<table>
<thead>
<tr>
<th>Title</th>
<th>Manifold Differential Evolution (MDE) : A global optimization method for geodesic Centroidal Voronoi Tessellations on meshes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author(s)</td>
<td>Liu, Yong-Jin; Xu, Chun-Xu; Yi, Ran; Fan, Dian; He, Ying</td>
</tr>
<tr>
<td>Date</td>
<td>2016</td>
</tr>
<tr>
<td>URL</td>
<td><a href="http://hdl.handle.net/10220/46265">http://hdl.handle.net/10220/46265</a></td>
</tr>
<tr>
<td>Rights</td>
<td>© 2016 Association for Computing Machinery (ACM). This is the author created version of a work that has been peer reviewed and accepted for publication by ACM Transactions on Graphics, Association for Computing Machinery (ACM). It incorporates referee’s comments but changes resulting from the publishing process, such as copyediting, structural formatting, may not be reflected in this document. The published version is available at: [<a href="http://dx.doi.org/10.1145/2980179.2982424">http://dx.doi.org/10.1145/2980179.2982424</a>].</td>
</tr>
</tbody>
</table>
Manifold Differential Evolution (MDE): A Global Optimization Method for Geodesic Centroidal Voronoi Tessellations on Meshes

Yong-Jin Liu\textsuperscript{1}  Chun-Xu Xu\textsuperscript{1}  Ran Yi\textsuperscript{1}  Dian Fan\textsuperscript{1}  Ying He\textsuperscript{2}
\textsuperscript{1} Tsinghua University  \textsuperscript{2} Nanyang Technological University

Figure 1: Given an initial Voronoi diagram with 1000 generators clustered at Bunny’s left ear, the intrinsic CVT (iCVT) algorithm iteratively improves the quality of the diagram by moving the generator of each Voronoi cell to its Riemannian center. After 1853 iterations, the algorithm gets stuck in a local optimal solution with energy $F = 4.218E-3$. Although for each Voronoi cell, the generator coincides with the mass center; one can clearly see that the Voronoi cells are not uniform. As an evolution-driven optimization method, manifold DE is insensitive to initialization, and is able to compute the global minimizer (in a probability close to 1) of the CVT energy $F = 3.692E-3$. Both visual checking and the cell energy histogram show that our result is more regular than that of the iCVT, since all Voronoi cells are of similar sizes and shapes. See also the accompanying video.

Abstract

Computing centroidal Voronoi tessellations (CVT) has many applications in computer graphics. The existing methods, such as the Lloyd algorithm and the quasi-Newton solver, are efficient and easy to implement; however, they compute only the local optimal solutions due to the highly non-linear nature of the CVT energy. This paper presents a novel method, called manifold differential evolution (MDE), for computing globally optimal geodesic CVT energy on triangle meshes. Formulating the mutation operator using discrete geodesics, MDE naturally extends the powerful differential evolution framework from Euclidean spaces to manifold domains. Under mild assumptions, we show that MDE has a provably probabilistic convergence to the global optimum. Experiments on a wide range of 3D models show that MDE consistently outperforms the existing methods by producing results with lower energy. Thanks to its intrinsic and global nature, MDE is insensitive to initialization and mesh tessellation. Moreover, it is able to handle multiply-connected Voronoi cells, which are challenging to the existing geodesic CVT methods.

Keywords: centroidal Voronoi tessellation, geodesic Voronoi diagram, differential evolution, global optimization, discrete geodesic

Concepts: Computing methodologies $\rightarrow$ Mesh geometry models; Shape analysis;

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org. © 2016 ACM.

SA ‘16 Technical Papers., December 05-08, 2016., Macao
ISBN: 978-1-4503-4514-9/16/12
DOI: http://dx.doi.org/10.1145/2980179.2982424

1 Introduction

A centroidal Voronoi tessellation (CVT) [Du et al. 1999] is a special Voronoi diagram of a given set such that the associated generating points are centers of mass (with respect to a given density function) of the corresponding Voronoi regions. As a powerful computational tool, CVT in Euclidean spaces has been extensively studied in the last decade (e.g., [Liu et al. 2009; Lévy and Liu 2010; Leung et al. 2015; Yan and Wonka 2016]). However, little progress was reported to geodesic CVT (GCVT) — the manifold counterpart — due to the fundamental differences of geometry and topology between Euclidean spaces and manifolds.

It is well known that Voronoi cells in $\mathbb{R}^n$ are simply connected, convex regions, whereas Voronoi cells on surfaces may have non-trivial topologies (see the right inset). In Euclidean spaces, the center of mass $c$ of a Voronoi cell $\Omega$ is defined as the integral of the weighted position coordinates of the points in $\Omega$, i.e., $c = \frac{\int_{\Omega} \rho(x) x \, dx}{\int_{\Omega} \rho(x) \, dx}$, where $\rho$ is the density function. Since $\Omega$ is convex, $c$ is guaranteed to be in the cell $\Omega$. Unfortunately, the Euclidean center of mass of a curved patch is not always in it. Du et al. [2003] suggested using the constrained mass center, which is defined by projecting the Euclidean mass center onto the surface. However, such a center is not unique in general. Since the conventional center-based definition cannot be trivially extended to manifolds, GCVT is defined as a minimizer of the GCVT energy (see Section 4.1).

To compute a reasonable approximation of GCVT, most of the existing methods (e.g., [Yan et al. 2009]) repeatedly compute restricted CVT (RCVT), defined as the intersection between a 3D Euclidean CVT and the input surface. Although it is conceptually simple and highly efficient, RCVT often produces poor results on models with rich geometric details and it is not able to eliminate topological ambiguity (e.g., parts that are geometrically close but topologically far away are often taken as one part) due to its ex-
trintrinsic nature. Wang et al. [2015] proposed an intrinsic method to approximate GCVT on triangle meshes. Their idea is to replace each extrinsic Euclidean center of mass by the intrinsic Riemannian center of mass, defined as the weighted average of the corners of a Voronoi cell. Riemannian centers are a good approximation of centers of mass only if Voronoi cells are simply connected and fairly regular. Otherwise, Riemannian centers may not exist or be far from centers of mass. It is also known that the CVT energy is highly non-linear [Lu et al. 2012], hence the commonly used methods, such as Lloyd’s algorithm [Du et al. 1999] and the quasi-Newton solver [Liu et al. 2009], are inevitably stuck to local optima.

