Manifold Differential Evolution (MDE): A Global Optimization Method for Geodesic Centroidal Voronoi Tessellations on Meshes

Yong-Jin Liu[1] Chun-Xu Xu[1] Ran Yi[1] Dian Fan[1] Ying He[2]
[1] Tsinghua University [2] Nanyang Technological University

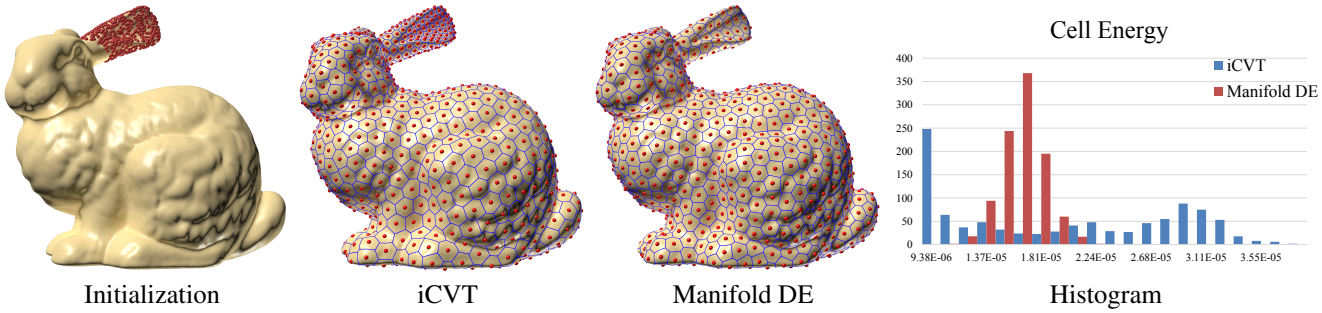


Figure 1: *Given an initial Voronoi diagram with 1000 generators clustered at Bunny's left ear, the intrinsic CVT (iCVT) algorithm iteratively improves the quality of the diagram by moving the generator of each Voronoi cell to its Riemannian center. After 1853 iterations, the algorithm gets stuck in a local optimal solution with energy $F = 4.218E\text{-}3$. 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 $F = 3.692E\text{-}3$. 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 provable 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

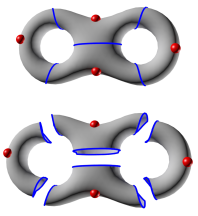
Concepts: •**Computing methodologies** → *Mesh geometry models; Shape analysis;*

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

ACM Reference Format
Liu, Y., Xu, C., Yi, R., Fan, D., He, Y. 2016. Manifold Differential Evolution (MDE): A Global Optimization Method for Geodesic Centroidal Voronoi Tessellations on Meshes. ACM Trans. Graph. 35, 6, Article 243 (November 2016), 10 pages. DOI = 10.1145/2980179.2982424 http://doi.acm.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 Ω is defined as the integral of the weighted position coordinates of the points in Ω, i.e., $c = \frac{\int_{x\in\Omega}\rho(x)x\,dx}{\int_{x\in\Omega}\rho(x)\,dx}$, where ρ is the density function. Since Ω is convex, c is guaranteed to be *in* the cell Ω. 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-

trinsic 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 GCVT 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 Ω_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 Ω be endowed with a C^2-continuous density function $\rho(x) > 0$. Du et al. [1999] defined an energy functional on Ω with respect to the Voronoi tessellation

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

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

$$s_i = c_i \left(= \frac{\int_{\Omega_i} \rho(x) x d\sigma}{\int_{\Omega_i} \rho(x) d\sigma} \right), \tag{2}$$

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 $\mathbb{R}^3$. 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 cannot 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 combinatorial 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].

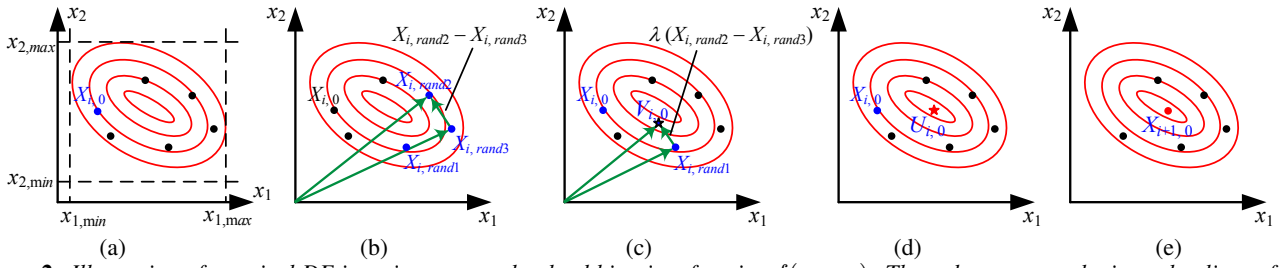


Figure 2: *Illustration of a typical DE iteration on a real-valued bivariate function $f(x_1, x_2)$. The red curves are the iso-value lines of the function f. Each agent is a 2D vector, hence can be visualized as a point. (a) shows six agents (black dots) in the i-th population. (b) To find the competitor for agent $X_{i,0}$, we pick three arbitrary agents from the population X_i, and compute the difference between the second and third agents. (c) Adding the scaled difference vector to the first agent, we obtain the mutative agent $V_{i,0}$. (d) Generate the competitor $U_{i,0}$ by binary crossover of $X_{i,0}$ and $V_{i,0}$. In this 2D case, $U_{0,0} = (x_1(V_{0,0}), x_2(X_{0,0}))$. (e) If $f(U_{i,0}) < f(X_{i,0})$, set the survivor $X_{i+1,0} = U_{i,0}$ for the $(i+1)$-th population, otherwise reject $U_{i,0}$.*

3 Preliminaries on Differential Evolution

Consider a real-valued objective function $f(X) : \Omega \subset \mathbb{R}^D \to \mathbb{R}$, whose search domain Ω 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 Ω. 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 $X_k = \{X_{k,j}\}_{j=1}^{N_p}$, where $X_{k,j} = \{x_{k,j,i}\}_{i=1}^{D}$ is the j-th vector. Each vector $X_{k,j}$ 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 Ω. 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 Ω are available, one can simply generate the initial population X_0 as

$$\begin{array}{l} x_{0,j,i} = x_{i,min} + rand_{j,i}[0,1] \cdot (x_{i,max} - x_{i,min}), \\ i = 1, 2, \cdots, D, \quad j = 1, 2, \cdots, N_p \end{array} \tag{3}$$

where $rand_{j,i}[0,1]$ generates a uniformly distributed random number in $[0,1]$ and is instantiated independently for each (j,i).

