. We leverage the tensor voting method in the specified Poisson disk samplings, including feature preserving sampling and adaptive sampling(see Section 4).

*Corresponding author (email: wgp@pku.edu.cn)

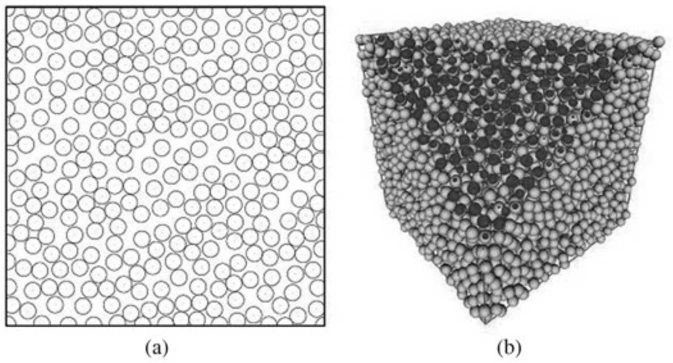

Figure 1 Possion disk distribution.(a) 2-D;(b) 3-D(Gamito [1]).

2 Related work

2.1 Basic concepts of Poisson disk sampling

Poisson-disk sampling(Figure 1) is a process of distributing uniform random samples on a domain embedded in an n-dimensional space based on a minimum distance criterium between samples. We propose an efficient Poisson-disk sampling method for generating samples on the domain $D = [0, 1]^n$, involving a unit hypercube in n-dimensional space. The outcome of the method is a sample set $X = \{x_i | x_i \in D, i = 1, 2, 3, ..., N\}$ for which the sampling conditions can be expressed as follows [10]:

$$\forall x_i \in X, \forall S \subseteq D : P(x_i \in S) = \int_S \mathrm{d}x, \tag{1}$$

$$\forall x_i, x_j \in X : \|x_i - x_j\| \geqslant 2r, \tag{2}$$

where the parameter r is called the distribution radius. $P(x_i \in S)$ is the probability of x_i falling inside a subset S of D. Condition (1) ensures uniform distribution, and Condition (2) helps to combat clustering by preventing samples from getting closer than some given value $2r$. The process of generating this Poisson disk distribution is called Poisson disk sampling. Poisson disk distribution meets blue noise properties. More details of its spectrum properties will be given in Subsection 5.1.

2.2 Poisson disk sampling in 2 dimensions and high dimensional European space

Poisson-disk sampling for image anti-aliasing was introduced to computer graphics by Dippé and Wold [1] in 1985. From then on, Poisson disk distribution has attracted great attention. The researches can roughly be divided into three categories:

The traditional approach is the Dart Throwing [1], which is one of the simplest techniques to generate Poisson-disk distribution. It iteratively refines an existing point set by generating random point locations in the sampling domain. A point is discarded if there already exists another point within a disk with certain radius r of it(Condition (2)). One drawback of this approach is that it cannot guarantee algorithm convergence since the available sampling area shrinks gradually. Meanwhile, it hardly leads to maximal Poisson-disk distribution. To alleviate this problem and make the approach more efficient, Jones [11] used a Voronoi diagram to keep track of empty spots in the sampling domain. Dunbar and Humphreys [12] described a modified Poisson disk algorithm to generate isotropic point distribution with good spectral quality in $O(n)$ time. White et al. [13] employed a quadtree structure to exclude covered areas from dart throwing. Wei [14] performed parallel dart throwing on the GPU, using a multi-resolution grid structure. Gamito [10] leveraged a tree structure to achieve Poisson disk distribution in high dimension space.

The second category is relaxation-based method or energy minimization technique, which initially places a point set on the surface and then improves the placement through point relocation to achieve Poisson disk distribution. The classic Lloyd iteration method presented by Lloyd [15] in 1982 falls into this category. However, excessive iteration will severely influence the blue noise properties of points.

Balzer [16] made some improvements by fixing area in iterative process but with less efficiency. Liu [17] proved a C^2 continuity of the energy function on a convex regional, and applied quasi-Newton method to accelerate Lloyd method.

For large-scale Poisson disk distribution, many tile-based methods are advanced to improve sampling performance. But the tiling structure destroys blue noise properties of sampling results. Ostromoukhov et al. [18,19] successively introduced Penrose and Polyomino structure to generate Poisson disk distribution. Wang Tiling and its variants were widely used too [20–22]. In the preprocessing stage, dart throwing or relaxation-based method is commonly used to generate Poisson disk distribution on initial templates.

2.3 Poisson disk sampling on mesh

While there are intensive studies of 2D Poisson disk sampling in recent years, direct Poisson disk sampling on 2-manifold mesh surface is rarely covered.

Fu [23] implemented the Dunbar's [12] sampling algorithm on 2-manifold surface and achieved the Poisson disk sampling on mesh. In order to fit 2-manifold mesh, it replaced the sampling boundary in Euclidean space with Iso-geodesic boundary. Since all points lay on the original mesh surface, high quality remeshing has turned out to be another contribution. But the process to calculate geodesic distance isoline is challenging, making the algorithm with less efficiency.