This paper aims at computing the globally optimized GCVT on arbitrary manifold triangle meshes. Towards this goal, we develop a differential evolution (DE) based method to minimize the CVT energy. The reason that we adopt DE is that DE is designed for global optimization over continuous spaces and it performs very well in many real-world applications [Das and Suganthan 2011]. However, all the existing DE solvers [Das et al. 2016] work only for Euclidean spaces, since the mutation operator — a critical operation to increase the diversity of the population, thereby increasing the probability of obtaining the global optimal solution — is heavily built upon a global coordinate system, which is not available to manifolds with non-trivial topology. To tackle this challenge, we elegantly formulate the mutation operator as the initial value and boundary value problems of discrete geodesics, and solve them efficiently using exact discrete geodesic algorithms. Thanks to its intrinsic and global nature, our method is insensitive to initialization and it is able to handle multiply-connected Voronoi cells, which are challenging to all existing geodesic CVT methods.

2 Related Work

2.1 Centroidal Voronoi Tessellation

Let $S = \{ s_i \}_{i=1}^m$ be a set of distinct sites in a connected compact region $\Omega \subset \mathbb{R}^2$. The Voronoi region $\Omega_i$ of $s_i$ is

$$\Omega_i = \{ x \in \Omega \mid ||x - s_i|| \leq ||x - s_j||, \forall i \neq j \},$$

where $|| \cdot ||$ is the Euclidean norm. Let the domain $\Omega$ be endowed with a $C^2$-continuous density function $\rho(x) > 0$. Du et al. [1999] defined an energy functional on $\Omega$ with respect to the Voronoi tessellation

$$F(S) = \sum_{i=1}^m \int_{\Omega_i} \rho(x)||x - s_i||^2 d\sigma \triangleq \sum_{i=1}^m F_i,$$

where the term $F_i$ expresses the compactness (or inertia momentum) of the Voronoi cell $\Omega_i$ associated with the generator $s_i$. The Voronoi tessellation $\bigcup_i \Omega_i$ is called a centroidal Voronoi tessellation if each site $s_i$ coincides with the centroid $c_i$ of its Voronoi cell,

$$s_i = c_i \left( \frac{\int_{\Omega_i} \rho(x) x d\sigma}{\int_{\Omega_i} \rho(x) d\sigma} \right),$$

which is a necessary condition of the minimizer of $F(S)$.

Lloyd’s algorithm [Lloyd 1982] iteratively moves the generator of a Voronoi cell to its mass center. Although it is easy to implement, Lloyd’s algorithm has only a linear convergence rate. Liu et al. [2009] proved that the CVT energy function is almost $C^2$ continuous everywhere, hence one can minimize the CVT energy using the Newton or quasi-Newton method with better convergence rate.

Although it is fairly simple to construct Voronoi diagrams in Euclidean spaces (especially in $\mathbb{R}^2$ and $\mathbb{R}^3$), computing Voronoi diagrams on curved surfaces is technically challenging [Liu et al. 2011; Liu 2015]. Some researchers [Alliez et al. 2003; Alliez et al. 2005; Rong et al. 2011a; Rong et al. 2011b] suggested using global parameterization so that Voronoi diagrams can be computed by scaling the Voronoi cells on the parametric domain. However, global parameterizations are computationally expensive and may suffer from serious numerical issues for models with complicated geometry and/or topology.

Rather than directly constructing the CVT on the input surface, Yan et al. [2009] proposed a different approach that repeatedly computes restricted Voronoi diagrams (RVD), defined as the intersection between the input mesh and a Voronoi diagram in $\mathbb{R}^d$. The RVD method is highly efficient and also flexible for computing CVTs with non-constant density functions. However, it may produce Voronoi regions that are disconnected. Such an issue can be fixed by localized RVD [Yan et al. 2014]. As extrinsic methods, RVD and its variants cannot deal with topology ambiguity.

Based on the Monte Carlo with minimization (MCM) framework, Lu et al. [2012] developed a global optimization method to compute Euclidean CVT. Their method transforms a continuous global optimization problem into a discrete one by confining the searching space only at those local minimizers of the objective function. In each iteration, MCM applies a Monte Carlo sampling that could jump out of the current local minimizer. Then it applies a local solver (e.g., the L-BFGS method [Liu et al. 2009]) to obtain a new local minimizer. The algorithm repeats this iterative procedure until the result cannot be improved. Although their method works well in Euclidean spaces, it lacks a theoretical guarantee of convergence. Moreover, it requires a local solver to compute local optimal solutions. To extend MCM to manifold domains, one has to borrow an intrinsic CVT method (e.g., [Wang et al. 2015]), which, however, is not able to handle multiply-connected Voronoi cells as mentioned above.

2.2 Global Optimization

There are two classes of global optimization methods, namely, deterministic methods and stochastic methods. Deterministic methods are suitable for functions that are convex/concave, or can be expressed as differences of convex functions or as a networking problem [Horst et al. 2000]. These methods also assume smoothness of the objective function and the availability of its derivatives. Unfortunately, the CVT energy, even in Euclidean spaces, is highly nonlinear. In curved spaces where geodesic distances are involved, it is not likely possible to obtain the derivatives. Hence, deterministic methods do not work for our application scenario.

In past few decades, stochastic methods have been extensively studied and applied to solve global optimization problems. Classic stochastic methods include Monte Carlo algorithms, simulated annealing, ant colony optimization, particle swarm optimization and evolutionary algorithms. Among them, evolutionary algorithms are popular due to their ability to generate an improved solution from a set of correlated sub-optimal solutions.