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 $X_{k,j}$, $j = 1, \cdots, N_p$, three other agents $X_{k,rand1}$, $X_{k,rand2}$ 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 *mutative* agent by adding to the third agent a scaled difference between the first two agents

$$V_{k,j} = X_{k,rand1} + \lambda(X_{k,rand2} - X_{k,rand3}) \tag{4}$$

where the scalar λ 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 mutative agents:

$$u_{k,j,i} = \begin{cases} v_{k,j,i}, & \text{if } (rand_{k,j,i}[0,1] \le C_r \text{ or } i = i_{rand}) \\ x_{k,j,i}, & \text{otherwise} \end{cases} \tag{5}$$

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:

$$X_{k+1,j} = \begin{cases} U_{k,j}, & \text{if } f(U_{k,j}) \le f(X_{k,j}) \\ X_{k,j}, & \text{otherwise} \end{cases} \tag{6}$$

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. Zaharie [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

$$C(g_i) = \{x \in M : d(x, g_i) \le d(x, g_j), \forall j \ne i\} \tag{7}$$

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 $T(G) = \cup_{i=1}^{m} C(g_i)$. We then define the GCVT energy as

$$F(G) = \sum_{i=1}^{m} \int_{x \in C(g_i)} d^2(x, g_i) dx. \tag{8}$$

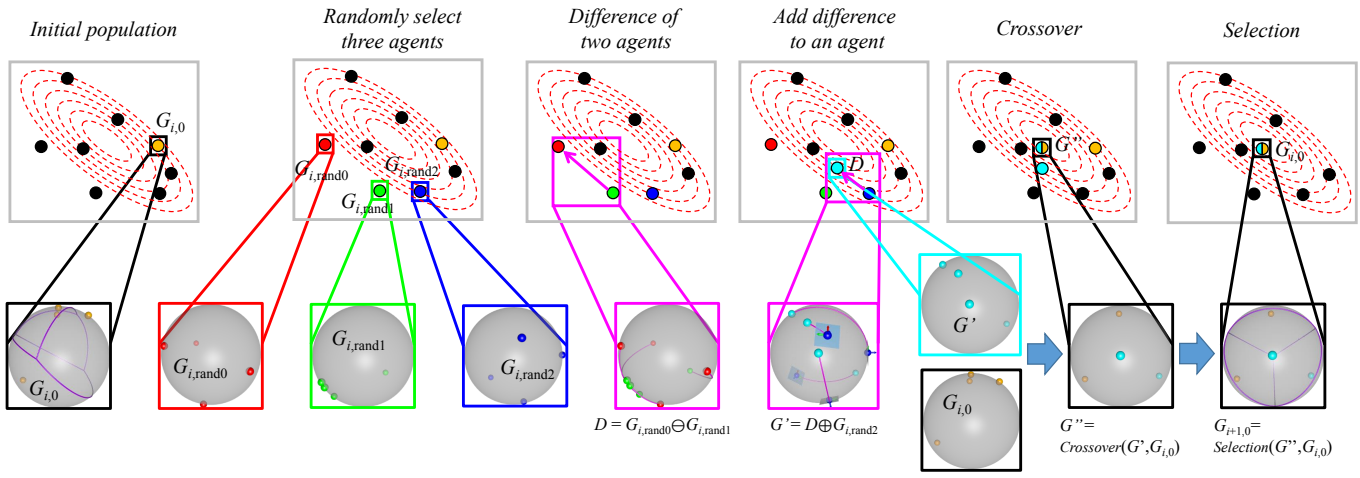


Figure 3: *Illustration of a typical iteration of MDE on a unit sphere. Each agent has four generators.*

Since M is compact, one can show that $F(G)$ is continuous and has at least one global minimum via a simple adaption of Lemma 3.4 in [Du et al. 1999] and Theorem 1.13 in [Horst et al. 2000]. A geodesic Voronoi diagram is a GCVT if it minimizes the above GCVT energy.

In a special case $M \subset \mathbb{R}^2$, $F(G)$ is the Euclidean CVT energy with a constant density $\rho(x) \equiv 1$ in Eqn. (1). Although extending the energy functional from Euclidean spaces to manifolds is trivial, to our knowledge, no one has ever defined it and tackled this problem before.

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., $X_k = \{G_{k,j}\}_{j=1}^{N_p}$.

5.1 Generator Operations

Recall that each element in an agent $G_{k,j} = \{g_{k,j,i}\}_{i=1}^{D}$ 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 $\ominus(y, x) : M \times M \to T_x(M)$, 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 $\oplus : T(M) \times M \to M$, which takes a tangent vector $\vec{v}$ 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 $\vec{v}$ from x to y along the geodesic $\gamma(x, y)$; the transported tangent vector is denoted by $\vec{v}'$;
- Step 2: compute a geodesic path γ' using the initial point y and the initial tangent direction $\vec{v}'$; the length of γ' equals the length of $\vec{v}$.

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

```
Initialize the first population X_0 of N_p agents and set i = 0
while the termination criterion is not met do
    G_{i+1} = ∅
    for each agent G_{i,j} ∈ X_i do
        Generate a mutative agent G'_{i,j}
        Generate a trial agent G''_{i,j}
        if F(G''_{i,j}) < F(G_{i,j}) then
            Add the survivor G''_{i,j} to G_{i+1}
        else
            Reject the trial G''_{i,j}
            for each generator g_{i,j,k} ∈ G_{i,j} do
                if the Voronoi cell is simply connected then
                    Move g_{i,j,k} to its Riemannian center
                end if
            end for
            Add G_{i,j} to X_{i+1}
        end if
    end for
    i++
end while
```

- Step 3: return the endpoint of γ'.