Cline [24] applied dart throwing algorithm on 2-manifold surface. It established index structure according to the area to satisfy Condition (1). Meanwhile, Cline extended the algorithm to the sampling of other surface types, including spheres, NUBRS, subdivision surface and implicit surfaces. He implemented his method to work with geodesic distance rather than Euclidean distance. But it couldn't avoid inherent problems of the dart throwing algorithm such as inefficiency and difficulty in getting maximal Poisson-disk distribution.

Li et al. [25] modified tile-based method on 2-manifold surface. It calculated the dual graph of the parametric surface and then tiled the templates on the dual graph. It reduced the complexity of the required templates. One fatal problem of the method is the distortions introduced in the process of parameterization.

3 Algorithm framework

The input of our sampling scheme is a 2-manifold mesh surface. In this section, the uniform sampling process will be discussed in detail.

3.1 Algorithm process

Inspired by Cline [24], we adopt similar index structure in this paper. In addition, a mesh clipping method is used to ensure the efficiency of the index structure, and collision detection frame based on Isophotic distance metric is introduced in order to satisfy the distance condition. Figure 2 shows basic steps of the algorithm:

a) In the pre-processing stage, subdivide the large triangles.

b) Establish area based triangle index structure. See Subsection 3.2 for more details.

c) Pick an active triangle $T = \{p_a, p_b, p_c\}$ from index structure. See Subsection 3.2 for more details.

d) Get a random sampling point p in T by bilinear interpolation using two stochastic parameters $\alpha, \beta \in [0, 1)$:

$$p = p_a * \alpha + (p_b * \beta + p_c * (1 - \beta)) * (1 - \alpha). \tag{3}$$

e) If there is no conflict after collision detection, add the point p to sampling point set and go to next step; otherwise, go back to c).

f) Cut the mesh locally using a ball with center p and radius $2r$. Remove the part inside the ball and add the new triangles into index structure. Update the index structure.

g) Return if the index structure is empty; otherwise, go back to c).

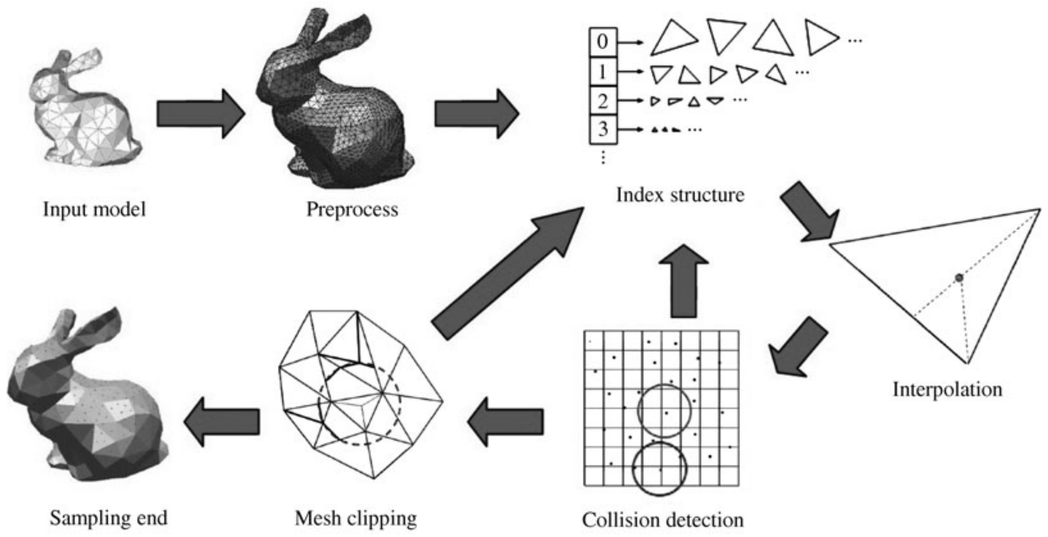

Figure 2 The algorithm process.

3.2 Index structure

This paper adopts Cline's [24] index structure that organizes triangles according to their areas.

- Index structure includes series of lists $\{\text{list}_0, \text{list}_1, \text{list}_2, \ldots, \text{list}_n\}$.
- Each list_i has a pair of area bounds: $B_{\max_i}$ and $B_{\min_i}$, where $B_{\max_i} = 2 * B_{\min_i}$ and $B_{\max_i} = B_{\min_{i-1}}$; B_{total_i} records the sum of triangle areas in list_i.
- Each triangle in list_i satisfies (Area(T) is area of T):

$$B_{\min_i} < \text{Area}(T) \leqslant B_{\max_i}. \tag{4}$$

To pick one triangle, we randomly select $\alpha \in [0, 1)$ and then find the first index i that meets Condition (5) by starting from index 0:

$$\alpha \leqslant \sum_{j=0}^{i} B_{\text{total}_j} \Bigg/ \sum_{j=0}^{n} B_{\text{total}_j}. \tag{5}$$

After finding the index i, pick a triangle T_j in sequence within list_i and then accept it with probability $A_{ij}/B_{\max_i}$, where the A_{ij} is the area of triangle T_j and $B_{\max_i}$ is the maximum area of triangles assigned to the list list_i. If T_j is not accepted, we continue to test the next triangle in the list list_i. If no triangle is accepted till the end, we choose the last triangle. Triangles in the list_i will invariably be at least half as large as $B_{\max_i}$, and the acceptance rate of each picked triangle is at least fifty percent, and the expectation of this triangle picking process will terminate in constant steps. Large triangle will be picked firstly, and so the sampling result satisfies Condition (1) approximately.

