Communications in Information and Systems
Volume 18, Number 2, 73–89, 2018

An approximate method for circle packing and disc covering

Shuangmin Chen, Yuen-Shan Leung, Shiqing Xin[†], Ying He, Yuanfeng Zhou, and Changhe Tu

Circle packing is to optimize the arrangement of circles (of equal or varying sizes) on a given domain with the maximal packing density such that no overlapping occurs. As an NP-hard problem, it is scientifically challenging so that no procedure is able to exactly solve the problem in deterministic polynomial time even for the Euclidean domains. In this paper, we develop an approximate method for packing a large number of circles (of similar sizes). In contrast to the existing methods that use nonlinear optimization with carefully designed strategies to deal with complex boundaries, our methods are purely geometrical and highly intuitive. Observing that circle packing is closely related to disc covering, we formulate both problems in a centroidal Voronoi tessellation (CVT)-like computational framework and show that a locally optimal solution to the covering (resp. packing) problem can be obtained by iteratively updating the centers of the circumscribed (resp. inscribed) circles of the Voronoi cells. Using geodesic exponential map, we can compute those centers efficiently on manifold triangle meshes, hereby extending our 2D algorithm to non-Euclidean domains. Experimental results on synthetic and real-world models demonstrate the efficacy of our method.

1. Introduction

Circle packing is to optimize the arrangement of circles (of equal or varying sizes) on a given domain with the maximal packing density such that no overlapping occurs. It has a wide range of applications in engineering field [1, 2, 5, 13, 21]. However, as an NP-hard problem, it is scientifically challenging so that no procedure is able to exactly solve the problem in deterministic polynomial time even for the Euclidean domains [7]. Hence, the existing *exact* methods are only able to pack tens of circles in simple

[†] Shiqing Xin is the corresponding author of this paper.

domains, such as a square or a disk [9, 17]. To pack circles in irregular or non-Euclidean domains, there are some theoretical research results [3, 19], but practical algorithms are quite few, to our best knowledge. Available *approximate* methods [10, 11, 14, 15, 22] are often driven by nonlinear optimization with carefully designed strategies to handle boundary condition. We refer readers to [12] for a comprehensive survey.

In this paper, we focus on a general setting for packing an arbitrary number of circles of similar sizes in non-Euclidean domains. Due to a lack of closed-form of geodesic distances, the above-mentioned methods cannot be extended for our case. Observing that circle packing is closely related to disc covering, which is to find the minimum number of disks (of user-specified sizes) such that the domain of interest is totally covered, we formulate both problems in a centroidal Voronoi tessellation (CVT)-like optimization framework as follows: Given a manifold triangle mesh M and a set of points $\{\mathbf{c}_i\}_{i=1}^n$ on M, we compute the geodesic Voronoi diagram that partitions M into n disjoint regions. We show that the supremum of the radii of the circumscribed circles of the Voronoi cells is the size of disk to cover the domain Ω, while the infimum of the radii of the inscribed circles is disk size for packing. Therefore, we can obtain the local optimal solution to the covering (resp. packing) problem by iteratively updating the centers of the circumscribed (resp. inscribed) circles of the Voronoi cells.

Since it is technically challenging to directly compute those centers in non-Euclidean domains, we adopt an indirect approach that first generates a set of randomly and uniformly distributed samples $\{\mathbf{s}_j\}$ on M and then approximately computes the centers using farthest Voronoi diagrams. We prove that if the samples $\{\mathbf{s}_j\}$ satisfy the ε-condition, i.e., for any point $\mathbf{q} \in M$, there exists a sample $\mathbf{s}_k$ whose distance to $\mathbf{q}$ is less than ε, the errors of the computed centers are less than ε. Based on our theoretical results, we develop a simple Lloyd-like method to iteratively compute the centers of circumscribed and inscribed circles. Experimental results on synthetic and real-world models demonstrate the efficacy of our method. See Figure 1 for an example on the Bunny model.

2. Problem statement and main results in 2D

In this section, we present the main results in Euclidean domains. Then we extend it to non-Euclidean domains in the next section. Let $\Omega \subset \mathbb{R}^2$ be a compact and connected 2D domain. Denote by $\partial\Omega$ the boundary of Ω. We assume the packing circles and covering disks are of the same size and their number is fixed, say, n. Our goal is to optimize the arrangement of the centers