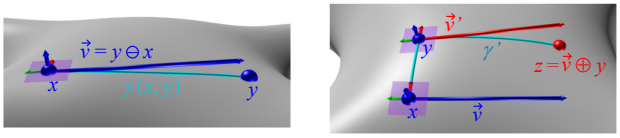
Let $\vec{v}_x$ and $\vec{v}_y$ be the unit tangent vectors of $\gamma(x, y)$ at the endpoints x and y, respectively. As shown in [Mitchell et al. 1987], a discrete geodesic path γ is an alternating sequence of vertices and (possibly empty) edge sequences such that the unfolded image of the path along any edge sequence is a straight line segment and the angle of γ passing through a vertex is greater than or equal to π. Then the parallel transport in Step 1 is simply implemented by finding a vector $\vec{v}' \in T_y(M)$ that preserves the angle $\vec{v}_x \cdot \vec{v} = \vec{v}_y \cdot \vec{v}'$.

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

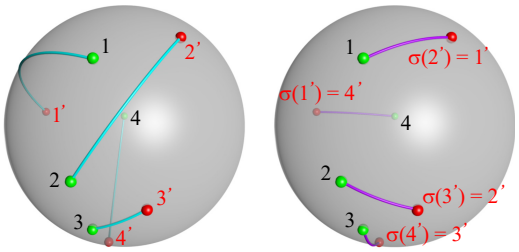
The triangle rule property shows that with GA operator, the tangent vector $c \ominus a$ at a behaves like a Euclidean vector $\overrightarrow{ac}$ and $\overrightarrow{ac} = \overrightarrow{ab} + \overrightarrow{bc}$.

Figure 4 illustrates the GS and GA operators on a curved surface. When $M \subset \mathbb{R}^2$ is a planar mesh, $\oplus$ and $\ominus$ are simply vector addition and subtraction.



(a) Generator subtraction (b) Generator addition

Figure 4: *Generator operations. (a) GS takes two points as input and returns a tangent vector at the second point. $y \ominus x$ is a tangent vector $\vec{v} \in T_x(M)$. (b) GA takes a tangent vector $\vec{v} \in T_x(M)$ 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 γ' using the point y and the transported tangent direction. The result z is a point on γ' such that $d(y, z) = \|\vec{v}\|$.*



(a) (b)

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 σ such that $g_{\sigma(i)} \in G$ is mapped to $g'_i \in G'$, $i = 1, 2, \cdots, m$.

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

$$G \ominus G' = \{g_{\sigma(i)} \ominus g'_i\}_{i=1}^m \tag{9}$$

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

$$D \oplus G''' = \{d_i \oplus g'''_i\}_{i=1}^m \tag{10}$$

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 kth generation as $X_k = \{G_{k,j}\}_{j=1}^{N_p}$, where $G_{k,j} = \{g_{k,j,i}\}_{i=1}^m$ is the jth agent and $g_{k,j,i}$ is the ith 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

N_p	Fertility model					Decocube model				
	$C_r = 0.6$	$C_r = 0.7$	$C_r = 0.8$	$C_r = 0.9$	$C_r = 1.0$	$C_r = 0.6$	$C_r = 0.7$	$C_r = 0.8$	$C_r = 0.9$	$C_r = 1.0$
200	6.690E-3 (5.48)	6.690E-3 (5.38)	6.690E-3 (5.52)	6.690E-3 (5.31)	6.690E-3 (5.44)	7.219E-2 (7.87)	7.219E-2 (7.99)	7.219E-2 (7.98)	7.219E-2 (7.93)	7.219E-2 (7.95)
250	6.691E-3 (7.04)	6.690E-3 (7.01)	6.690E-3 (7.17)	6.690E-3 (6.95)	6.690E-3 (7.07)	7.219E-2 (10.60)	7.218E-2 (10.44)	7.219E-2 (10.70)	7.219E-2 (10.70)	7.219E-2 (10.34)
300	6.691E-3 (9.58)	6.690E-3 (9.71)	6.691E-3 (9.64)	6.691E-3 (9.69)	6.689E-3 (9.61)	7.218E-2 (15.09)	7.219E-2 (15.26)	7.219E-2 (15.25)	7.219E-2 (15.14)	7.219E-2 (15.26)

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

Models in $\mathbb{R}^2$	Number of generators	Mean (standard deviation) of the CVT energy per Voronoi cell					Time (sec.)		
		L-BFGS	MCM	MDE	r_1	r_2	L-BFGS	MCM	MDE
Pentagon	500	3.788E-6 (3.1E-7)	3.678E-6 (3.5E-7)	3.663E-6 (2.7E-7)	3.41%	0.40%	0.61	116.99	39.74
Hexagon	500	4.481E-6 (3.9E-7)	4.384E-6 (3.6E-7)	4.360E-6 (2.1E-7)	2.36%	0.54%	0.70	140.43	45.11
Heptagon	500	4.981E-6 (4.2E-7)	4.860E-6 (4.1E-7)	4.847E-6 (3.3E-7)	2.76%	0.28%	0.90	157.36	70.03
Decagon	500	5.765E-6 (4.9E-7)	5.608E-6 (4.7E-7)	5.602E-6 (4.3E-7)	2.91%	0.10%	0.94	181.22	133.33
Dumbbell	200	6.473E-8 (1.6E-8)	5.837E-8 (6.2E-9)	5.815E-8 (5.4E-9)	11.32%	0.38%	1.19	233.23	222.49
Models in $\mathbb{R}^3$	Number of vertices	The mean (standard deviation) of the GCVT energy per Voronoi cell					Time (sec.)		
		iCVT	MCM-iCVT	MDE	r_1	r_2	iCVT	MCM	MDE
Armadillo	172,974	7.489E-6 (7.2E-6)	7.313E-6 (1.6E-6)	7.218E-6 (1.0E-6)	3.77%	1.32%	33,677	74,970	106,471
Knot	80,000	1.506E-5 (3.6E-3)	1.322E-5 (3.2E-3)	1.310E-5 (8.1E-4)	14.93%	0.91%	21,060	277,415	31,124
Bunny	72,020	1.799E-5 (1.1E-5)	1.600E-5 (3.6E-6)	1.575E-5 (1.7E-6)	14.24%	1.62%	11,686	222,604	50,586
Decocube	60,198	7.344E-5 (6.5E-5)	7.307E-5 (1.5E-5)	7.219E-5 (8.8E-6)	1.73%	1.23%	10,872	177,101	28,559
Sphere	32,594	2.592E-5 (4.8E-6)	2.566E-5 (3.2E-6)	2.557E-5 (2.9E-6)	1.37%	0.35%	2,359	81,459	7,745
Fertility	29,994	6.975E-6 (3.0E-6)	6.776E-6 (1.3E-6)	6.690E-6 (7.9E-7)	4.26%	1.28%	2,632	83,135	19,119