3.3 Mesh clipping method based on BFS

After generating a sampling point p within the triangle T, we clip the mesh locally using a ball with center p and radius $2r$ based on BFS (breadth-first search algorithms). The part inside the ball is defined as the sampling point's territory on which no point will be sampled in the subsequent steps.

Figure 3 shows the mesh clipping procedure. We check each triangle around the sampling points according to the BFS and find all triangles on the boundary. In order to avoid the errors caused by floating-point calculations, we first collect all edges intersected with the ball (Figure 3(a), dotted line) and then calculate all intersections on those edges. By inserting those intersections into mesh, triangles are degraded into polygons which are then cleared up by subsequent subdivision processing. The subdivision result is shown in Figure 3(c).

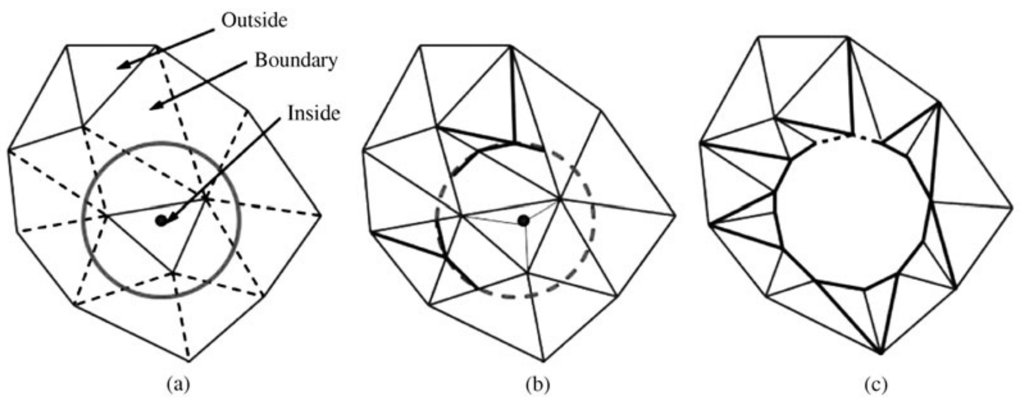

Figure 3 Mesh clipping process. (a) Collection of intersection edge; (b) calculation of intersect; (c) clipping.

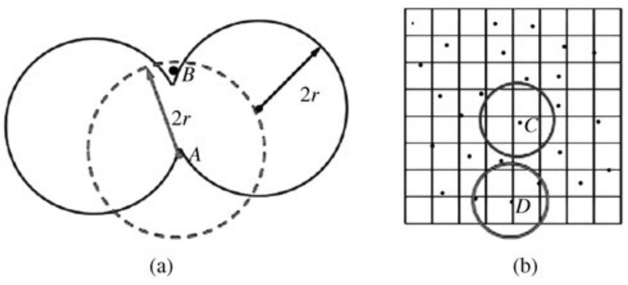

Figure 4 (a) Collision detection necessity; (b) collision detection principle.

Straight line in Figure 3(c) is used to replace the circular arc in this stage which produces error equal to the arc height. In order to control the error, we insert a new point(Figure 3(c), dotted line) in the arc when the error is larger than the threshold defined by the user.

3.4 Collision detection frame based on isophotic distance metric

An unexpected case may be found using the BFS mesh clipping algorithm, as shown in Figure 4(a), where the solid arc stands for cut part by existing sampling point set. Point A and dashed circle represent the current sampling point and cutting boundary, respectively. Point B located in point A's territory will not be cut because we use BFS to find neighbor triangles, and sampling in point B will destroy Condition (2).

To solve this problem, collision detection is needed between the new sampling point and the existing sampling point set. If the distance between the new sampling point and any point in existing sampling point set is less than $2r$ (like the point D in Figure 4(b)), the new sampling point will not be accepted and the sampling process continues. If not, the new sampling point is added into existing sampling point set (like the point C in Figure 4(b)). We use uniform grid to accelerate this collision detection process (Figure 4(b)).

Models always have some thin sheets (see Figure 5). Because the distance between the points of A and B under the collision detect radius, these two points cannot be accepted simultaneously while otherwise it should be. In order to overcome this kind of problem, we introduce Isophotic distance metric which substitutes for Euclidean distance in collision detection frame inspired by reference [26]. This metric distance takes the normal distance into consideration, so it is performs more sensitively for high curvature area on surface.