Examples of evolutionary algorithms include genetic algorithms, evolutionary programming, evolution strategies, genetic programming, and differential evolution [Boender and Romeijn 1995; Engelbrecht 2006]. Each method has its own merits and limitations. Genetic algorithms and simulated annealing are designed for combinational optimizations, i.e., the search space is discrete, and evolutionary algorithms can minimize continuous functions with or without derivatives. Differential evolution (DE), which is a competitive form of evolutionary computing, has proven to be a powerful tool for solving real-valued functions, due to its simplicity, robustness and better convergence rate compared with other evolutionary algorithms [Vesterstrom and Thomsen 2004; Qu et al. 2016].
3 Preliminaries on Differential Evolution

Consider a real-valued objective function \( f(X) : \Omega \subseteq \mathbb{R}^D \rightarrow \mathbb{R} \), whose search domain \( \Omega \) is non-empty and bounded in \( \mathbb{R}^D \). Denote \( X = (x_1, x_2, \cdots, x_D) \) the variable, where \( x_i \in \mathbb{R} \).

As a population-based real parameter optimization algorithm, differential evolution (DE) starts from a randomly chosen population of \( N_p \) \( D \)-dimensional vectors that sample the objective function, then iteratively searches for a global optimal solution in \( \Omega \). In each iteration, DE performs a sequence of operations, including mutation, crossover and selection (Figure 2). The algorithm terminates when the user-specified stopping criteria are met.

Let us denote the \( k \)-th population by \( X_k = \{X_{k,i}\}_{j=1}^{N_p} \), where \( X_{k,i} = (x_{k,j,1}, x_{k,j,2}, \cdots, x_{k,j,D}) \) is the \( j \)-th vector. Each vector \( X_{k,i} \) is called an agent.

**Initialization.** DE begins with a randomly initialized population \( X_0 \), where the \( N_p \) agents \( X_{k,j} \) are uniformly randomized in the domain \( \Omega \). When the lower bound \( X_{min} = (x_{1, min}, \cdots, x_{D, min}) \) and upper bound \( X_{max} = (x_{1, max}, \cdots, x_{D, max}) \) of \( \Omega \) are available, one can simply generate the initial population \( X_0 \) as

\[
x_{0,i,j} = x_{i, min} + rand \cdot [0, 1] \cdot (x_{i, max} - x_{i, min}), \quad i = 1, 2, \cdots, D, \quad j = 1, 2, \cdots, N_p
\]

where \( rand \) is a random number in \([0, 1]\) and is instantiated independently for each \( j \).

**Mutation.** Mutation in biology is a sudden change in the gene characteristics of a chromosome. In the DE setting, mutation perturbs the variable vectors with the scaled difference of two randomly selected agents. Let \( k \) denote the current iteration count. For each agent \( X_{k,j} = (x_{k,j,1}, \cdots, x_{k,j,D}) \) and \( X_{k,rand1}, X_{k,rand2} \) and \( X_{k,rand3} \) are sampled randomly from \( X_k \), where the indices \( rand1, rand2 \) and \( rand3 \) are distinct and they are not equal to \( j \).

We compute the mutant agent by adding to the third agent a scaled difference between the first two agents

\[
V_{k,j} = X_{k,rand1} + \lambda (X_{k,rand2} - X_{k,rand3})
\]

where the scalar \( \lambda \) is usually set \( \lambda \in [0.4, 1] \).

**Crossover.** To enhance the diversity of the population, a binomial crossover builds a competitor \( U_{k,j} = \{ u_{k,j,i} \}_{i=1}^{D} \) by copying elements from the original and the mutant agents:

\[
u_{k,j,i} = \begin{cases} v_{k,j,i} & \text{if } (rand_{k,j,i} \leq C_r \text{ or } i = i_{rand}) \\ x_{k,j,i} & \text{otherwise} \end{cases}
\]

where \( C_r \in [0, 1] \) is the crossover rate and \( i_{rand} \) is a random index ranging in \([1, D]\) to ensure that \( U_{k,j} \) gets at least one element from \( V_{k,j} \).

**Selection.** For each agent \( X_{k,j} \) and its competitor \( U_{k,j} \) in the \( k \)-th generation, the selection operation chooses a survivor \( X_{k+1,j} \) for the next generation:

\[
X_{k+1,j} = \begin{cases} U_{k,j} & \text{if } f(U_{k,j}) \leq f(X_{k,j}) \\ X_{k,j} & \text{otherwise} \end{cases}
\]

Note that the population size is a constant over generations.

**Termination conditions.** DE terminates when one of the following conditions is met: (1) the iteration count exceeds a user-specified threshold; (2) the current solution does not improve; and (3) a prescribed objective function value is obtained.

It is worth noting that there are many variants of DE, which differ in the way of crossover. The aforementioned DE is the canonical DE/rand/1/bin algorithm. For a \( C^2 \) continuous objective function possessing a single global optimum (regardless of the number of local optima), Ghosh et al. [2012] proved that DE/rand/1/bin decreases the objective function monotonically and the convergence is guaranteed. In practice, DE/rand/1/bin also works very well for functions with multiple global optima by finding one of them [Price et al. 2005]. We refer readers to the two recent surveys [Das and Suganthan 2011; Das et al. 2016] of DE and its popular variants.

In the evolution strategies theory, the explorative power of an evolution algorithm is determined by its population diversity. Zähare [2001] showed that the expected population variance in DE/rand/1/bin is greater than those of other evolution algorithms, justifying the good performance of the DE algorithms.

4 Overview

4.1 Problem Statement

Let \( M \) be a 2-manifold mesh. Consider a set \( G = \{ g_i \}_{i=1}^m \) of generators on \( M \). For generator \( g_i \), the geodesic Voronoi cell \( C(g_i) \) is defined as

\[
C(g_i) = \{ x \in M : d(x, g_i) \leq d(x, g_j), \forall j \neq i \}
\]

where \( d(p, q) \) is the geodesic distance between \( p \) and \( q \) on \( M \). The geodesic Voronoi tessellation \( T(G) \) of \( G \) on \( M \) is the union of all geodesic Voronoi cells \( T(G) = \bigcup_{i=1}^m C(g_i) \). We then define the GCVT energy as

\[
F(G) = \sum_{i=1}^m \int_{x \in C(g_i)} d^2(x, g_i) dx
\]
We propose two techniques to tackle the above-mentioned challenges due to the following reasons.