Table 2: *The mean and standard deviation of the CVT/GCVT energies per Voronoi cell. The 2D and 3D models are scaled into a unit square and a unit cube respectively. The ratios* $r_1 = \frac{F(\text{L-BFGS/iCVT}) - F(MDE)}{F(MDE)} \times 100\%$ *and* $r_2 = \frac{F(MCM) - F(MDE)}{F(MDE)} \times 100\%$ *measure the improvement of MDE in terms of the energy. We use 1000 generators for all 3D models.*

integers and are not equal to j. A mutative agent is computed as

$$G'_{k,j} = \lambda(G_{k,rand2} \ominus G_{k,rand3}) \oplus G_{k,rand1} \tag{11}$$

where λ is a scalar parameter.

Crossover. a binomial crossover is used to build a competitor $G''_{k,j}$ by copying elements from original and mutative agents:

$$g''_{k,j,i} = \begin{cases} g'_{k,j,i}, & \text{if } (rand_{k,j,i}[0,1] \leq Cr \text{ or } i = i_{rand}) \\ g_{k,j,i}, & \text{otherwise} \end{cases} \tag{12}$$

where the crossover rate C_r is a scalar parameter.

Selection. Each agent $G_{k,j}$ in the current generation has a offspring $G_{k+1,j}$ in the next generation by the selection operator:

$$G_{k+1,j} = \begin{cases} G''_{k,j}, & \text{if } f(G''_{k,j}) \leq f(G_{k,j}) \\ T(G_{k,j}), & \text{otherwise} \end{cases} \tag{13}$$

where $T(G_{k,j})$ performs as follows: for each generator $g_{k,j,i}$, if its Voronoi cell is simply connected, $g_{k,j,i}$ is moved to its Riemannian center; otherwise, it remains unchanged.

Termination conditions. The MDE algorithm is terminated if the relative change $\frac{o_k - o_{k+1}}{o_k}$ does not exceed a threshold τ in contiguous n_t iterations, where o_k is the population's best objective function value at generation k, τ and n_t are two scalar parameters.

6 Experimental Results

Implementation. We adopted 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 [Berntsen and Espelid 1992] to evaluate the area integral on triangle meshes. For each triangle, we used a symmetric quadrature 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 λ in the mutation equation (11), the crossover rate C_r in equation (12), the termination threshold τ, 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 more regular than those of MCM (see Figure 6).

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 CVTs. 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 global optimal solution. As Table 2 reports, MDE consistently outperforms MCM and iCVT in terms of energy. The small standard deviation of Voronoi cell areas

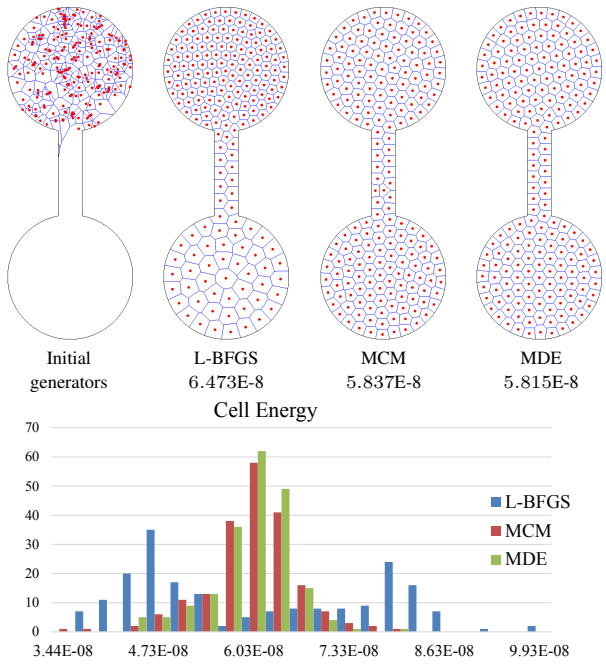


Figure 6: *2D Euclidean result. Since the initial seeds are clustered, the L-BFGS method gets stuck in a local optimal solution. Both the MCM method and the MDE method are insensitive to initialization. The cell energy histogram shows that the MDE result is more regular than that of MCM. See the accompanying video.*

Model	Standard deviation of Voronoi cell areas		
	iCVT	MCM-iCVT	MDE
Armadillo	1.346E-3	7.507E-4	4.697E-4
Knot	3.619E-3	1.104E-3	8.061E-4
Bunny	3.718E-3	1.108E-3	5.664E-4
Decocube	8.746E-3	2.191E-3	1.325E-3
Sphere	1.183E-3	8.198E-4	7.292E-4
Fertility	1.366E-3	6.109E-4	3.921E-4

Table 3: *Among the three methods, MDE produces GCVTs with the least standard deviation of Voronoi cell areas, justifying the better quality.*

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 $Q(t) = \frac{6}{\sqrt{3}}\frac{S}{ph}$, 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 $Q_{ave}(M)$ and $Q_{min}(M)$ denote the average and minimum $Q(t)$ of the input mesh M. $Q_{ave} = 1$ for completely regular triangulations and $Q_{ave} \ll 1$ 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.