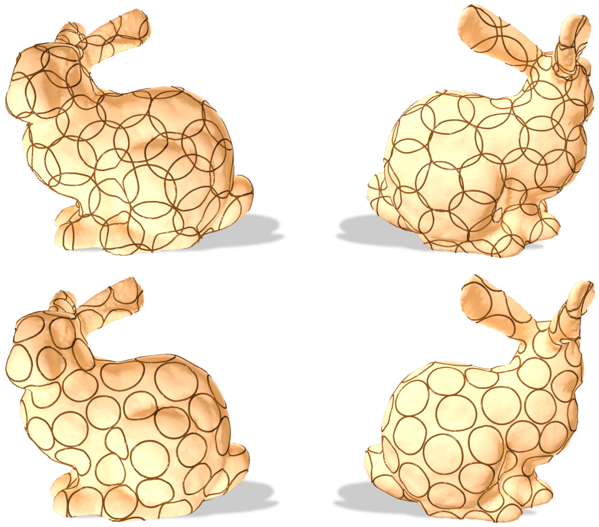


Figure 1: Our method provides approximate solutions to the disc covering (the top row) and circle packing (the bottom row) problems on triangle meshes.

$\{\mathbf{c}_i\}_{i=1}^n$ such that the disk covering/packing radii reach the extremum. To be more specific, we answer the following questions:

Q1: What's the minimum radius R for covering Ω using the given number of disks?

Q2: What's the maximum radius r for packing the given number of circles into Ω?

Mathematically, the covering radius R is given by

$$R = \sup_{\mathbf{x}} \min_{i} \|\mathbf{x} - \mathbf{c}_i\|, \mathbf{x} \in \Omega, \tag{1}$$

while the packing radius r is

$$r = \min\left(\frac{1}{2} \times \min_{i \neq j} \|\mathbf{c}_i - \mathbf{c}_j\|, \min_{1 \leq i \leq n, \mathbf{x} \in \partial\Omega} \|\mathbf{x} - \mathbf{c}_i\|\right). \tag{2}$$

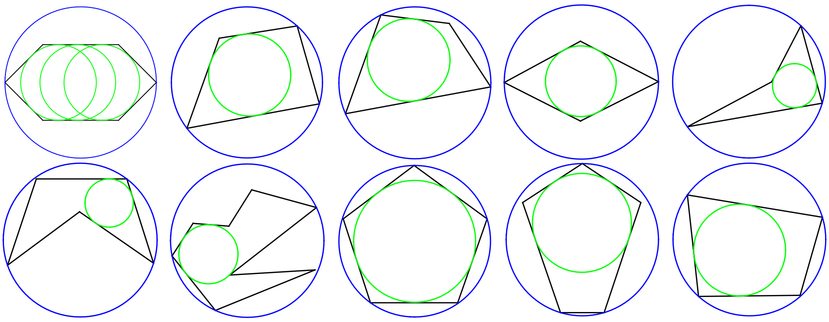


Figure 2: Circumscribed and inscribed circles of polygons. Note that the circumscribed circles (blue) are unique, whereas the inscribed circles (green) are not (e.g., the left-most polygon).

Most of the existing methods [10, 11, 14, 15, 22] solve the packing and covering problems using non-linear optimization. However, since the objective functions are highly non-linear, their performance is quite low. Moreover, as mentioned above, it is difficult to extend those methods to non-Euclidean domains due to lack of closed-form formula to measure geodesic distances.

Our method is based on Voronoi diagrams. Let Ω_i denote the Voronoi cell for point $\mathbf{c}_i$. Before presenting the technical details, we define the circumscribed circle and inscribed circle of Ω_i. The circumscribed circle is the *smallest* circle that can cover Ω_i, while the inscribed circle is the *largest* circle contained in Ω_i. Note that our definitions are slightly different from the conventional ones where the circumcircle has to pass through all the vertices of the given polygon and the inscribed circle has to touch all the edges. It is easy to prove that the circumcircle is unique, whereas the inscribed circle is not. See Figure 2 for examples of inscribed and circumscribed circles for polygons.

We are motivated by the Lloyd algorithm for computing centroidal Voronoi tessellation [4], a Voronoi diagram whose generating points are the centroids (centers of mass) of the corresponding Voronoi regions. Lloyd's algorithm computes CVT in an iterative manner. For each iteration, it improves the result by moving the generators of the Voronoi regions to the corresponding centers of masses. Lloyd's method is conceptually simple and easy to implement. It decreases the CVT energy monotonically at each iteration and is guaranteed to converge. As a gradient decent method, Lloyd's method has a linear convergence rate.

Similar to Lloyd's method, our method also starts with random points in the domain Ω. At each iteration, it computes the Voronoi diagram for the given generators, and then moves them to the centers of the circumscribed/inscribed circles of the Voronoi cells. Theorem 1 guarantees that the algorithm converges in finite steps under certain assumptions.