### 4.2 Technical Challenges

Since the CVT energy function is highly nonlinear, so is the GCVT energy. Therefore, the conventional local minimization methods, such as Lloyd’s algorithm and the quasi-Newton solver, inevitably get stuck to local optimal solutions. As mentioned above, differential evolution is a powerful computational framework for global optimization of real-valued continuous functions. However, it is highly non-trivial to apply the classic DE/rand/1/bin algorithm to the GCVT problem due to the following reasons.

- For a conventional real-valued function, the order of the variables \((x_1, x_2, \cdots, x_D)\) does matter, since swapping \(x_i\) and \(x_j\) usually produces a new function. However, the variables in the CVT energy are orderless, due to the combinatorial nature of the Voronoi diagram, i.e., the area integral is performed separately on each Voronoi cell. The orderless property brings a serious issue to differential evolution. Given two agents with exactly the same set of generators but in different order, DE may consider them as two distinct agents. However, they produce the same Voronoi diagram, hence the same energy.

- Among the three major operations in a DE iteration, mutation is heavily built upon a global coordinate system, which is only available to Euclidean domains. Moreover, due to the lack of a closed-form of geodesic distances, evaluating the integral is technically challenging.

To our knowledge, all the existing DE algorithms work only for domains in Euclidean spaces.

### 4.3 Key Ideas

We propose two techniques to tackle the above-mentioned challenges.

- To assign a reasonable order to the generators in an agent, we propose an agent matching operator.
- To eliminate the use of global coordinate system, we reformulate the mutation operator using discrete geodesics. Specifically, the difference between two generator sets is to solve the boundary value problem of geodesic, i.e., finding the shortest path between two generators, each of which belongs to a generator set. Adding the difference to the third generator set is equivalent to the initial value problem of geodesic, which computes the unique geodesic, given an initial point and velocity.

We call our method manifold differential evolution (MDE) due to its intrinsic and coordinate-free nature.

### 5 Manifold Differential Evolution

This section presents the algorithmic details of MDE and Appendix proves its probabilistic convergence under some assumptions. We outline the pseudocode of MDE in Algorithm 1 and illustrate a typical iteration in Figure 3. In MDE, an agent in a population \(X_k\) is denoted as \(G_{k,j}\), i.e., \(X_k = \{G_{k,j}\}_{j=1}^{N_k}\).

#### 5.1 Generator Operations

Recall that each element in an agent \(G_{k,j} = \{g_{k,j}\}_{i=1}^{D}\) is a generator, i.e., a point on the mesh \(M\). Let us denote by \(\gamma(x, y)\) the geodesic path between two points \(x, y \in M\). We define the generator subtraction (or GS) operator \(\ominus (y, x) : M \times M \rightarrow T_x(M)\), which takes two points \(x\) and \(y\) as input and produces a tangent vector at \(x\), whose magnitude is the length of \(\gamma(x, y)\) and whose direction is the starting tangent direction of \(\gamma(x, y)\) (Figure 4(a)).

We then define the generator addition (or GA) operator \(\oplus : T(M) \times M \rightarrow M\), which takes a tangent vector \(\bar{v}\) at \(x\) and a point \(y\) as input and produces a point \(z\), which is computed as follows (Figure 4(b)):

- Step 1: parallel transport the tangent vector \(\bar{v}\) from \(x\) to \(y\) along the geodesic \(\gamma(x, y)\); the transported tangent vector is denoted by \(\bar{v}'\).
- Step 2: compute a geodesic path \(\gamma'\) using the initial point \(y\) and the initial tangent direction \(\bar{v}'\); the length of \(\gamma'\) equals the length of \(\bar{v}'\).
Algorithm 1: Manifold DE for minimizing the GCVT energy

**Input:** A 2-manifold triangle mesh $M$, the number $m$ of generators in an agent and the population size $N_p$.

**Output:** The generators that minimize the GCVT energy function.

1. Initialize the first population $X_0$ of $N_p$ agents and set $i = 0$.
2. While the termination criterion is not met:
   3. $G_{i+1} = \emptyset$.
   4. For each agent $G_{i,j} \in X_i$ do:
      5. Generate a mutative agent $G'_{i,j}$.
      6. Generate a trial agent $G''_{i,j}$.
      7. If $F(G''_{i,j}) < F(G_{i,j})$ then:
         8. Add the survivor $G''_{i,j}$ to $G_{i+1}$.
      9. Else:
         10. Reject the trial $G''_{i,j}$.
         11. For each generator $g_{k,j}, k \in G_{i,j}$ do:
            12. If the Voronoi cell is simply connected then:
               13. Move $g_{k,j}$ to its Riemannian center.
            14. End if.
      15. End for.
      16. Add $G_{i,j}$ to $X_{i+1}$.
   17. End if.
   18. End for.
   19. $i++$.
   20. End while.

- Step 3: return the endpoint of $\gamma'$.

Let $\vec{v}_x$ and $\vec{v}_y$ be the unit tangent vectors of $\gamma(x, y)$ at the endpoints $x$ and $y$, respectively. As shown in [Mitchell et al. 1987], a discrete geodesic path $\gamma$ is an alternating sequence of vertices and (possibly empty) edge sequences such that the unfolded image of the path along any edge sequence is a straight line segment and the angle of $\gamma$ passing through a vertex is greater than or equal to $\pi$. Then the parallel transport in Step 1 is simply implemented by finding a vector $\vec{v}' \in T_y(M)$ that preserves the angle $\vec{v}' \cdot \vec{v} = \vec{v}' \cdot \vec{v}$. Let $a, b$ and $c$ be distinct points on $M$ and $\lambda \in \mathbb{R}$ a non-zero scalar. The generator operators $\oplus$ and $\ominus$ have the following properties:

- Identity: $(a \oplus b) \ominus b = a$.
- The triangle rule property: $(c \ominus (a \oplus b)) \ominus (b \ominus a)$.