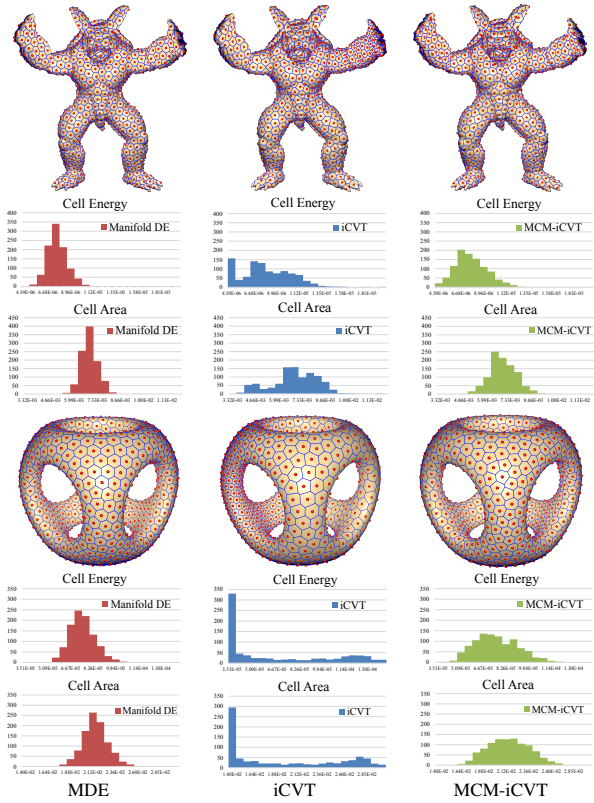
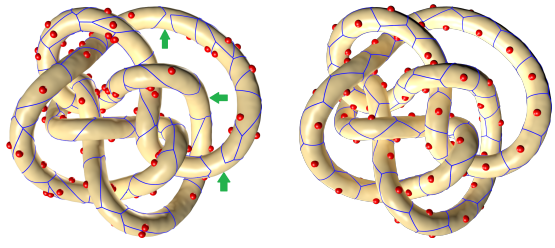


Figure 7: *Comparison on 3D models. Both visual checking and statistics show that the MDE results are of better quality than those of iCVT and MCM-iCVT. Images are rendered in high-resolution, allowing for close-up examination.*



(a) A random initialization (b) MDE result

Figure 8: *Given the initialization in (a), the iCVT method cannot proceed due to multiply-connected Voronoi cells (pointed by arrows). MDE can deal with such issues easily and produce high-quality GCVT.*

Model	Number of vertices	Number of generators	Type	Mesh quality		Quality of dual Delaunay mesh M_{CVT}			
				$Q_{min}(M)$	$Q_{ave}(M)$	$Q_{min}(M_{CVT})$	$Q_{ave}(M_{CVT})$	θ_{min}	$\theta_{min,ave}$
Kitten	21,920	1,000	Regular	0.147	0.788	0.643	0.921	39.48°	53.61°
			Irregular	0.001	0.380	0.485	0.894	30.46°	51.69°
			Noisy	0.108	0.800	0.579	0.891	33.84°	51.39°
Bunny	25,002	1,000	Regular	0.107	0.791	0.608	0.918	33.65°	53.26°
			Irregular	0.001	0.359	0.384	0.883	23.06°	50.80°
			Noisy	0.038	0.778	0.537	0.887	32.50°	51.02°
Eight	16,174	1,000	Regular	0.648	0.900	0.613	0.918	33.72°	53.48°
			Irregular	0.001	0.412	0.313	0.889	20.04°	51.20°
			Noisy	0.564	0.889	0.594	0.898	35.85°	51.69°

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. θ_{min} (resp. $\theta_{min,ave}$) are the smallest (resp. mean) angle of the minimal angles of all triangles in M_{CVT}.*

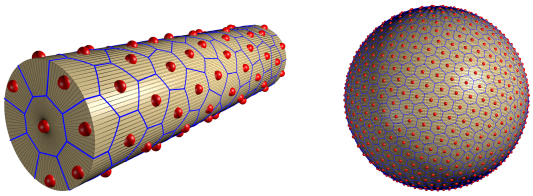


Figure 9: *Using exact geodesic algorithm, MDE works well on meshes with high degree of anisotropy.*

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

ALLIEZ, P., DE VERDIÈRE, É. C., DEVILLERS, O., AND ISENBURG, M. 2003. Isotropic surface remeshing. In *Shape Modeling International*, 49–58.

ALLIEZ, P., DE VERDIÈRE, É. C., DEVILLERS, O., AND ISENBURG, M. 2005. Centroidal voronoi diagrams for isotropic surface remeshing. *Graphical Models 67*, 3, 204–231.

BERNTSEN, J., AND ESPELID, T. O. 1992. Algorithm 706: Dcutri: An algorithm for adaptive cubature over a collection of triangles. *ACM Trans. Math. Softw. 18*, 3, 329–342.

BOENDER, C. G. E., AND ROMEIJN, H. E. 1995. Stochastic methods. In *Handbook of Global Optimization*, Kluwer Academic Publishers, 829–869.

CHENG, P., MIAO, C., LIU, Y.-J., TU, C., AND HE, Y. 2016. Solving the initial value problem of discrete geodesics. *Computer-Aided Design 70*, 144–152.

CRANE, K., WEISCHEDEL, C., AND WARDETZKY, M. 2013. Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph. 32*, 5, 152:1–152:11.

DAS, S., AND SUGANTHAN, P. N. 2011. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation 15*, 1, 4–31.

DAS, S., MULLICK, S. S., AND SUGANTHAN, P. N. 2016. Recent advances in differential evolution – an updated survey. *Swarm and Evolutionary Computation 27*, 1–30.

DU, Q., FABER, V., AND GUNZBURGER, M. 1999. Centroidal Voronoi tessellations: Applications and algorithms. *SIAM Review 41*, 4, 637–676.

DU, Q., GUNZBURGER, M. D., AND JU, L. 2003. Constrained centroidal Voronoi tessellations for surfaces. *SIAM Journal on Scientific Computing 24*, 5, 1488–1506.

ENGELBRECHT, A. P. 2006. *Fundamentals of Computational Swarm Intelligence*. John Wiley & Sons.

GHOSH, S., DAS, S., VASILAKOS, A. V., AND SURESH, K. 2012. On convergence of differential evolution over a class of continuous functions with unique global optimum. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 42*, 1, 107–124.