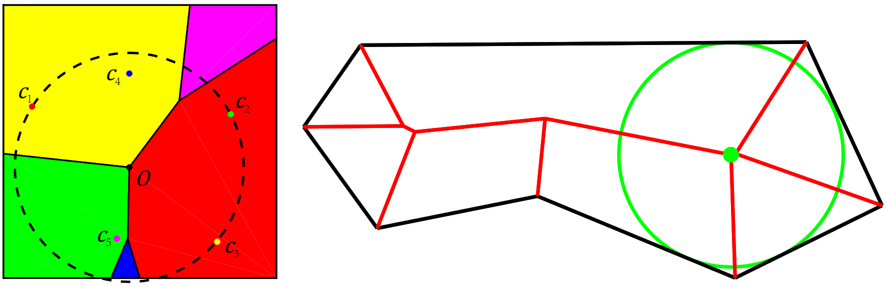
We show the pseudocode of our algorithm in Algorithm 1 and illustrate it on a unit square with $n = 4$ discs/circles in Figure 4. Since the final result is dependent on the initial positions, our algorithm is locally optimal, which can be observed from the circle packing result.

```
Input: $\Omega$: a compact and connected 2D domain;
       $n$: the number of circles/discs;
       $\delta$: the convergence threshold.
Output: The covering (resp. packing) radius and the centers of the
        disks (resp. circles).
Generate $n$ random points $\{\mathbf{c}_i^{(0)} | \mathbf{c}_i^{(0)} \in \Omega, i = 1, \ldots, n\}$;
do
    Taking $\{\mathbf{c}_i^{(k)}\}$ as the generators, compute the Voronoi diagram;
    for each Voronoi cell $\Omega_i$ do
        Compute the center of the circumscribed (resp. inscribed)
          circle of $\Omega_i$;
        Update $\mathbf{c}_i^{(k+1)}$ to the center;
    end
    $k \leftarrow k + 1$;
while $\max_i \|\mathbf{c}_i^{(k+1)} - \mathbf{c}_i^{(k)}\| \geq \delta$;
Compute the radius of the packing circles (resp. covering discs).
```

Algorithm 1: Solving the circle packing and disc covering problems in 2D.

Theorem 1. *Algorithm 1 converges in finite steps if the domain is convex.*

Proof. The proof is based on the following key observations:



(a) (b)

Figure 3: (a) The farthest-point Voronoi diagram of 5 sites partitions the plane into 5 regions. Each site $\mathbf{c}_i$ is associated with a convex polygonal region $\mathcal{C}(\mathbf{c}_i)$ (in the same color) such that $\mathbf{c}_i$ is the farthest neighbor of every point in $\mathcal{C}(\mathbf{c}_i)$. The smallest enclosing circle passes through points c_1, c_2 and c_3. The center O is the point common to $\mathcal{C}(c_1)$, $\mathcal{C}(c_2)$ and $\mathcal{C}(c_3)$. (b) The Voronoi diagram induced by the edge set is exactly the skeleton of the polygon. The maximum inscribed circle must be centered at one of the vertices of the skeleton.

- The center of the circumscribed circle of Ω_i must be a vertex of the farthest-point Voronoi diagram [1] of the vertices of Ω_i [20]. If the circumscribed circle passes through sites $\mathbf{c}_{i_1}, \ldots, \mathbf{c}_{i_k}$, the center is the point common to the farthest-point Voronoi cells of these sites (see Figure 3(a)).
- The center of the inscribed circle of Ω_i is a vertex of the Voronoi diagram of the edges of Ω_i (see Figure 3(b)).

During the k-the iteration, each generator $\mathbf{s}_i^k, 1 \leq i \leq n$, dominates a Voronoi cell Ω_i^k in the Voronoi decomposition. Let $\bigodot(C_i^{(k)}, R_i^{(k)})$ and $\odot(c_i^k, r_i^k)$ be the circumscribed and inscribed circles of Ω_i^k. Our iterative scheme implies that $s_i^{(k+1)}$ is exactly $C_i^{(k)}$ in the disc covering case while $c_i^{(k)}$ in the circle packing case. Obviously, $\bigcup \bigodot(C_i^{(k)}, R^{(k)})$ covers the domain Ω and $\bigcup \odot(c_i^{(k)}, r^{(k)})$ is a valid packing. Let $R^{(k)} \triangleq \max\{R_i^{(k)}\}$ and $r^{(k)} \triangleq \min\{r_i^{(k)}\}$.

[1] The farthest-point Voronoi diagram associates each site $\mathbf{c}_i$ a convex polygonal region such that $\mathbf{c}_i$ is the farthest neighbor of every point in the region. The farthest-point Voronoi diagram is also known as the $(n-1)$-th order Voronoi diagram for a set of n generators[20].