The triangle rule property shows that with GA operator, the tangent vector $c \ominus a$ at $a$ behaves like a Euclidean vector $\vec{a}c$ and $\vec{a}c = \frac{a + b + c}{3}$. Figure 4 illustrates the GS and GA operators on a curved surface. When $M \subseteq \mathbb{R}^2$ is a planar mesh, $\oplus$ and $\ominus$ are simply vector addition and subtraction.

Figure 4: Generator operations. (a) GS takes two points as input and returns a tangent vector at the second point. $y \oplus x$ is a tangent vector $\vec{v} \in T_x(M)$. (b) GA takes a tangent vector $\vec{v} \in T_y(M)$ at $x$ and a point $y \in M$ as input, and returns a point $z \in M$ on the surface. We parallel transport $\vec{v}$ to point $y$ along geodesic $\gamma(x, y)$, and then compute a geodesic $\gamma'$ using the point $y$ and the transported tangent direction. The result $z$ is a point on $\gamma'$ such that $d(y, z) = \|\vec{v}\|$.

Figure 5: Agent matching. (a) Consider two agents $G = \{g_1, g_2, g_3, g_4\}$ and $G' = \{g_1', g_2', g_3', g_4'\}$, the initial matching $(g_i, g_i')$ has a high cost, due to the long geodesic distances between the paired generators. (b) Solving the minimum-weight perfect matching problem, we obtain a permutation $\sigma = (4, 1, 2, 3)$, leading to the matching with minimum cost.

5.2 Agent Operations

We define the agent matching (or AM) operator to align two agents $G = \{g_i\}_{i=1}^m$ and $G' = \{g'_i\}_{i=1}^m$. We build a complete bipartite graph $K_{m,m}$ with vertex set $V(K_{m,m}) = G \cup G'$. The weight for an edge $e(g_i, g'_j)$, $\forall g_i \in G$ and $\forall g'_j \in G'$, is the geodesic distance between $g_i$ and $g'_j$ on $M$. We assign orders to the generators in agents $G$ and $G'$ by finding a perfect matching in $K_{m,m}$ with the minimum sum of edge costs. We solve the minimum-weight perfect matching problem using the Hungarian algorithm [Korte and Vygen 2006] in $O(m^3)$ time (Figure 5). The perfect matching is denoted by a permutation $\sigma$ such that $o(\sigma(i)) \in G$ is mapped to $o'(\sigma(i)) \in G'$, $i = 1, 2, \ldots, m$.

By applying the GS operator to the matched agents $G$ and $G'$, we define agent subtraction (AS) as:

$$G \ominus G' = \{g_{\sigma(i)} \ominus g'_i\}_{i=1}^m$$

Let $D$ denote the subtraction of two agents, say $G'$ and $G''$. Then for an arbitrary third agent $G'''$, we define the agent addition (AA) operator as:

$$D \oplus G''' = \{d_i \oplus g'''_i\}_{i=1}^m$$

We abuse the notations by using $\ominus$ and $\oplus$ for both agent and generator subtraction/addition.

Armed with the AM, AS and AA operators, we can formulate the manifold mutation operator in a similar way as the Euclidean setting except that vector difference and vector addition are replaced by agent subtraction and agent addition.

5.3 Algorithm

MDE starts by sampling the objective function (Eq. (8)) at an initial population of $N_p$ agents (each agent is a generator set). Denote the population at the $k$th generation as $X_k = \{G_{k,j}\}_{j=1}^{N_p}$, where $G_{k,j} = \{g_{k,j,1}\}_{i=1}^m$ is the $j$th agent and $g_{k,j,1}$ is the $i$th generator in $G_{k,j}$.

We outline the MDE algorithm in the pseudocode of Algorithm 1. We present the details of initialization, mutation, crossover, selection and termination conditions, in this subsection. We prove the probabilistic convergence in Appendix.

Initialization. For each agent $G_{0,j}$ of the first generation $X_0$, we randomly select $m$ vertices as its generators.

Mutation. For each agent $G_{k,j}$ in the current generation $X_k$, three other agents $G_{k,rand1}, G_{k,rand2}$ and $G_{k,rand3}$ are randomly sampled from $X_k$, where $rand1, rand2$ and $rand3$ are three distinct
### 6 Experimental Results

#### Implementation

We implemented the FWP-MMP algorithm [Xu et al. 2015] to compute the exact discrete geodesic paths and measure geodesic distances on triangle meshes, Cheng et al. [2016]'s method to solve the initial value problem, and the adaptive cubature algorithm [Bernshten and Espelid 1992] to evaluate the area integral on triangle meshes. For each triangle, we used a symmetric quadratic rule of polynomial degree 5 with 7 points [Wandzurat and Xiao 2003]. Computational results show that this simple quadrature produces high-quality results, which are almost identical to those computed by a highly-accurate-but-more-expensive quadrature of polynomial degree 30 with 175 points. The timings were measured on a PC with Intel(R) Core(TM) i7-2600 3.40GHz CPU and 8GB RAM.

#### Time complexity

Note that the agent matching operation and computing the exact geodesic between two points take $O(m^3)$ and $O(n^2 \log n)$ time, where $m$ and $n$ are numbers of generators and mesh vertices, respectively. Therefore, MDE runs in $O(N_p K (m^3 + mn^2 \log n))$ time, where $N_p$ is population size and $K$ is iteration number.

#### Parameter setting

There are five parameters in Algorithm 1, i.e., the population size $N_p$, the mutation ratio $\lambda$ in the mutation equation (11), the crossover rate $C_r$ in equation (12), the termination threshold $\tau$, and the maximal iteration count $n_t$. Since the variables of our objective function are highly dependent, we choose a high crossover rate $C_r = 0.9$ to encourage population diversity and set $\lambda = 0.8$, which is proportional to $C_r$. In addition, we set $N_p = 200$, $\tau = 1\%$, and $n_t = 5$ in termination condition. It is worth noting that MDE is insensitive to parameters and does not require fine tuning: through extensive evaluation, we observe that the GCVT energies are highly consistent when $C_r \in [0.6, 1.0]$ and $N_p \in [200, 300]$. See two examples in Table 1. We also observe that the running time increases significantly when $N_p$ becomes larger.