In this paper, we defined the weighted Isophotic distance metric as

$$\text{dist\_isophotic}(p_1, p_2) = \text{dist\_euclid}(p_1, p_2) \times \left(1 + \left(\frac{1 - n_1 \cdot n_2}{2}\right)^b\right), \tag{6}$$

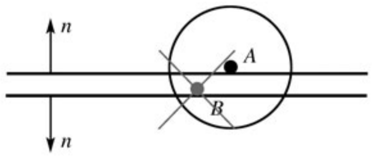

Figure 5 Thin sheet.

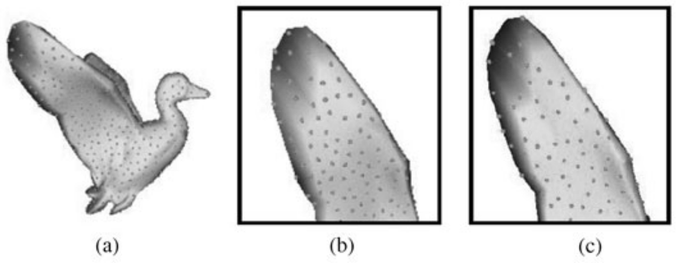

Figure 6 Comparison between Isophotic distance metric and Euclidean distance (sampling radius r=0.3). (a) Duck sampling (our method); (b) Duck wing sampling(our method); (c) Duck wing sampling (Euclidean distance).

where dist_euclid (p_1, p_2) is the Euclidean distance between point p_1 and p_2. n_1, n_2 are normals of the points p_1, p_2 respectively, b is the parameter for adjusting the influence of the normal. In experiment, we find that b=6 adjusts the influence well. Due to the parameter b in Eq. (6), the error of the Isophotic distance in planner part is almost the same as the Euclidean distance (when b=6, the angle of n_1n_2 is 90° such that $\text{dist\_isophotic}(p_1,p_2)=1.016*\ \text{dist\_euclid}(p_1,p_2)\approx\text{dist\_euclid}(p_1,p_2)$). This weighted Isophotic distance metric is easy to calculate, and meanwhile it works in our experiment well (see Figure 6).

4 Feature preserving sampling and adaptive sampling

Using sampling point set to represent the model, feature preserving sampling and adaptive sampling always provide good expressions. That explains why these two sampling methods become popular in CG field. In this section we will discuss these two issues in detail.

4.1 Feature preserving sampling

In feature preserving sampling, some sampling points must be placed in regions with sharp features. To this end, before sampling, we should extract a set of feature edges and corners from the input mesh first. Since the input mesh is a discrete approximation of the underlying smooth surface, feature extraction may be sensitive to noise. We improve the feature detection method based on the tensor voting [27] which can perform efficient and reliable feature detection. The algorithm goes as follows:

First of all, we assign the voting tensor T_v of a vertex v in an arbitrary point data set as a weighted covariance matrix which is called a vote. Then we define the normal voting tensor of a vertex on a triangular mesh as the weighted sum of neighbor triangles' votes, which is a 3 × 3 tensor matrix with three eigen-values λ_1, λ_2, λ_3 (assume $\lambda_1 \geqslant \lambda_2 \geqslant \lambda_3$). All vertices of the input mesh are classified according to the observation that there is a close correspondence between eigenvalue's (λ_1,λ_2, λ_3) distribution and geometrical features. Figure 7 demonstrates this correspondence.

Kim [27] adopted K-means iterative method to achieve automatic feature detection. In practical applications, we have found that the result of K-means depends on a good initial seed placement. So, using a threshold specified by user we classify the feature points to improve the previous method's performance.

After the classification, a region growing is performed for each seed triangle, and the boundaries of the regions are extracted as the edges. The experimental results show that the proposed algorithm is effective in almost all cases. Figure 8 is the feature detection result of Fandisk.

To imply feature preserving sampling, we make sampling on corners, edges, faces in turn to make sure that sampling points lie on sharp features. Figure 9 shows the feature preserving sampling result.

4.2 Adaptive sampling

Adaptive sampling is a variant of uniform sampling that adjusts sampling radius automatically according to the differential properties of surface. The sampling radius decreases in high curvature surface, and increases in low curvature surface.

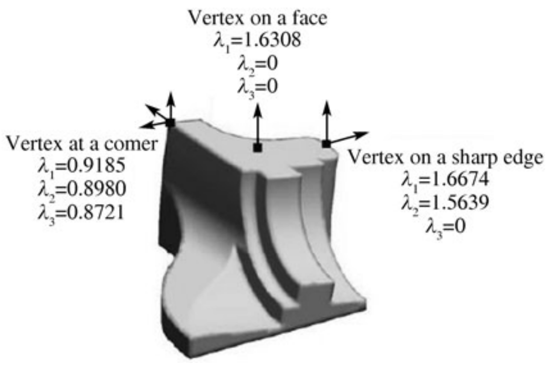

Figure 7 Eigenvalues of the normal voting tensor for different features [27].

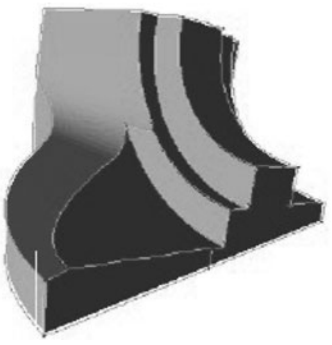

Figure 8 Features.