In the following, we first prove the convergence for the disc covering algorithm. The key is to show that each circumscribed circle $\bigodot(C_i^k, R^k)$ in the previous iteration is able to cover the Voronoi cell $\Omega_i^{(k+1)}$, dominated by $s_i^{(k+1)}$ or equivalently C_i^k, in the next iteration. Consider an arbitrary point $p \in \Omega_i^{(k+1)}$. Since in the k-th iteration $\bigcup \bigodot(C_i^{(k)}, R^{(k)})$ covers the whole domain Ω, the point p must be inside some circle(s). We further observe that p must be inside $\bigodot(C_i^{(k)}, R^{(k)})$ since p is a point belong to $\Omega_i^{(k+1)}$, i.e. p is closer to $s_i^{(k+1)}$ $(C_i^{(k)})$ than $s_j^{(k+1)}$ $(C_j^{(k)}), j \neq i$. Due to the arbitrariness of p, we conclude that $\bigodot(C_i^{(k)}, R^{(k)})$ is able to cover the cell $\Omega_i^{(k+1)}$. Therefore, $R^{(k+1)}$ must be less than or equal to $R^{(k)}$, which implies the convergence. Note that the center of $\bigodot(C_i^{(k)}, R^{(k)})$ must be inside the domain Ω due to the fact that the center of the minimum enclosing circle of a convex shape must be located in the interior of the shape or on the boundary of the shape.

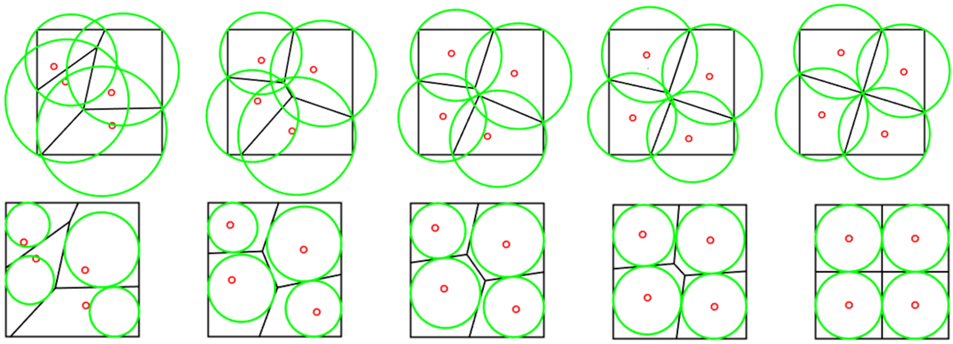
Next we come to the circle packing algorithm. Similar to disc covering, our goal is to show that each inscribed circle $\odot(c_i^{(k)}, r^{(k)})$ in the previous iteration is located inside the Voronoi cell $\Omega_i^{(k+1)}$, dominated by $s_i^{(k+1)}$ or equivalently $c_i^{(k)}$, in the next iteration. That is to say, for any point p on the boundary of $\Omega_i^{(k+1)}$, we cannot find an index $\exists j$ such that both distances $\|ps_i^{(k+1)}\|$ $(\|pc_i^{(k)}\|)$ and $\|ps_j^{(k+1)}\|$ $(\|pc_j^{(k)}\|)$ are less than $r^{(k)}$. This is true since any two generators are at least $2r^{(k)}$ apart in the k-th iteration. □

Remark 1. The convergence means the monotonic change of the radii of the circumscribed (inscribed) circles, which is independent of the centers of the circumscribed (inscribed) circle sequence. Therefore, although the inscribed circle may be not unique, our algorithm can still converge.

Remark 2. When the algorithm converges, the radii of the circumscribed (inscribed) circles of Voronoi regions may be different.

3. An approximate algorithm for computing circumscribed/inscribed circles

The key step in Algorithm 1 is to compute the circumscribed/inscribed circle for each Voronoi cell. For 2D domains, many efficient algorithms are readily available. For example, the sweep line algorithm [6] generates a Voronoi diagram from a set of n 2D points using $O(n \log n)$ time and $O(n)$ space. Megiddo's algorithm [18] computes the circumscribed circles for 2D polygons. The algorithm recursively enlarges the enclosing circle for a progressive point set until the circle covers all the points. Megiddo's algorithm takes $O(k)$ time to determine the smallest enclosing circle for k points. For



(a) Initialization (b) Iter #1 (c) Iter #3 (d) Iter #6 (e) Iter #10

Figure 4: Illustration of our 2D covering (row 1) and packing (row 2) algorithms in a unit square for $n = 4$. With random initialization, both algorithms converge in only 10 iterations. Note that the circle packing result is *locally* optimal.

a 2D polygon, the maximum inscribed circle must touch boundary edges or concave vertices at at least 3 points. Garcia-Castellanos and Lombardo's algorithm [8] finds the best vertex-edge triplet by iteratively increasing the radius of the up-to-date inscribed circle.