HORST, R., PARDALOS, P. M., AND THOAI, N. V. 2000. *Introduction to Global Optimization*. 2nd edition, Kluwer Academic Publishers.

KIMMEL, R., AND SETHIAN, J. 1998. Computing geodesic paths on manifolds. *Proceedings of National Academy of Sciences 95*, 8431–8435.

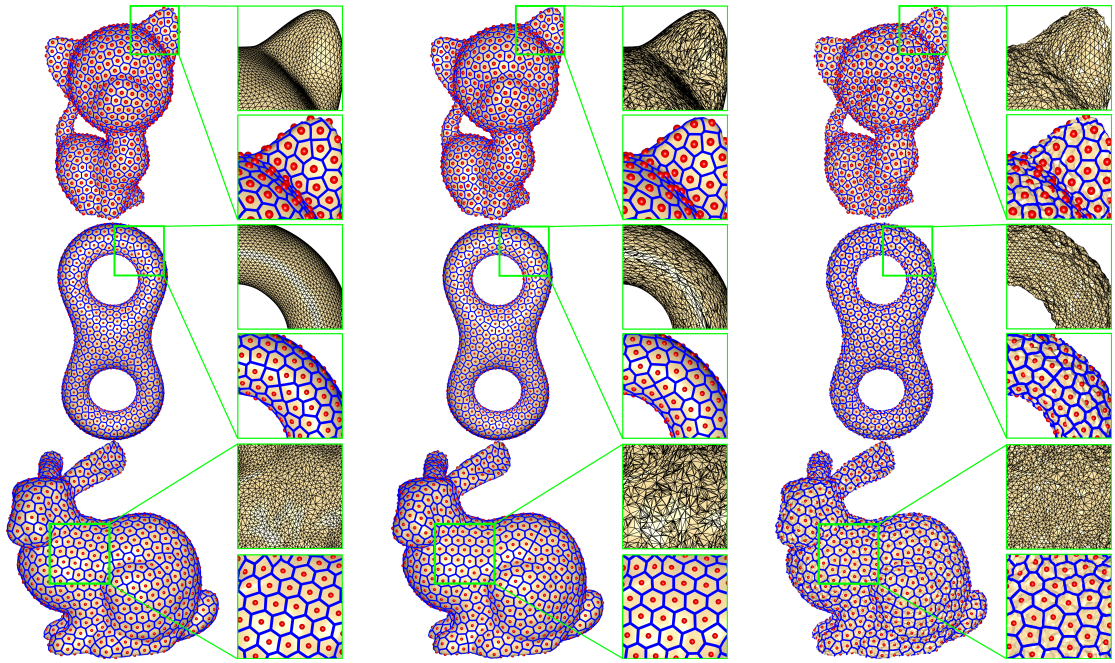
KORTE, B., AND VYGEN, J. 2006. *Combinatorial Optimization: Theory and Algorithms*. 3rd edition, Springer-Verlag.

LEUNG, Y.-S., WANG, X., HE, Y., LIU, Y.-J., AND WANG, C. C. L. 2015. A unified framework for isotropic meshing based on narrow-banded euclidean distance transformation. *Computational Visual Media 1*, 3, 239–251.

LÉVY, B., AND LIU, Y. 2010. Lp centroidal voronoi tessellation and its applications. *ACM Trans. Graph. 29*, 4, 119:1–119:11.

LIU, Y., WANG, W., LÉVY, B., SUN, F., YAN, D.-M., LU, L., AND YANG, C. 2009. On centroidal Voronoi tessellation – energy smoothness and fast computation. *ACM Trans. Graph. 28*, 4, 101:1–101:17.

LIU, Y.-J., CHEN, Z., AND TANG, K. 2011. Construction of iso-contours, bisectors, and Voronoi diagrams on triangulated surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence 33*, 8, 1502–1517.



(a) Regular triangulations (b) Irregular triangulations (c) Noisy meshes

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.*

LIU, Y.-J. 2015. Semi-continuity of skeletons in 2-manifold and discrete voronoi approximation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 37*, 9, 1938–1944.

LLOYD, S. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory 28*, 2, 129 – 137.

LU, L., SUN, F., PAN, H., AND WANG, W. 2012. Global optimization of centroidal Voronoi tessellation with monte carlo approach. *IEEE Transactions on Visualization and Computer Graphics 18*, 11, 1880–1890.

MITCHELL, J. S., MOUNT, D. M., AND PAPADIMITRIOU, C. H. 1987. The discrete geodesic problem. *SIAM Journal on Computing 16*, 4, 647–668.

PAPOULIS, A. 1991. *Probability, Random Variables and Stochastic Processes*. Third Edition, McGraw-Hill, Inc.

PRICE, K., STORN, R. M., AND LAMPINEN, J. A. 2005. *Differential Evolution: A Practical Approach to Global Optimization*. Springer.

QU, B., LIANG, J., WANG, Z., CHEN, Q., AND SUGANTHAN, P. 2016. Novel benchmark functions for continuous multimodal optimization with comparative results. *Swarm and Evolutionary Computation 26*, 23–34.

RONG, G., JIN, M., SHUAI, L., AND GUO, X. 2011. Centroidal voronoi tessellation in universal covering space of manifold surfaces. *Comput. Aided Geom. Des. 28*, 8, 475–496.

RONG, G., LIU, Y., WANG, W., YIN, X., GU, X., AND GUO, X. 2011. GPU-assisted computation of centroidal Voronoi tessellation. *IEEE Transactions on Visualization and Computer Graphics 17*, 3, 345–356.

VESTERSTROM, J., AND THOMSEN, R. 2004. A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In *Proc. 6th Congress on Evolutionary Computation*, 1980–1987.

WANDZURAT, S., AND XIAO, H. 2003. Symmetric quadrature rules on a triangle. *Computers & Mathematics with Applications 45*, 12, 1829–1840.

WANG, X., YING, X., LIU, Y.-J., XIN, S.-Q., WANG, W., GU, X., MUELLER-WITTIG, W., AND HE, Y. 2015. Intrinsic computation of centroidal Voronoi tessellation (cvt) on meshes. *Computer-Aided Design 58*, 51–61.