#### Comparisons on 2D Euclidean domains

The MCM algorithm [Lu et al. 2012] is a global optimization method for CVT in Euclidean spaces, which adopt the L-BFGS method [Liu et al. 2009] as the local solver. Note that in $\mathbb{R}^2$, the agent operations in MDE are the conventional vector operations and geodesic distances become Euclidean distances. As shown in Table 2, MDE consistently outperforms MCM in terms of runtime performance and CVT energies. The cell energy histogram also shows that the MDE results are the conventional vector operations and geodesic distances between two points take $O(m^3)$ and $O(n^2 \log n)$ time, where $m$ and $n$ are numbers of generators and mesh vertices, respectively. Therefore, MDE runs in $O(N_p K (m^3 + mn^2 \log n))$ time, where $N_p$ is population size and $K$ is iteration number.

#### Comparisons on 3D meshes

To evaluate our algorithm, we compare to two state-of-the-art methods, the iCVT algorithm [Wang et al. 2015] and the MCM algorithm [Lu et al. 2012]. Note that the original MCM algorithm was designed for computing Euclidean CTVs. We extend it to manifold domains by using iCVT as its local solver. Although MCM is a global optimization method, it lacks a theoretical guarantee of finding the optimal solution. As Table 2 reports, MDE consistently outperforms MCM and iCVT in terms of energy. The small standard deviation of Voronoi cell areas

### Table 1: The GCVT energy and running time (hours in brackets) of MDE under different parameter settings.

<table>
<thead>
<tr>
<th>Models in $\mathbb{R}^2$</th>
<th>Number of generators</th>
<th>Mean (standard deviation) of the CVT energy per Voronoi cell</th>
<th>Time (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pentagon</td>
<td>500</td>
<td>$3.788E-6 (3.1E-7)$</td>
<td>0.62</td>
</tr>
<tr>
<td>Hexagon</td>
<td>500</td>
<td>$4.481E-6 (3.9E-7)$</td>
<td>0.62</td>
</tr>
<tr>
<td>Heptagon</td>
<td>500</td>
<td>$4.981E-6 (4.2E-7)$</td>
<td>0.62</td>
</tr>
<tr>
<td>Decagon</td>
<td>500</td>
<td>$5.765E-6 (4.1E-7)$</td>
<td>0.62</td>
</tr>
<tr>
<td>Dumbbell</td>
<td>500</td>
<td>$6.478E-8 (1.6E-6)$</td>
<td>0.62</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Models in $\mathbb{R}^3$</th>
<th>Number of vertices</th>
<th>Mean (standard deviation) of the GCVT energy per Voronoi cell</th>
<th>Time (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Armadillo</td>
<td>172,974</td>
<td>$7.489E-6 (6.7E-6)$</td>
<td>0.62</td>
</tr>
<tr>
<td>Knot</td>
<td>80,000</td>
<td>$1.509E-5 (3.5E-5)$</td>
<td>0.62</td>
</tr>
<tr>
<td>Bunny</td>
<td>72,020</td>
<td>$1.720E-5 (1.1E-5)$</td>
<td>0.62</td>
</tr>
<tr>
<td>Decoube</td>
<td>60,198</td>
<td>$7.344E-5 (6.5E-5)$</td>
<td>0.62</td>
</tr>
<tr>
<td>Sphere</td>
<td>32,254</td>
<td>$2.592E-4 (4.8E-6)$</td>
<td>0.62</td>
</tr>
<tr>
<td>Fertility</td>
<td>29,394</td>
<td>$6.976E-6 (3.1E-6)$</td>
<td>0.62</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fertility model</th>
<th></th>
<th>MDE</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_r = 0.6$, $C_r = 0.7$</td>
<td></td>
<td>$C_r = 0.8$, $C_r = 1.0$</td>
<td></td>
</tr>
<tr>
<td>2006.609E-3 (5.48)</td>
<td>$7.219E-2$</td>
<td>$7.219E-2$</td>
<td>0.61</td>
</tr>
<tr>
<td>2506.601E-3 (7.01)</td>
<td>$7.219E-2$</td>
<td>$7.219E-2$</td>
<td>0.61</td>
</tr>
<tr>
<td>3006.601E-3 (9.71)</td>
<td>$7.219E-2$</td>
<td>$7.219E-2$</td>
<td>0.61</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decocube model</th>
<th></th>
<th>MDE</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_r = 0.6$, $C_r = 0.7$</td>
<td></td>
<td>$C_r = 0.8$, $C_r = 1.0$</td>
<td></td>
</tr>
<tr>
<td>2006.609E-3 (5.52)</td>
<td>$7.219E-2$</td>
<td>$7.219E-2$</td>
<td>0.61</td>
</tr>
<tr>
<td>2506.601E-3 (7.01)</td>
<td>$7.219E-2$</td>
<td>$7.219E-2$</td>
<td>0.61</td>
</tr>
<tr>
<td>3006.601E-3 (9.71)</td>
<td>$7.219E-2$</td>
<td>$7.219E-2$</td>
<td>0.61</td>
</tr>
</tbody>
</table>
justifies that our results are more regular than those of MCM and iCVT. See the statistics in Table 3 and visual results in Figure 7. Moreover, in most cases, we observe that MDE runs significantly faster than MCM. We also observe that iCVT often fails on models with tubular parts, since the Voronoi cells are not simply connected especially when there are only a few generators (see Figure 8).

**Robustness.** We tested MDE on meshes with both regular and irregular triangulations and measured the GCVT quality by its dual Delaunay triangulation. Following [Yan et al. 2009], we define the anisotropy $Q(t)$ of a triangle $t$ as $Q(t) = \frac{S}{\sqrt{p \cdot h^3}}$, where $S$ is the area of $t$, $p$ is the radius of the inscribed circle of $t$ and $h$ is the longest side of $t$. Let $Q_{\text{ave}}(M)$ and $Q_{\text{min}}(M)$ denote the average and minimum $Q(t)$ of the input mesh $M$. $Q_{\text{ave}} = 1$ for completely regular triangulations and $Q_{\text{ave}} \ll 1$ for meshes with highly irregular triangulations. We also added Gaussian noise by perturbing each vertex’s position along its normal direction. The perturbation is a random number in $[-2\%, 2\%]$ multiplying the diagonal length of the mesh’s bounding box. As Table 4 and Figure 10 show, MDE is robust to noise and insensitive to mesh triangulation.