Extending Algorithm 1 to triangle meshes is non-trivial due to two reasons: First, many algorithms in Euclidean spaces take for granted the readily available distance function. Second, many properties in Euclidean spaces do not hold in non-Euclidean domains due to the fundamental differences in geometry and topology. For example, a key step in Megiddo's algorithm [18] is to move the center along the bisector of two supporting points. Given two points on a polyhedral surface, there may exist multiple geodesic paths of equal length, hence the bisectors are not unique, making it difficult to generalize Megiddo's algorithm.

To tackle the above-mentioned challenges and develop a unified framework for both Euclidean and non-Euclidean domains, we propose an indirect approach for computing circumscribed and inscribed circles. Our key idea is to sample the given domain using a set of randomly and uniformly distributed points $\{\mathbf{s}_j\}$. We prove that if the samples $\{\mathbf{s}_j\}$ satisfy the ε-condition, i.e., for any point $\mathbf{q} \in M$, there exists a sample $\mathbf{s}_k$ whose distance to $\mathbf{q}$ is less than ε, the errors of the computed centers are less than ε. In the following, we describe the method for 2D domains and then extend it to triangle meshes in the next section.

Definition 2. Let $S = \{\mathbf{s}^j\}$ be a sample set in $\Omega \subset \mathbb{R}^2$. We say S is an ϵ-dense set w.r.t. Ω if for any $p \in \Omega$, there exists a sample point $\mathbf{s}^j \in S$ such that $\|\mathbf{s}^j - p\| \leq \epsilon$.

Similarly, we can define the ϵ-dense sample set $\{\mathbf{s}^j|_{\partial\Omega}\}$ w.r.t. the boundary $\partial\Omega$, where each sample point $\mathbf{s}^j|_{\partial\Omega}$ is located on $\partial\Omega$.

Our goal is to approximate the circumscribed and inscribed disk radii using an ϵ-dense sample set $\{\mathbf{s}^j\} \bigcup \{\mathbf{s}^j|_{\partial\Omega}\}$. The following theorem shows the approximation error is determined by the sampling density ϵ.

Theorem 3. *Given a set of generators $\{c_i\}$, the domain Ω is partitioned into a set of Voronoi cells $\{\Omega_i\}$. Let $\{\mathbf{s}^j\}$ be an ϵ-dense set of sample points w.r.t. $\Omega \subset \mathbb{R}^2$ and $\{\mathbf{s}^j|_{\partial\Omega}\}$ be ϵ-dense w.r.t. the boundary $\partial\Omega$. If we approximate the circumscribed disk radius R_i and the inscribed disk radius r_i according to the sample set as discussed above, then the errors are bounded by*

$$0 \leq R_i - R_i' \leq \epsilon \quad \textit{and} \quad 0 \leq r_i' - r_i \leq \epsilon.$$

Proof. Denote by $\mathcal{D}(\widetilde{c}_i^{\otimes}, R_i')$ the circumscribed disk of $\{\mathbf{s}_i^j\}$ that is ϵ-dense. It is easy to show that the disk $\mathcal{D}(\widetilde{c}_i^{\otimes}, R_i' + \epsilon)$ is able to cover every point $p \in \Omega_i$, and thus $R_i' + \epsilon \geq R_i$. Combining with $R_i' \leq R_i$, we have $0 \leq R_i - R_i' \leq \epsilon$.

Similarly, when we shrink the approximate inscribed disk $\mathcal{D}(\widetilde{c}_i^{\odot}, r_i')$ into $\mathcal{D}(\widetilde{c}_i^{\odot}, r_i' - \epsilon)$, it must be completely located in Ω_i, the Voronoi region of c_i. Therefore, $0 \leq r_i' - r_i \leq \epsilon$ also holds. □

4. Extension to triangle meshes

Although Theorem 2 is presented for 2D domains, its idea can be extended for computing circumscribed and inscribed disks on triangle meshes, given effective tools for measuring geodesic distances and constructing geodesic Voronoi diagrams (GVD).

Note that the state-of-the-art algorithms [16, 23] are able to compute *exact* GVDs on arbitrary manifold triangle meshes. However, they are computationally expensive. For a triangle mesh with n vertices and k $(\leq n)$ generators, the state-of-the-art algorithms run in $O(n^2 \log n)$ time. To improve the performance, we use the saddle vertex graph (SVG) method [24]. Let $M = (V, E, F)$ be the input manifold triangle mesh, where V, E and F are the sets of vertices, edges and faces, respectively. SVG solves the discrete geodesic problem by constructing an undirected graph G so that a geodesic