XU, C., WANG, T. Y., LIU, Y., LIU, L., AND HE, Y. 2015. Fast wavefront propagation (FWP) for computing exact geodesic distances on meshes. *IEEE Trans. Vis. Comput. Graph. 21*, 7, 822–834.

YAN, D., AND WONKA, P. 2016. Non-obtuse remeshing with centroidal Voronoi tessellation. *IEEE Trans. Vis. Comput. Graph. 22*, 9, 2136–2144.

YAN, D.-M., LÉVY, B., LIU, Y., SUN, F., AND WANG, W. 2009. Isotropic remeshing with fast and exact computation of restricted Voronoi diagram. *Comput. Graph. Forum 28*, 5, 1445–1454.

YAN, D., BAO, G., ZHANG, X., AND WONKA, P. 2014. Low-resolution remeshing using the localized restricted voronoi diagram. *IEEE Trans. Vis. Comput. Graph. 20*, 10, 1418–1427.

YING, X., WANG, X., AND HE, Y. 2013. Saddle vertex graph (svg): A novel solution to the discrete geodesic problem. *ACM Trans. Graph. 32*, 6, 170:1–170:12.

YING, X., XIN, S., AND HE, Y. 2014. Parallel Chen-Han (PCH) algorithm for discrete geodesics. *ACM Trans. Graph. 33*, 1, 9:1–9:11.

ZAHARIE, D. 2001. On the explorative power of differential evolution algorithms. In *Proc. 3rd Int. Workshop Symbolic and Numerical Algorithms on Scientific Computing*.

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 $\mathbf{G}_{k,j}$.

Lemma 1 *In the population at any kth generation, $k \geq 1$, the random variables $\mathbf{G}_{k,1}, \mathbf{G}_{k,2}, \cdots, \mathbf{G}_{k,N_p}$ 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 $\mathbf{G}_{k,1}, \mathbf{G}_{k,2}, \cdots, \mathbf{G}_{k,N_p}$ are mutually independent. To create an instantiation of $\mathbf{G}_{k+1,j}$, we only need to know four instantiations $G_{k,j}$, $G_{k,rand1}$, $G_{k,rand2}$, $G_{k,rand3}$ and $G_{k,rand4}$ at kth generation (ref. Eqs. (11-13)). In other words, the creation of $\mathbf{G}_{k+1,j}$ does not need any information from $\mathbf{G}_{k+1,j'}$, $j' \neq j$. Therefore, $\mathbf{G}_{k+1,1}, \mathbf{G}_{k+1,2}, \cdots, \mathbf{G}_{k+1,N_p}$ are mutually independent. □

Let α be a curve passing through a point $p \in M$ and denote the unit tangent vector of α at p as t_p. At any point $p \in M$, a local Frenet frame $\{t_p, n_p, b_p\}$ exists, where t_p is a unit tangent vector at p, n is the unit normal vector at p and $b_p = t_p \times n_p$. The tangent plane to M at p is spanned by any two independent tangent vectors at p and is denoted as $T_p(M)$. Given any nonzero vector $v \in T_p(M)$, there exists a unique parameterized geodesic $\gamma : (-\varepsilon, \varepsilon) \to M$, with $\gamma(0) = p$ and $\gamma'(0) = v$. The exponential map $exp_p(v) : v \in T_p(M) \to M$ at p maps every nonzero $v \in T_p(M)$ to a point q on M in two steps: (1) find the unique geodesic γ, with $\gamma(0) = p$ and $\gamma'(0) = v/|v|$, and (2) find q on γ with $d(p, q) = |v|$. 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 2π is called a *saddle* vertex.

Lemma 2 *For any point $p \in M$, let r_p be the maximal radius such that the geodesic disk $D_p = \{q \in M : d(p, q) \leq r_p\}$ 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 $d = g_1 \ominus g_2$ 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 $v(g_i)$, $i = 1, 2$, is an instantiation of a random variable $\mathbf{v}(g_i)$, and d is an instantiation of a random variable $\mathbf{d} = \mathbf{v}_{g_1} - (-\mathbf{v}_{g_2}) = \mathbf{v}_{g_1} + \mathbf{v}_{g_2}$. All these random variables $\mathbf{v}(g_1)$, $\mathbf{v}(g_2)$ and $\mathbf{d}$ are defined in $\mathbb{R}^2$ and their probability density functions are denoted as $f_{\mathbf{v}(g_1)}(v(g_1))$, $f_{\mathbf{v}(g_2)}(v(g_2))$ and $f_{\mathbf{d}}(d)$.

Let $D = G \ominus G' = \{g_{\sigma(i)} \ominus g'_i\}_{i=1}^m = \{d_i\}_{i=1}^m$ as defined in Eq. (9). We pack all m 2-dimensional vectors $\{v(g_{\sigma(i)})\}_{i=1}^m$ into a vector in $\mathbb{R}^{2m}$ and denote it as $V(G)$. Similarly, $V(G')$ is a vector in $\mathbb{R}^{2m}$ by packing $\{v(g'_i)\}_{i=1}^m$. Let $V(G)$ and $V(G')$ be instantiations of random variables $\mathbf{V}(G)$ and $\mathbf{V}(G')$, respectively. Each d_i is an instantiation of a random variable $\mathbf{d}_i = \mathbf{v}_{g_{\sigma(i)}} + \mathbf{v}_{g'_i}$. By packing m 2-dimensional vectors $\{d_i\}_{i=1}^m$, we regard D is an instantiation of a random variable $\mathbf{D} = \mathbf{V}(G) + \mathbf{V}(G')$ defined in $\mathbb{R}^{2m}$. Denote the probability density functions of $\mathbf{V}(G)$, $\mathbf{V}(G')$ and D as $f_{\mathbf{V}(G)}(V(G))$, $f_{\mathbf{V}(G')}(V(G'))$ and $f_{\mathbf{D}}(D)$, respectively.