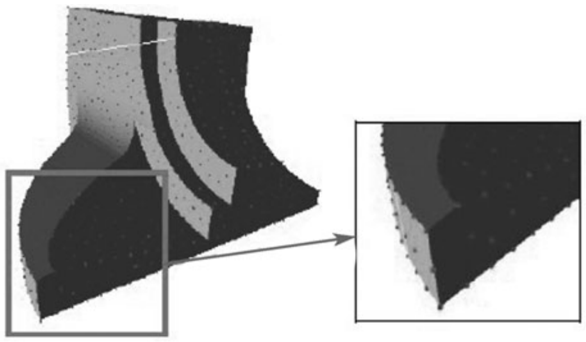

Figure 9 Feature preserving sampling.

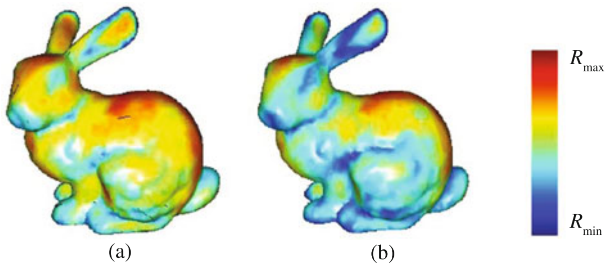

Figure 10 Comparison between adaptive sampling radius calculation methods. (a) Curvature method; (b) our method.

For adaptive Poisson disk sampling, we use the eigenvalues in subsection 4.1 to calculate adaptively sampling radius as follows:

$$R_i = R_{\min} + (\text{Value}_i)^{\text{Contrast\_rate}} \times (R_{\max} - R_{\min}), \tag{7}$$

$$\text{Value}_i = \frac{1}{(\alpha * (\lambda_1 - \lambda_2) + \beta * (\lambda_2 - \lambda_3) + (1 - \alpha - \beta) * \lambda_3) * C + 1} \qquad (\lambda_1 \geqslant \lambda_2 \geqslant \lambda_3), \tag{8}$$

where $R_{\max}$ and $R_{\min}$ are the maximum radius and minimum radius assigned by user, respectively. Contrast_rate is the contrast ratio indicating the transitional smoothness from the maximum radius to minimum radius (in this paper it is set to 1). C is a parameter; α, β are weights (In our experiment, C=100, $\alpha = 0.001$, $\beta = 0.5$). One significant advantage of adaptive sampling is that it shares the framework (tensor voting method) with feature preserving sampling. So, we integrate feature preserving sampling and adaptive sampling. The performance is improved largely because many intermediate results can be reused. On the other hand, results of this method represent the surface's differential properties as well as the method based on curvature [23] (see Figure 10).

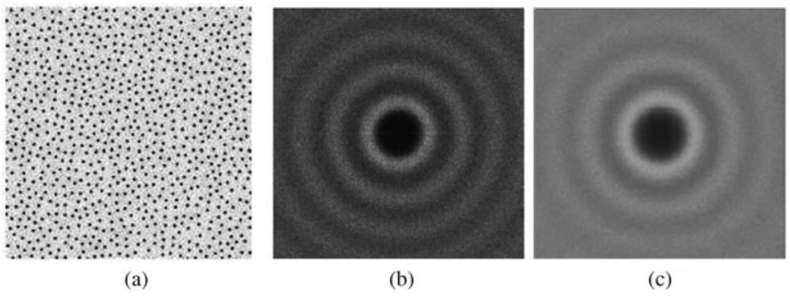

Figure 11 Spectral analysis of sampling results. (a) Our sampling result; (b) our method average spectral analysis; (c) average spectral analysis of [17].

Through the above steps, we get the vertex radius on the mesh. It is easy to calculate the arbitrary sampling point radius by bilinearly interpolating the three vertices radius of the corresponding triangle:

$$R_p = \sum_{V_i \in \mathrm{Tri}(p)} \omega_i R_i. \tag{9}$$

We use the same sampling process as introduced in the section 3 by adopting the new sampling radius in clipping mesh step. Fu [23] has proved that adaptive sampling boundary is a combination of a series of conic segments. For simplicity of calculation, we use $R_p + R_{\min}$ as the cutting radius of sample point p.

We also modify the collision detection (see subsection 3.4 for details). Through recording each sampling point radius, the collision detection ensures that the distance between any two points is larger than the sum of their radii in sampling point set.

5 Results and analyses

We implemented our algorithm on a PC (Intel Core Duo CPU 2.67 GHz, 2 GB RAM). Here we give a comparisons between Fu's [23] method and ours.

5.1 Blue noise properties analysis

As we know, the analysis of blue noise properties of Poisson disk distribution is limited to 2D plane sampling, and the analysis in high dimensional space and manifold surface is rare. Therefore, we adopt the analysis of blue noise properties of planer mesh. Our analysis method is based on [28,29]. The generated average power spectrum is depicted in Figure 11(b).