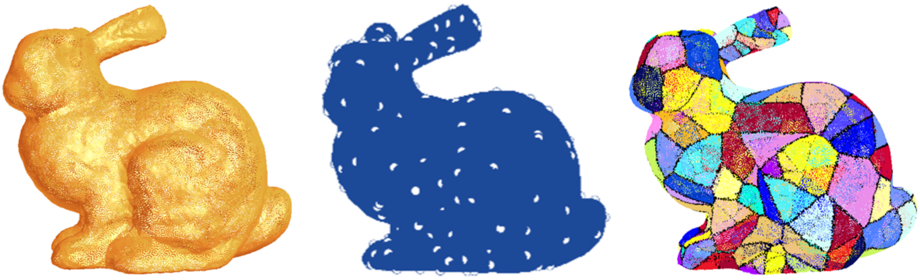


Figure 5: Computing geodesic Voronoi diagram by clustering samples.

path on the mesh M is (approximately) equal to a shortest path on G. As a pre-computation method, SVG enables information reuse and allows us to compute geodesic distances efficiently and accurately.

Let $\{\mathbf{s}^j\}$ denote the samples on M. We randomly generate n sites $\{\mathbf{c}_i^{(0)}\} \in M$ as the initial covering (resp., packing) disk centers. To compute the geodesic Voronoi diagram, we take $\{\mathbf{c}_i^{(k)}\}$ as the source points and compute the geodesic distance field using the saddle vertex graph G, which assigns each sample a distance to its nearest source. If a sample is assigned two approximately equal distances, it is on the bisector of the corresponding sources. We then cluster all samples $\{\mathbf{s}^j\}$ into n groups, each of which is an *approximate* Voronoi cell (see Figure 5). To compute the the inscribed/circumscribed disk center, we adopt the exponential map, which maps all samples of a Voronoi cell to the tangent plane so that the 2D algorithm in Section 2 can be applied.

5. Results

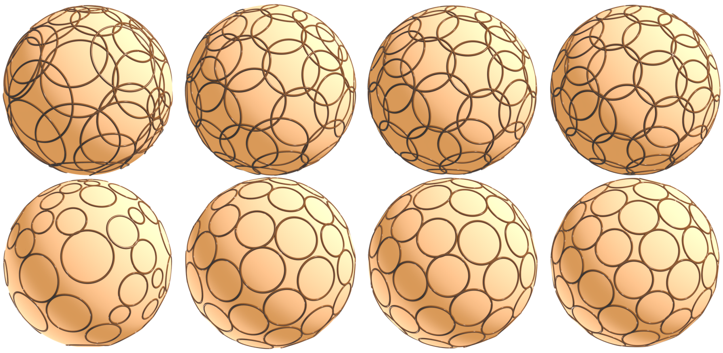
We implemented our algorithms in C++ and tested them on a PC with an Intel Xeon 2.66GHz CPU and 8GM memory. Figure 6 illustrates our algorithm on a unit sphere, and Figure 5 and Figure 7 show the results on the common 3D models. Table 1 gives detailed experimental data statistics. For example, we observe that our algorithm takes about 50 iterations to reach the stable results and each iteration takes less than a minute.

```
Input: $\mathcal{M}$: a manifold triangle mesh; $n$ the number of disks; $\epsilon$: the
       given tolerance; $\delta$: the convergence tolerance.
Output: The covering (resp., packing) radii and the disk centers
        $\{\mathbf{c}_i^*\}$.
Generate a set of sample points $\{\mathbf{s}^j\}$, where both $\{\mathbf{s}^j|_M\}$ and
  $\{\mathbf{s}^j|_{\partial M}\}$ are with a density $\epsilon$;
Construct a saddle vertex graph $G$ on $M$ to facilitate distance
  computation;
Randomly generate $n$ sites $\{\mathbf{c}_i^{(0)}\} \in M$ as the initial covering (resp.,
  packing) disk centers;
do
    Taking $\{\mathbf{c}_i^{(k)}\}$ as the source points, compute the geodesic
      distance field using $G$ and cluster $\{\mathbf{s}^j\}$ into $n$ groups;
    for each cluster do
        Compute the exponential map centered at $\mathbf{c}_i^{(}k)$;
        Map the samples in the cluster to the tangent plane $TP_{\mathbf{c}_i^{(k)}}$;
        Compute the center of the circumscribed (resp., inscribed)
          disk center $\mathbf{c}_i^{(k+1)}$;
    end
while $\max_i |\mathbf{c}_i^{(k+1)} - \mathbf{c}_i^{(k)}| \geq \delta$;
Compute the covering (resp., packing) disk radius.
```

Algorithm 2: Solving the circle packing and disc covering problems on triangle meshes.

