Geometric Matching Algorithms for Two Realistic Terrains[⋆]

Sang Duk Yoon, Min-Gyu Kim, Wanbin Son, and Hee-Kap Ahn

Department of Computer Science and Engineering, POSTECH, Pohang, Korea
{egooana,alsrbbk,mnbiny,heekap}@postech.ac.kr

Abstract. We consider a geometric matching of two realistic terrains, each of which is modeled as a piecewise-linear bivariate function. For two realistic terrains f and g where the domain of g is relatively larger than that of f, we seek to find a translated copy f' of f such that the domain of f' is a sub-domain of g and the L_∞ or the L_1 distance of f' and g restricted to the domain of f' is minimized. In this paper, we show a tight bound on the number of different combinatorial structures that f and g can have under translation in their projections on the xy-plane. We give a deterministic algorithm and a randomized algorithm that compute an optimal translation of f with respect to g under L_∞ metric. We also give a deterministic algorithm that computes an optimal translation of f with respect to g under L_1 metric.

1 Introduction

In the terrain matching problem, we are given two terrains and the goal is to measure the similarity between two terrains. Terrain matching has been extensively used for various applications to locate the exact position of objects such as aircrafts [7,13,18], cruise missiles [3,9], underwater vehicles [16,19,20], rockets and robots for space missions [10,17].

In these applications, terrain matching is used to specify the location of an object by constructing local terrain data around the object and finding the most similar sub-terrain in the existing global terrain data. A typical method to find the most similar sub-terrain is feature matching. Well known examples of features are linear edges, $2D$ curves, contour lines and Gaussian curvatures [7,9,13]. These features describe some characteristics of a terrain, but may not fully reflect the geometric properties of the terrain.

Moroz and Aronov [15] and Agarwal et al. [1] dealt with terrain matching as a geometric matching problem. They defined a terrain f as a piecewise-linear bivariate function $f : \mathbb{D}_f \to \mathbb{R}$, where $\mathbb{D}_f$ is a triangulated domain of f in the xy-plane. For each vertex of the triangulation, the function value $f(v)$ is given, and the other values are given by the linear interpolation within each triangle. They gave algorithms that compute exact L_∞, L_1 (under vertical scaling and

[⋆] This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. 2011-0030044)

translation) and L_2 distances between two terrains, respectively. These algorithms only handle two input terrains defined on the same domain, and to the best of our knowledge, there is no research about matching two triangulated terrains defined on different domains.

We deal with a geometric matching problem concerning two terrains defined by the piecewise-linear bivariate functions on triangulated domains, but in this paper we do not require that two terrains have the same domain. Let $\mathbb{D}_f + t = \{p + t \mid p \in \mathbb{D}_f\}$ be a translated image of $\mathbb{D}_f$ by a translation vector $t \in \mathbb{R}^2$. We define the distance between two terrains as follows.

Definition 1. *Let $f : \mathbb{D}_f \to \mathbb{R}$ and $f' : \mathbb{D}_{f'} \to \mathbb{R}$ be two terrains such that $\mathbb{D}_{f'} = \mathbb{D}_f + t$ for a translation vector $t \in \mathbb{R}^2$. The distance $d_\infty(f, f')$ between f and f' under L_∞ metric is*

$$\min_{h\in\mathbb{R}} \max_{p\in\mathbb{D}_f} |(f(p) + h) - f'(p + t)|$$

and the distance $d_1(f, f')$ between f and f' under L_1 metric is

$$\min_{h\in\mathbb{R}} \iint\limits_{p\in\mathbb{D}_f} |(f(p) + h) - f'(p + t)|dp.$$

We use two different distances to measure the similarity between two terrains. The distance d_∞ measures the min-max vertical distance under vertical translation h and the distance d_1 measures the minimum volume under vertical translation h.

Our problem can be stated as follows:

Problem (Terrain Matching). Given two terrains $f : \mathbb{D}_f \to \mathbb{R}$ and $g : \mathbb{D}_g \to \mathbb{R}$, find an optimal translation vector t^* such that $\mathbb{D}^* = \mathbb{D}_f + t^* \subset \mathbb{D}_g$ and the distance between f and $g\restriction_{\mathbb{D}^*}$ is minimized, where $g\restriction_{\mathbb{D}^*}$ denotes the restriction of g to $\mathbb{D}^*$.

In many computational geometric problems, there is a certain gap between the worst-case computational complexity of an algorithm and the actual running time of the algorithm on inputs from real world applications [14]. The same phenomenon happens for the terrain matching problem. So, it is an important issue to develop an algorithm that is efficient for a realistic input. We assume that our input terrains satisfy some realistic constraints. A *realistic terrain* is a terrain with three additional constraints. In the following definition, k and r are assumed to be positive constants.