**Discussion.** In our current implementation, we adopt the MMP algorithm [Mitchell et al. 1987; Xu et al. 2015] for computing discrete geodesics, which play a critical role in agent addition and subtraction. As an exact algorithm, it is able to measure geodesic distances between any pair of points (not necessarily mesh vertices), and works for meshes of arbitrary triangulation. Such a feature is not available to the existing approximate algorithms (e.g., the fast marching method [Kimmel and Sethian 1998] and the heat method [Crane et al. 2013]), which compute geodesic distances on mesh vertices only. To compute Voronoi edges (i.e., bisectors), these algorithms apply linear interpolation to mesh edges, thereby producing at most 1 bisector across a mesh edge. However, for models with a high degree of anisotropy, it is common that multiple Voronoi edges cross a mesh edge (see Figure 9). We would like to also point out that the price to pay for the accuracy and robustness of MDE is the high computational cost. In the future, we will adopt faster geodesic algorithms, such as the SVG method [Ying et al. 2013] and the parallel Chen-Han algorithm [Ying et al. 2014], to improve the performance of MDE.
Table 4: MDE is robust to noise and insensitive to mesh triangulation. The quality is measured by the Delaunay triangulations dual to the GCVTs shown in Figure 10. \( \theta_{\text{min}} \) (resp. \( \theta_{\text{min,ave}} \)) are the smallest (resp. mean) angle of the minimal angles of all triangles in MCVT.

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of vertices</th>
<th>Number of generators</th>
<th>Type</th>
<th>Mesh quality</th>
<th>Quality of dual Delaunay mesh MCVT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( Q_{\text{min}}(M) )</td>
<td>( Q_{\text{ave}}(M) )</td>
</tr>
<tr>
<td>Kitten</td>
<td>21,920</td>
<td>1,000</td>
<td>Regular</td>
<td>0.147</td>
<td>0.788</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Noisy</td>
<td>0.001</td>
<td>0.380</td>
</tr>
<tr>
<td>Bunny</td>
<td>25,002</td>
<td>1,000</td>
<td>Regular</td>
<td>0.108</td>
<td>0.800</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Noisy</td>
<td>0.017</td>
<td>0.791</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Irregular</td>
<td>0.001</td>
<td>0.359</td>
</tr>
<tr>
<td>Eight</td>
<td>16,174</td>
<td>1,000</td>
<td>Regular</td>
<td>0.648</td>
<td>0.900</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Noisy</td>
<td>0.001</td>
<td>0.412</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Irregular</td>
<td>0.564</td>
<td>0.809</td>
</tr>
</tbody>
</table>

Figure 9: Using exact geodesic algorithm, MDE works well on meshes with high degree of anisotropy.

7 Conclusion

In this paper, we extended the powerful differential evolution (DE) framework to manifold domains and applied it to minimize the GCVT energy. We proposed three manifold-based agent operations, namely agent matching, agent subtraction and agent addition, and then formulated the crossover operator using discrete geodesics. We showed that MDE has a provable probabilistic convergence to the global optimum. Computational results showed that MDE outperforms the existing global CVT methods in terms of both runtime performance and quality. Moreover, MDE is able to handle multiply-connected Voronoi cells, which are challenging to the Riemannian center-based method.

Acknowledgements

This project was partially supported by the National Key Research and Development Plan (2016YFB1001202), Natural Science Foundation of China (61322206, 61432003, 61521002, 61661130156), Royal Society-Newton Advanced Fellowship, TNList Foundation, MOE2013-T2-2-011 and RG23/15.

References


The MDE results are not sensitive to mesh triangulation and tessellation. Images are rendered in high-resolution, allowing for close-up examination. See Table 4 for the statistics.


---

**Figure 10:** The MDE results are not sensitive to mesh triangulation and tessellation. Images are rendered in high-resolution, allowing for close-up examination. See Table 4 for the statistics.
A Probabilistic Convergence

The population at the k-th generation X_k consists of N_p agents \( \{G_{k,j}\}_{j=1}^{N_p} \) in which each \( G_{k,j} \) is regarded as an instantiation of a random variable \( G_{k,j} \).

Lemma 1 In the population at any k-th generation, \( k \geq 1 \), the random variables \( G_{1,k}, G_{2,k}, \ldots, G_{N_p,k} \) are mutually independent.

Proof. First, each agent in the initial population is randomly selected from mesh vertices. So the random variables in the initial population are mutually independent. Secondly, we prove Lemma 1 by induction.

Assume that \( G_{1,k}, G_{2,k}, \ldots, G_{N_p,k} \) are mutually independent.

To create an instantiation of \( G_{k+1,j} \), we only need to know four instantiations \( G_{k+1,j}, G_{k+1,j+1}, G_{k+1,j+2}, G_{k+1,j+3} \) at 4th generation (ref. Eqs. (11-13)). In other words, the creation of \( G_{k+1,j} \) does not need any information from \( G_{k+1,j'}, j' \neq j \).

Therefore, \( G_{k+1,1}, G_{k+1,2}, \ldots, G_{k+1,N_p} \) are mutually independent.

Let \( \sigma \) be a curve passing through a point \( p \) in \( M \) and denote the unit tangent vector of \( \sigma \) at \( p \) as \( \gamma' \). At any point \( p \in M \), a local Frenet frame \( \{t_p, n_p, \tilde{b}_p\} \), where \( t_p \) is a unit tangent vector at \( p, n \) is the unit normal vector at \( p \) and \( \tilde{b}_p \times t_p \times n_p \). The tangent plane to \( M \) at \( p \) is spanned by any two independent tangent vectors at \( p \) and is denoted as \( T_p(M) \).