Table 1: The detailed experimental data statistics

Model	#Vertex	#Seed	#Iteration	Time(s) per iteration
Fig. 1 (top)	120002	100	78	7.43
Fig. 1 (bottom)	120002	100	69	7.16
Fig. 6 (top)	655362	100	47	19.41
Fig. 6 (bottom)	655362	100	52	18.54
Fig. 7 (#1)	90496	100	44	11.25
Fig. 7 (#2)	90496	100	59	10.96
Fig. 7 (#3)	90496	300	53	15.21
Fig. 7 (#4)	90496	300	82	15.87



Initialization Iteration #10 Iteration #20 Iteration #50

Figure 6: Illustration of our algorithm on the unit sphere. The first row shows how the covering radius decreases and the second row shows how the packing radius increases.

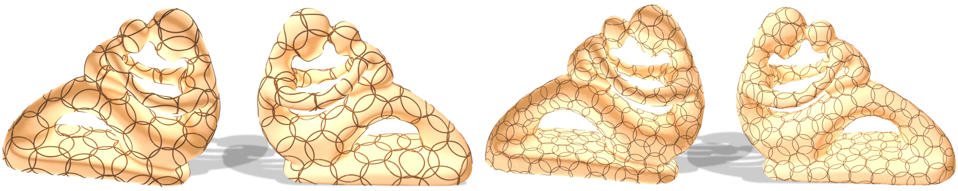
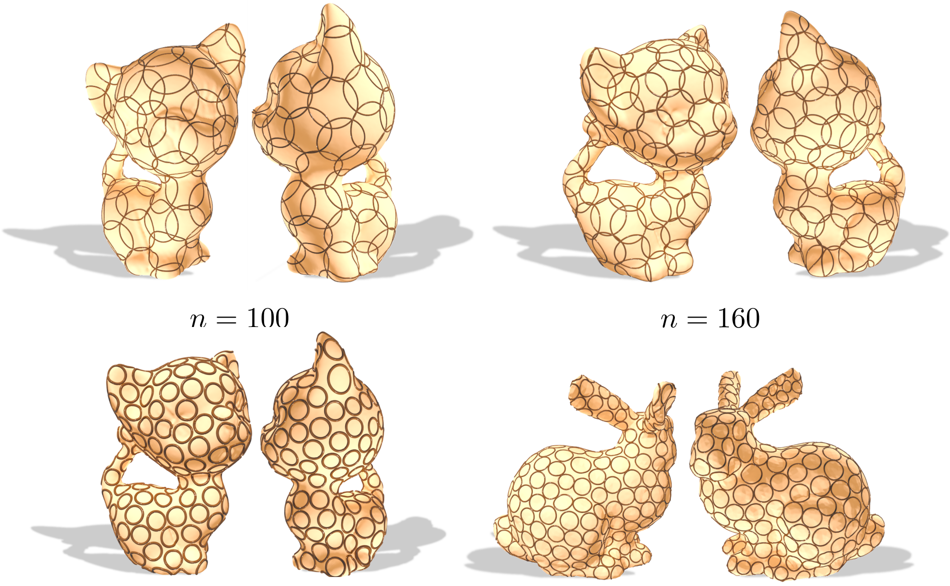


Figure 7: Disc covering with $n = 100$ and 300 on the Fertility model.

6. Conclusions

We developed efficient algorithms for the circle packing and disc covering problems on polyhedral surfaces. In contrast to the existing methods that use nonlinear optimization with carefully designed strategies to deal with complex boundaries, our methods are purely geometrical and highly intuitive. We formulated both problems in a centroidal Voronoi tessellation (CVT)-like computational framework and showed that a locally optimal solution to the covering (resp. packing) problem can be obtained by iteratively updating the centers of the circumscribed (resp. inscribed) circles of the Voronoi cells. Using saddle vertex graph and exponential map, we developed efficient methods to compute those centers on arbitrary manifold triangle meshes.



$n = 100$

$n = 160$

Figure 8: More results on 3D meshes.

Experimental results on synthetic and real-world models demonstrated the efficacy of our method.

Our method has two limitations, which are worth of further improvement. First, although the produced circles/discs are similar, it cannot guarantee they have the same radius. Second, similar to Lloyd's method for computing CVT, our method computes only the local optimal solution.

Acknowledgement

We are very grateful to the editors and anonymous reviewers for their insightful comments and suggestions. This work is partially supported by NSF of China (61772016, 61772312), China 973 Program (2015CB352502) and the key research and development project of Shandong province (2017GGX10110).

References

[1] U. Bücking, *Approximation of conformal mappings by circle patterns*, Geometriae Dedicata **137** (2008), no. 1, 163–197.

[2] I. Castillo, F. J. Kampas, and J. D. Pintér, *Solving circle packing problems by global optimization: numerical results and industrial applications*, European Journal of Operational Research **191** (2008), no. 3, 786–802.

[3] H. Coxeter, *Arrangements of equal spheres in non-Euclidean spaces*, Acta Mathematica Academiae Scientiarum Hungarica **5** (1954), no. 3-4, 263–274.