Compared to the average power spectrum generated by Lagae [9], the average power spectrum of our method also has a peak in the center of DC component. The energy in peripheral wide low-frequency region is extremely low, while energy is mainly concentrated in the high frequency part. This shows that our sampling results have Poisson disk distribution's blue noise properties.

5.2 Sampling time

This paper greatly improves the algorithm speed due to simpler arithmetic logic and less time consumption in computation of geodesic distance. For most models it takes a very short time to make sampling.

The sampling procedure were performed on 10 typical models and the results were compared with Fu's [23] method. Table 1 lists the comparison data. The first row shows our algorithm results while the second row shows the contrast algorithm results for each chart.

The last column displays the ratio of the sampling rate. It shows that our algorithm is at least 10 times as fast as contrast algorithm. The specific speed varies with respect to specific tesselation and topology.

Table 1 Comparison of sampling times.

Modle	algorithm	face number	sampling time (ms)	sampling number	sampling rate (points/ms)	sampling rate ratio (Our/FuYan [23])
Fandisk	Our algorithm	12,946	7,172	1,469	0.2048	19.37
	FuYan [23] algorithm		36,125	382	0.0106	
Casting	Our algorithm	10,204	5,031	1,834	0.3645	10.04
	FuYan [23] algorithm		31,843	1,156	0.0363	
Block	Our algorithm	4,208	2,500	2,026	0.8104	12.46
	FuYan [23] algorithm		8,469	551	0.0651	
Sharp sphere	Our algorithm	18,864	11,656	3,559	0.3053	195.78
	FuYan [23] algorithm		229,547	358	0.0016	
Rock arm	Our algorithm	18,794	5,015	1,488	0.2967	32.68
	FuYan [23] algorithm		71,157	646	0.0091	
Dinosaur	Our algorithm	19,458	6,515	727	0.1116	31.35
	FuYan [23] algorithm		111,532	397	0.0036	
Duck	Our algorithm	4,990	5,156	1,714	0.3324	15.58
	FuYan [23] algorithm		14,532	310	0.0213	
Dolphin	Our algorithm	5,592	2,156	376	0.1744	102.04
	FuYan [23] algorithm		77,234	132	0.0017	
Man head	Our algorithm	53,966	56,781	1,601	0.0282	56.77
	FuYan [23] algorithm		940,235	467	0.0005	
Hand	Our algorithm	17,290	7,594	2,162	0.2847	106.47
	FuYan [23] algorithm		89,750	240	0.0027	

5.3 Sampling results

Figures 12(a)–(j) show the sampling results of the 10 models mentioned above, where (a)–(g) are feature preserving sampling examples. It can be seen from these figures that the points are sampled accurately on the corners and edges (Duck wing edges and Fandisk edges). Figure 12(e)–(j) show adaptive sampling examples. The differences between flat part and high curvature part are obvious.

Our algorithm does collision detection based on Euclidean distance. Because the geodesic distance between two points is equal or greater than its Euclidean distance, the sampling points must satisfy the collision detection based on geodesic distance. Also it should be noted that we introduce isophotic distance metric to solve the thin sheet problems which also have errors. However, by Eq. (6) in subsection 3.4, this error relative to the Euclidean distance is trivial. So, we naturally reach the conclusion that our algorithm satisfies Condition (2) approximately. In this paper, the process of taking triangle from index structure does not fully obey area Condition (1); meanwhile, accidental error is inevitable when we apply Euclidean distance to clipping triangles instead of geodesic distance. So, this method can only satisfy Condition (1) approximately. To sum up, the algorithm is an approximate Poisson disk sampling algorithm.

5.4 Remesh application

Remeshing is one application of our algorithm. We implemented Fu's [23] post-processing method to make remeshing. Figure 13 shows the excellent result of remeshing through a typical example. The remesh results are only affected by the differential properties of the initial surface, except for the initial mesh tesselation. It is an improvement compared with traditional remesh method implemented by adjusting the vertex positions [30].

5.5 Limitations

Although this algorithm gets fairly good results in sampling speed and quality, it still has some limitations. First of all, this method needs several user-defined parameters, thus increasing the difficulty in user