Let $G''' = D \oplus G'' = \{d_i \oplus g''_i\}_{i=1}^m = \{g'''_i\}_{i=1}^m$ as defined in Eq. (10). Each point/generator g''_i can be inversely mapped by $exp_{g''_i}$ as the zero vector $\widetilde{v}(g''_i) = \mathbf{0}$ in the tangent space $T_{g''_i}(M)$. Let $\widetilde{V}(G'')$ be a vector in $\mathbb{R}^{2m}$ by packing m 2-dimensional zero vectors $\{\widetilde{v}(g''_i)\}_{i=1}^m$, which is an instantiation of a random variable $\widetilde{\mathbf{V}}(G'')$. If $g'''_i \in D_{g''_i}$, $i = 1, 2, \cdots, m$, G''' can be regarded as an instantiation of a random variable $\widetilde{\mathbf{V}}(G''') = \mathbf{D} + \widetilde{\mathbf{V}}(G'')$ defined in $\mathbb{R}^{2m}$, whose probability density function is $f_{\widetilde{\mathbf{V}}(G''')}(\widetilde{V}(G'''))$.

Lemma 3 *Let $r = \min\{r_p : \forall p \in M\}$ and $\widetilde{D}_p = \{q \in M : d(p, q) \leq r\}$. Let $G = (G_{k,j_1} \ominus G_{k,j_2}) \oplus G_{k,j_3}$, $j_1 \neq j_2$, $j_1 \neq j_3$ and $j_2 \neq j_3$. If $g_{k,j_1,\sigma(i)} \in \widetilde{D}_{g_{k,j_2,i}}$, $i = 1, 2, \cdots, m$, where $\sigma(i)$ is the permutation defined by agent matching in Section 5.2, then $f_{\widetilde{\mathbf{V}}(G)}(\widetilde{V}(G)) = f_{\mathbf{V}(G_{k,j_1})}(V(G_{k,j_1})) * f_{\mathbf{V}(G_{k,j_2})}(V(G_{k,j_2})) * f_{\widetilde{\mathbf{V}}(G_{k,j_3})}(\widetilde{V}(G_{k,j_3}))$, where $*$ is the $2m$-dimensional linear convolution operation.*

Proof. First, we have $f_{\mathbf{d}}(d) = f_{\mathbf{v}(g_1)}(v(g_1)) * f_{\mathbf{v}(g_2)}(v(g_2))$ [Papoulis 1991], where $*$ is the 2-dimensional linear convolution operation, because:

- $\mathbf{g}_1$ and $\mathbf{g}_2$ are two independent random variables of generators g_1 and g_2 respectively, and $g_1 \in D_{g_2}$,
- $d = g_1 \ominus g_2$ and $\mathbf{d} = \mathbf{v}_{g_1} + \mathbf{v}_{g_2}$ is a function of two random variables $\mathbf{g}_1$ and $\mathbf{g}_2$ in $\mathbb{R}^2$.

Secondly, $D = G_{k,j_1} \ominus G_{k,j_2} = \{g_{k,j_1,\sigma(i)} \ominus g_{k,j_2,i}\}_{i=1}^m = \{d_i\}_{i=1}^m$ and $\mathbf{D} = \mathbf{V}(G_{k,j_1}) + \mathbf{V}(G_{k,j_2})$. Since $\mathbf{G}_{k,j_1}$ and $\mathbf{G}_{k,j_2}$ are independent, $\mathbf{v}(g_{k,j_1,\sigma(i)})$ and $\mathbf{v}(g_{k,j_2,i})$ are independent, $i = 1, 2, \cdots, m$. Then we have $f_{\mathbf{D}}(D) = f_{\mathbf{V}(G_{k,j_1})}(V(G_{k,j_1})) * f_{\mathbf{V}(G_{k,j_2})}(V(G_{k,j_2}))$.

Thirdly, since $\mathbf{G}_{k,j_1}$, $\mathbf{G}_{k,j_2}$ and $\mathbf{G}_{k,j_3}$ are mutually independent, $\mathbf{D}$ and $\widetilde{\mathbf{V}}(G_{k,j_3})$ are independent. Since $g_{k,j_1,\sigma(i)} \in \widetilde{D}_{g_{k,j_2,i}}$, $i = 1, 2, \cdots, m$, the length of tangent vectors $d_i = g_{k,j_1,\sigma(i)} \ominus g_{k,j_2,i}$ is no larger than r. Therefore, $g_i = d_i \oplus g_{k,j_3,i}$ satisfies $g_i \in \widetilde{D}_{g_{k,j_3,i}}$, $i = 1, 2, \cdots, m$. Then we have $f_{\widetilde{\mathbf{V}}(G)}(\widetilde{V}(G)) = f_{\mathbf{D}}(D) * f_{\widetilde{\mathbf{V}}(G_{k,j_3})}(\widetilde{V}(G_{k,j_3}))$. That completes the proof. □

Liu et al. [2009] showed that the objective function $F(G)$ in Eq. (1) is C^2 in a convex region in $\mathbb{R}^2$ if the density function $\rho(x)$ is C^2. When we examine their proof, it is also applicable to the domain of a compact manifold M. Therefore if $\rho(x)$ is C^2 on M, then $F(G)$ is C^2 on M.

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 $g_{k,j_1,\sigma(i)} \in D_{g_{k,j_2,i}}$, $\forall k, j_1, j_2, i$, $j_1 \neq j_2$, then $f_{\mathbf{G}_{k,j}}(G_{opt}) > 0$ for all $k > 0$ and all $j = 1, 2, \cdots, N_p$. Furthermore, $f_{\mathbf{G}_{k,j}}(G_{k,j})$ converges to $\delta(\widetilde{D}_{opt})$ as $k \to \infty$, where $\widetilde{D}_{opt}$ is a $2m$-dimensional vector by packing m two-dimensional vectors in $\{\widetilde{d}_i\}_{i=1}^m = G_{k,j} \ominus G_{opt}$ and δ is the $2m$-dimensional Dirac delta function which is zero everywhere except at the place where the agent $G_{k,j}$ is exactly the same as the global optimal solution G_{opt}.*

Theorem 1 requires that $g_{k,j_1,\sigma(i)} \in D_{g_{k,j_2,i}}$, $\forall k, j_1, j_2, i$, $j_1 \neq j_2$. This condition is not always satisfied in Algorithm 1. But in our experiments, Algorithm 1 works quite well even without explicitly guaranteeing this condition.