[4] Q. Du, V. Faber, and M. Gunzburger, *Centroidal Voronoi tessellations: applications and algorithms*, SIAM Rev. **41** (1999), no. 4, 637–676.

[5] D. Eppstein, *Faster circle packing with application to nonobtuse triangulation*, International Journal of Computational Geometry & Applications **7** (2011), no. 5, 485–491.

[6] S. Fortune, *A sweepline algorithm for Voronoi diagrams*, in: Proceedings of the Second Annual Symposium on Computational Geometry, 313–322 (1986).

[7] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto, *Optimal packing and covering in the plane are NP-complete*, Information processing letters **12** (1981), no. 3, 133–137.

[8] D. Garcia-Castellanos and U. Lombardo, *Poles of inaccessibility: A calculation algorithm for the remotest places on earth*, Scottish Geographical Journal **123** (2007), no. 3, 227–233.

[9] T. Gensane, *Dense packings Of equal spheres in a cube*, Journal of Combinatorics **11** (2004), no. 1, 295–297.

[10] J. A. George, J. M. George, and B. W. Lamar, *Packing different-sized circles into a rectangular container*, European Journal of Operational Research **84** (1995), no. 95, 693–712.

[11] A. Grosso, A. Jamali, M. Locatelli, and F. Schoen, *Solving the problem of packing equal and unequal circles in a circular container*, Journal of Global Optimization **47** (2010), no. 1, 63–81.

[12] M. Hifi and R. M'hallah, *A literature review on circle and sphere packing problems: models and methodologies*, Advances in Operations Research (2009).

[13] L. Kharevych, B. Springborn, and P. Schröder, *Discrete conformal mappings via circle patterns*, ACM Transactions on Graphics (TOG) **25** (2006), no. 2, 412–438.

[14] I. Litvinchev, L. Infante, and E. L. O. Espinosa, Approximate Circle Packing in a Rectangular Container: Integer Programming Formulations and Valid Inequalities, Springer International Publishing (2014).

[15] I. S. Litvinchev and L. Ozuna, *Approximate packing circles in a rectangular container: valid inequalities and nesting*, Journal of Applied Research & Technology **12** (2014), no. 4, 716–723.

[16] Y. Liu, Z. Chen, and K. Tang, *Construction of iso-contours, bisectors, and Voronoi diagrams on triangulated surfaces*, IEEE Trans. Pattern Anal. Mach. Intell. **33** (2011), no. 8, 1502–1517.

[17] B. D. Lubachevsky and R. L. Graham, *Minimum perimeter rectangles that enclose congruent non-overlapping circles*, Discrete Mathematics **309** (2009), no. 8, 1947–1962.

[18] N. Megiddo, *Linear-time algorithms for linear programming in R3 and related problems* **12** (1983), no. 4, 329–338.

[19] D. Minda and B. Rodin, *Circle packing and Riemann surfaces*, Journal D'Analyse Mathématique **57** (1991), no. 1, 221–249.

[20] M. I. Shamos and D. Hoey, *Closest-point problems*, in: Proceedings of the 16th Annual Symposium on Foundations of Computer Science (1975), 151–162.

[21] K. Stephenson, *The approximation of conformal structures via circle packing*, Series in Approximations and Decompositions **11** (1999), 551–582.

[22] Y. G. Stoyan and G. N. Yaskov, *Mathematical model and solution method of optimization problem of placement of rectangles and circles taking into account special constraints*, International Transactions in Operational Research **5** (1998), no. 1, 45–57.

[23] C. Xu, Y. Liu, Q. Sun, J. Li, and Y. He, *Polyline-sourced geodesic Voronoi diagrams on triangle meshes*, Comput. Graph. Forum **33** (2014), no. 7, 161–170.

[24] X. Ying, X. Wang, and Y. He, *Saddle vertex graph (SVG): a novel solution to the discrete geodesic problem*, ACM Trans. Graph. **32** (2013), no. 6, 170:1–170:12.

School of Information and Technology
Qingdao University of Science and Technology
Qingdao, 266061, China
E-mail address: `csmqq@163.com`

School of Computer Engineering
Nanyang Techonological University
639798, Singapore
E-mail address: `debbieleung22@gmail.com`

School of Computer Science and Technology
Shandong University at Qingdao
Qingdao, 266237, China
E-mail address: `xinshiqing@sdu.edu.cn`

School of Computer Engineering
Nanyang Techonological University
639798, Singapore
E-mail address: `yhe@ntu.edu.sg`

School of Software, Shandong University
Jinan, 250101, China
E-mail address: `yfzhou@sdu.edu.cn`

School of Computer Science and Technology
Shandong University at Qingdao
Qingdao, 266237, China
E-mail address: `chtu@sdu.edu.cn`