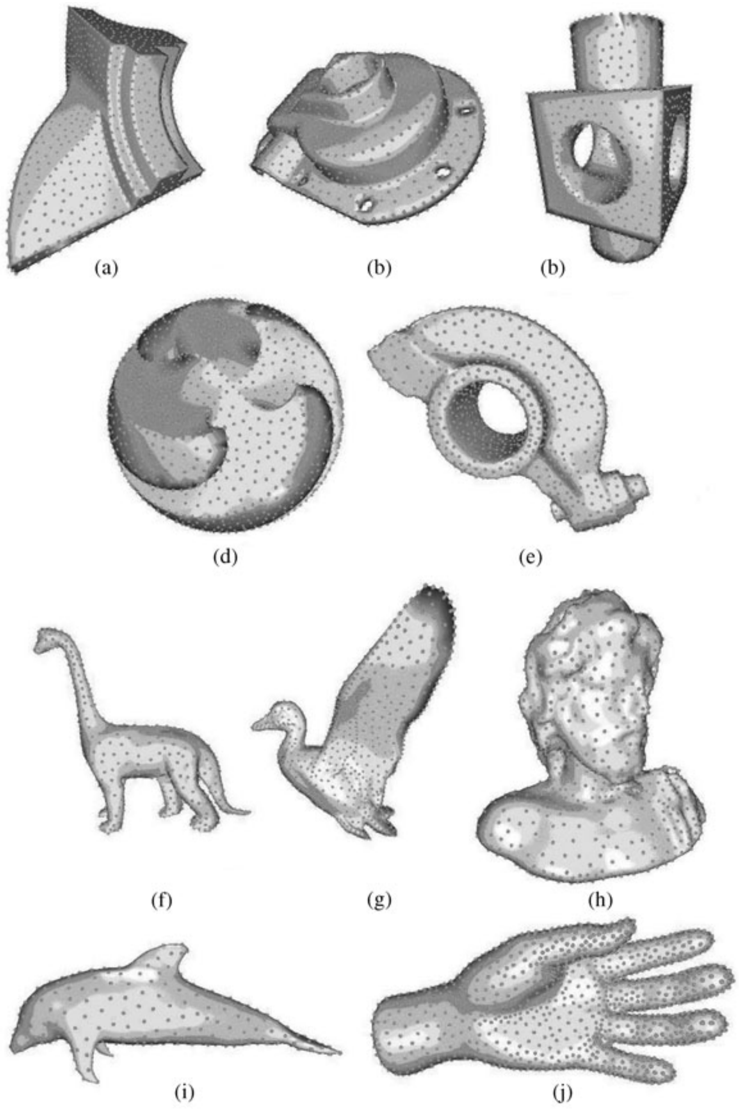

Figure 12 Sampling result. (a) Fandisk (r= 0.3); (b) Casting (r= 0.3); (c) Block (r = 0.3); (d) Sharp sphere (r= 0.3); (e) Rock arm($R_{\max} = 0.4, R_{\min} = 0.2$); (f) Dinosaur ($R_{\max} = 0.3, R_{\min} = 0.15$); (g) Duck($R_{\max} = 0.4, R_{\min} = 0.2$); (h) Man head ($R_{\max} = 0.4, R_{\min} = 0.2$); (i) Dolphin($R_{\max} = 0.4, R_{\min} = 0.2$); (j) Hand ($R_{\max} = 0.3, R_{\min} = 0.1$).

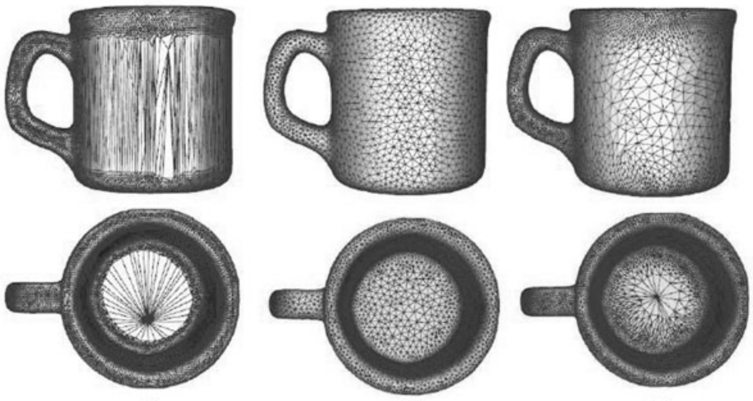

Figure 13 Comparison between remesh methods.(a) The original model;(b) our algorithm; (c) Yue's [30] algotithm.

interaction. This can be improved by setting initial value of the parameters. Second, the algorithm needs connectivity information of the model, and it will fail when dealing with the non-manifold model. We can solve this by preprocessing models. Finally, when $R_{\max} \gg R_{\min}$, the result of adaptive sampling is not good enough because the errors introduced by sampling points with large radius influence sampling points with small radius. This problem can be solved via improving the adaptive method.

6 Conclusion and future work

We present an algorithm for generating Poisson disk distribution directly on mesh. Area based triangle index structure and point selection method are the kernel of our algorithm. A mesh clipping method is used to ensure the efficiency of the index structure; collision detection frame based on Isophotic distance metric is introduced in order to keep distance condition satisfied. Our algorithm is easy to implement, fast and robust. Sampling results show good blue noise properties. Some minor modifications will make the algorithm suitable for feature preserving sampling and adaptive sampling, suggesting that this algorithm has good flexibility. As for future work, isophotic distance metric will be used to implement the Poisson disk sampling on non-manifold surfaces and point clouds. Parallel Poisson disk sampling and spectrum analysis of Poisson disk distribution on manifold surfaces will also be considered.

Acknowledgements

This work was supported by National Basic Program of China (Grant No. 2010CB328002) and National Natural Science Foundation of China (Grant Nos. 60925007, 60833007, 90915010). We thank Zhu LiFeng, Li ShengRen for their valuable opinions, and thank Zhou BingFen, Ares Lagae, Gamito Manuel and He Tao for their helps in experiment. We also thank Schlömer for providing PSA analysis tools1) . The models are from AIM@SHAPE model repository and Mesh Segmentation BenchMark [31].