Definition 2 ([14]). *A terrain $f : \mathbb{D}_f \to \mathbb{R}$ is a realistic terrain if it satisfies the followings:*
(a) The triangulation of $\mathbb{D}_f$ is a k-low-density triangulation.
(b) For the smallest rectangle that contains $\mathbb{D}_f$, the ratio of the length of a short side to the length of a long side is $1 : r$.
(c) The longest edge in the triangulation of $\mathbb{D}_f$ is at most constant times as long as the shortest one.

A planar triangulation $\mathcal{T}$ is called a *k-low-density triangulation* if for any axis-aligned square R with side length s, the number of edges of $\mathcal{T}$ with length greater than or equal to s that intersect R is at most k. Also, by Definition 2(b), we assume that the domain of a realistic terrain is an axis-aligned rectangle with constant side-length ratio.

1.1 Our results

We present first algorithms for matching two triangulated terrains with different domains and also show geometric properties between two realistic terrains. To solve the terrain matching problem under L_∞ metric, we first gather two-dimensional translation vectors t such that $\mathbb{D}_f + t \subset \mathbb{D}_g$. Then, we subdivide the set of translation vectors into a partition such that the translation vectors t of a cell of the partition correspond to "one combinatorial structure" between two triangulations of $\mathbb{D}_f + t$ and $\mathbb{D}_g$. For each cell of the partition, we can find a translation vector that minimizes the distance among the translation vectors in the cell by reducing it to a linear programming problem.

To solve the terrain matching problem under L_1 metric, we need to treat the amount of vertical translation h of f explicitly. After we concatenate h as a third coordinate of the two-dimensional translation vectors $t = (t_x, t_y)$, we subdivide the set of three-dimensional translation vectors (t_x, t_y, h) into a partition such that the volume function between f and g for the translation vectors (t_x, t_y, h) of a cell can be expressed by a single formula. For each cell of the partition, we find a three-dimensional translation vector that minimizes the distance among the translation vectors of the cell by using numerical methods.

Our results are twofold: Let m (resp. n) be the number of triangles in the triangulation of $\mathbb{D}_f$ (resp. $\mathbb{D}_g$), assuming that $m \leq n$. The side lengths of $\mathbb{D}_f$ (resp. $\mathbb{D}_g$) are a and $\mathsf{a}r$ (resp. a' and $\mathsf{a}'r'$) for a positive constant r (resp. r'). Let $A = m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n$.

1. Under L_∞ metric,
   - We show that the number of different combinatorial structures between the triangulation of $\mathbb{D}_f$ and the triangulation of $\mathbb{D}_g$ is $O(nmA)$, and this bound is tight if $\mathsf{a}' > 2\mathsf{a}$.
   - We present a deterministic algorithm and a randomized algorithm for the terrain matching problem. The deterministic algorithm runs in $O(nmA^{4/3+\delta})$ time using $O(n + A^2)$ space for a fixed $\delta > 0$, and the randomized algorithm runs in $O(nmA \log n \log^2 A \log^2 \log A)$ expected time using $O(n + A^2)$ space. The randomized algorithm outperforms the deterministic algorithm when $m = \Omega(\log^3 A)$.
   - The time complexity of the randomized algorithm is near linear to the number of different combinatorial structures. It seems hard to avoid searching the whole combinatorial structures, so our algorithms run reasonably fast.
2. Under L_1 metric,

– We show that the number of different formulae of the volume function defined between f and g is $O(nmA^4)$.
– We present a deterministic algorithm for the terrain matching problem that runs in $O(nmA^4)$ time using $O(n + A^3)$ space.

The factor $A = (m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$ is $O(m)$ when $m : n \approx \mathsf{a}^2 : \mathsf{a}'^2$. This condition holds when edge lengths of triangles in both realistic terrains are asymptotically the same. Many real world applications use terrains with this condition to match. With $A = O(m)$, the running times and the space complexities of our algorithms are linear to n (except the randomized algorithm); it means that our algorithms can be used as a query algorithm for finding the most similar part of large terrain data g for query terrain data f which runs in time linear to the size of the database.

2 Translation Space under L_∞ metric

Let $\mathcal{S}$ be a set of translation vectors t that satisfies $\mathbb{D}_f + t \subset \mathbb{D}_g$, i.e., $\mathcal{S} = \{t \in \mathbb{R}^2 \mid \mathbb{D}_f + t \subset \mathbb{D}_g\}$. We call $\mathcal{S}$ the *translation space* of f and g. As mentioned before, our goal is to find an optimal translation vector in $\mathcal{S}$. To find it among infinitely many translation vectors in $\mathcal{S}$, we need to investigate geometric properties of input terrains.

Let the triangulations of $\mathbb{D}_f$ and $\mathbb{D}_g$ be $\mathcal{T}_f$ and $\mathcal{T}_g$, respectively. In this section, we show that $\mathcal{S}$ can be subdivided into the finite number of cells such that the interior of each cell induces the same combinatorial structure of the overlay of the triangulations. Then we show a tight upper bound on the number of the combinatorially different sets of translation vectors.