Generic non-zero vector \( v \in T_p(M) \) has an unique-parameterized geodesic \( \gamma : (-\varepsilon, \varepsilon) \to M, \gamma(0) = p \) with \( \gamma'(0) = v = \exp_p(v) \). The exponential map \( \exp_p(v) \) is \( v = \exp_p(T_p(M) \to M \to p \) maps any vector \( v \in T_p(M) \) to a point \( q \) on \( M \) in two steps: (1) find the unique geodesic \( \gamma \) with \( \gamma(0) = 0 \) and \( \gamma'(0) = v = \exp_p(v) \), and (2) find \( q \) on \( \gamma \) as \( d(p,q) = |v| \).

A local area on \( M \) is called a normal neighborhood \( N_p \) of \( p \) in \( M \), if \( N_p \) is the image \( N_p = \exp_p(U) \) of a neighborhood \( U \subset T_p(M) \) around \( p \) restricted to which \( \exp_p \) is a diffeomorphism. On \( M \) a vertex at which the sum of surrounding angles is larger than \( 2\pi \) is called a saddle vertex.

Lemma 2 For any point \( p \in M \), let \( \gamma_p \) be the maximal radius such that the geodesic disk \( D_p = \{ q \in M : d(p,q) \leq \gamma_p \} \) centered at \( p \) does not contain any saddle vertices and is homeomorphic to a planar disk. \( D_p \) is a normal neighborhood of \( p \).

Proof. Mitchell et al. [1987] showed that except for its two endpoints, a geodesic path on \( M \) can pass one or several mesh vertices if and only if these mesh vertices are saddle. Since \( D_p \) is homeomorphic to a planar disk, for every point \( q \in D_p \), there is a unique geodesic in \( D_p \) connecting \( p \) and \( q \). That completes the proof.

Let \( g_1 \) and \( g_2 \) be two generators on \( M \) and \( d = g_1 \circ g_2 \) as defined in Section 5.1. If \( g_1 \in D_{g_2} \), both points \( g_1 \) and \( g_2 \) can be inversely mapped by \( \exp_{g_2} \) to two unique vectors \( v_{g_1} \) and \( v_{g_2} \) in the tangent space \( T_{g_2}(M) \), in which \( v_{g_2} \) is a zero vector \( \mathbf{0} \). We regard that \( v_{g_1}, v_{g_2} \) as a random vector \( \mathbf{0} \).

Denote the probability density functions of \( V(G) \) and \( D \) as \( f_{V(G)}(V(G)) \) and \( f_D(D) \), respectively. Let \( G'' = D \circ G'' = \{d_1 \circ g_1\}_{i=1}^{m} = \{g''_i\}_{i=1}^{m} \) as defined in Eq. (10). Each point/generator \( g_i \) can be inversely mapped by \( \exp_{g_i} \) as the zero vector \( \mathbf{0} \) in the tangent space \( T_{g_i}(M) \).

Let \( V(G'' \) be a vector in \( \mathbb{R}^{2m} \) by packing \( m \) 2-dimensional zero vectors \( \{v(g''_i)\}_{i=1}^{m} \), which is an instantiation of a random variable \( V(G'') \).

If \( g''_i \in D_{g''_i}, i = 1, 2, \ldots, m, G'' \) can be regarded as an instantiation of a random variable \( V(G'') = D + V(G'') \) defined in \( \mathbb{R}^{2m} \), whose probability density function is \( f_{V(G'')} V(G'') \) (Pa-poulis 1991), where \( * \) is the 2m-dimensional linear convolution operation.

Proof. First, we have \( f_d(D) = f_{V(g_1)}(v(g_1)) \times f_{V(g_2)}(v(g_2)) \). Secondly, \( D = G_{k,j} \circ G_{k,j} = \{g_{k,j} \circ g_{k,j} \}_{i=1}^{m} = \{d_i\}_{i=1}^{m} \) and \( V(G_{k,j}) = V(G_{k,j}) + V(G_{k,j}) \). Since \( G_{k,j} \) and \( G_{k,j} \) are independent, \( v(g_{k,j} \circ g_{k,j}) \) and \( v(g_{k,j} \circ g_{k,j}) \) are independent, \( i = 1, 2, \ldots, m \). Then we have \( f_{V(G)}(V(G)) = f_{V(G_{k,j})} V(G_{k,j}) \times f_{V(G_{k,j})} V(G_{k,j}) \).

Thirdly, since \( G_{k+1,j}, G_{k+1,j} \) and \( G_{k+1,j} \) are mutually independent, \( D_{G_{k+1,j}} \) are independent. Since \( g_{k+1,j} \circ g_{k+1,j} \) is no longer than \( r \). Therefore, \( d_i \circ g_{k,j} \circ g_{k,j} \) satisfies \( g_i \in D_{g_{k,j} \circ g_{k,j}}, i = 1, 2, \ldots, m. \) Then we have \( f_{V(G)}(V(G)) = f_{V(G_{k,j} \circ g_{k,j})} V(G_{k,j} \circ g_{k,j}) \).

That completes the proof.

Theorem 1 [Ghosh et al. 2012] If the objective function \( F(G) \) in Eq. (8) is \( C^2 \) and has a unique global minimum \( G_{opt} \) on \( M \), and if \( g_{k,j} \circ g_{k,j} \in D_{g_{k,j} \circ g_{k,j}}, i, j = 1, 2, \ldots, N_p, \) then \( f_{G_{opt}}(G_{opt}) > 0 \) for all \( k > 0 \) and all \( l = 1, 2, \ldots, N_p \). Furthermore, \( f_{G_{opt}}(G_{opt}) \) converges to \( \delta(D_{opt}) \) as \( k \to \infty \), where \( D_{opt} \) is a 2m-dimensional vector by packing \( m \) two-dimensional vectors in \( \{d_i\}_{i=1}^{m} \). That completes the proof.

Theorem 1 requires that \( g_{k,j} \circ g_{k,j} \in D_{g_{k,j} \circ g_{k,j}}, i, j = 1, 2, \ldots, N_p, \) this condition is not always satisfied in Algorithm 1 but in our experiments, Algorithm 1 works quite well even without explicitly guaranteeing this condition.