References

1 Dippé M A Z, Wold E H. Antialiasing through stochastic sampling. In: Proceedings of the SIGGRAPH Conference. New York: ACM, 1985. 69–78

2 Cook R L. Stochastic sampling in computer graphics. ACM Trans Graphic, 1986, 5: 51–72

3 Mitchell D P. Generating antialiased images at low sampling densities. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '87. New York: ACM, 1987. 65–72

4 Hachisuka T, Jarosz W, Weistroffer R P, et al. Multidimensional adaptive sampling and reconstruction for ray tracing. ACM Trans Graphic, 2008, 27: Article No. 33

5 Lehtinen J, Zwicker M, Turquin E, et al. A meshless hierarchical representation for light transport. ACM Trans Graphic, 2008, 27: Article No. 37

6 Mitchell D P. Spectrally optimal sampling for distribution ray tracing. SIGGRAPH Comput Graph, 1991, 25: 157–164

7 Deussen O, Hanrahan P, Lintermann B, et al. Realistic modeling and rendering of plant ecosystems. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '98. New York: ACM, 1998. 275–286

8 Deussen O, Hiller S, van Overveld C, et al. Floating points: a method for computing stipple drawings. Comput Graph Forum, 2000, 19: 40–51

9 Lagae A, Dutré P. A comparison of methods for generating Poisson disk distributions. Comput Graph Forum, 2008, 27: 114–129

10 Gamito M N, Maddock S C. Accurate multidimensional Poisson-disk sampling. ACM Trans Graphic, 2009, 29: Article No.8

11 Jones T R. Efficient generation of Poisson-disk sampling patterns. J Graph Tool, 2006, 11: 27–36

12 Dunbar D, Humphreys G. A spatial data structure for fast Poisson-disk sample generation. ACM Trans Graphic, 2006, 25: 503–508

13 White K B, Cline D, Egbert P K. Poisson disk point sets by hierarchical dart throwing. In: Proceedings of the IEEE Symposium on Interactive Ray Tracing. Washington DC: IEEE Computer Society, 2007. 129–132

1) Schlömer T. PSA, http://code.google.com/p/psa/

14 Wei L. Parallel Poisson disk sampling. ACM Trans Graphic, 2008, 27: Article No. 20

15 Lloyd S P. Least squares quantization in PCM. IEEE Trans Inform Theory, 1982, 28: 129–137

16 Balzer M, Schlömer T, Deussen O. Capacity-constrained point distributions: a variant of Lloyd's method. In: ACM SIGGRAPH. New York: ACM, 2009. Article No. 86

17 Liu Y, Wang W, L vy B, et al. On centroidal voronoi tessellation-energy smoothness and fast computation. ACM Trans Graphic, 2009, 28: Article No. 101

18 Ostromoukhov V, Donohue C, Jodoin P. Fast hierarchical importance sampling with blue noise properties. In: ACM SIGGRAPH. New York: ACM, 2004. 488–495

19 Ostromoukhov V. Sampling with polyominoes. ACM Trans Graphic, 2007, 26: Article No. 78

20 Hiller S, Deussen O, Keller A. Tiled blue noise samples. In: Proceedings of the Vision Modeling and Visualization Conference. Aka GmbH, 2001. 265–272

21 Lagae A, Dutré P. A procedural object distribution function. ACM Trans Graphic, 2005, 24: 1442–1461

22 Lagae A, Dutré P. An alternative for wang tiles: colored edges versus colored corners. ACM Trans Graphic, 2006, 25: 1442–1459

23 Fu Y, Zhou B. Direct sampling on surfaces for high quality remeshing. Comput Aided Geom D, 2009, 26: 711–723

24 Cline D, Jeschke S, White K, et al. Dart throwing on surfaces. Comput Graph Forum, 2009, 28: 1217–1226

25 Li H, Lo K, Leung M, et al. Dual Poisson-disk tiling: an efficient method for distributing features on arbitrary surfaces. IEEE Trans Vis Comput Gr, 2008, 14: 982–998

26 Pottmann H, Steiner T, Hofer M, et al. The isophotic metric and its applications to feature sensitive morphology on surfaces. LNCS, 2004, 3024: 560–572

27 Kim H S, Choi H K, Lee K H. Feature detection of triangular meshes based on tensor voting theory. Comput Aided Design, 2009, 41: 47–58

28 Ulichney R A. Digital Halftoning. Boston: MIT Press, 1987. 189–205

29 McCool M, Fiume E. Hierarchical Poisson disk sampling distributions. In: Proceedings of the Conference on Graphics interface'92. San Francisco: Morgan Kaufmann Publishers Inc., 1992. 94–105

30 Yue W N, Guo Q W, Zhang J, et al. 3D triangular mesh optimization for geometry processing in CAD. In: ACM International Conference on Solid and Physical Modeling 2007 (SPM '07). New York: ACM, 2007. 23–34

31 Chen X, Golovinskiy A, Funkhouser T. A benchmark for 3D mesh segmentation. In: ACM SIGGRAPH. New York: ACM, 2009. Article No. 73