2.1 Candidate pairs defining the distance between two terrains

For a translation vector t, let $\mathcal{O}(f, g, t)$ be the overlay of $\mathcal{T}_f + t$ and $\mathcal{T}_g$ where $\mathcal{T}_f + t = \{p + t \mid p \in \mathcal{T}_f\}$ be a translated image of $\mathcal{T}_f$ (Figure 1). The following lemma shows that there is a vertex $v \in \mathcal{O}(f, g, t)$ that realizes the distance between f and $g\restriction_{\mathbb{D}_f+t}$, i.e., $d_\infty(f, g\restriction_{\mathbb{D}_f+t}) = |(f(v-t) + h') - g(v)|$, where $h' = \operatorname*{argmin}_{h\in\mathbb{R}} \max_{p\in\mathbb{D}_f} |(f(p)+h) - g(p+t)|$. Let $f[\mathbb{D}] = \{(x, y, f(p)) \mid p = (x, y) \in \mathbb{D} \subseteq \mathbb{D}_f\}$, and $g[\mathbb{D}] = \{(x, y, g(p)) \mid p = (x, y) \in \mathbb{D} \subseteq \mathbb{D}_g\}$.

Lemma 1. *There is a vertex of $\mathcal{O}(f, g, t)$ that realizes $d_\infty(f, g\restriction_{\mathbb{D}_f+t})$ for any translation vector $t \in \mathcal{S}$.*

Proof. Assume that none of the vertices of $\mathcal{O}(f, g, t)$ realizes $d_\infty(f, g\restriction_{\mathbb{D}_f+t})$. Let $p \in \mathbb{D}_f$ be a point that realizes $d_\infty(f, g\restriction_{\mathbb{D}_f+t})$. There are two possible cases: $p + t$ is contained in a cell of $\mathcal{O}(f, g, t)$ or in the interior of an edge of $\mathcal{O}(f, g, t)$.

Suppose that $p + t$ is contained in a cell of $\mathcal{O}(f, g, t)$. This means that $p + t$ lies in the interior of a triangle $\triangle$ of $\mathcal{T}_f + t$ and in the interior of a triangle $\triangle'$ of $\mathcal{T}_g$. In $\mathbb{R}^3$, $f[\triangle]$ and $g[\triangle']$ must be parallel; otherwise, it contradicts the fact

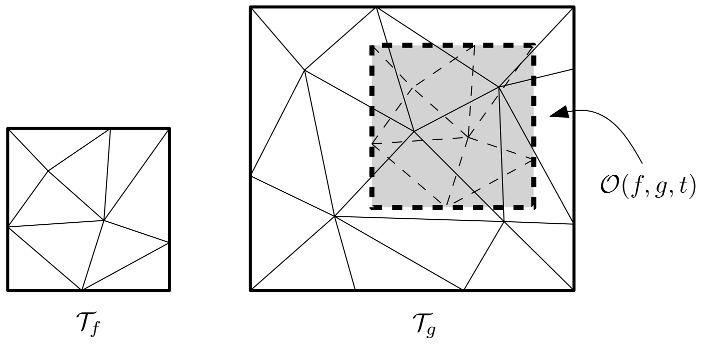

Fig. 1. An example of the overlay $\mathcal{O}(f,g,t)$.

that p realizes $d_\infty(f, g\restriction_{\mathbb{D}_f+t})$. Therefore, any vertex v made by the overlay of $\triangle$ and $\triangle'$ also realizes $d_\infty(f, g\restriction_{\mathbb{D}_f+t})$, a contradiction.

Suppose that $p+t$ is contained in the interior of an edge of $\mathcal{O}(f,g,t)$. Without loss of generality, we say that $p+t$ lies in the interior of a triangle $\triangle$ of $\mathcal{T}_f+t$ and in the interior of an edge e of $\mathcal{T}_g$. In $\mathbb{R}^3$, $f[\triangle]$ and $g[e]$ are parallel; otherwise, it contradicts the fact that p realizes $d_\infty(f, g\restriction_{\mathbb{D}_f+t})$. Therefore, any vertex v made by the overlay of $\triangle$ and e also realizes $d_\infty(f, g\restriction_{\mathbb{D}_f+t})$, a contradiction $\sqcup\!\sqcap$

Each vertex of $\mathcal{O}(f,g,t)$ corresponds to a vertex-triangle pair or an edge-edge pair of two triangulations $\mathcal{T}_f+t$ and $\mathcal{T}_g$. We define the *combinatorial structure* $\mathcal{C}(t)$ between $\mathcal{T}_f$ and $\mathcal{T}_g$ at $t \in \mathcal{S}$ as the set of these pairs between $\mathcal{T}_f+t$ and $\mathcal{T}_g$.

2.2 Subdividing translation space

Now we describe how to subdivide $\mathcal{S}$ into cells such that $\mathcal{C}(t) = \mathcal{C}(t')$ for any two translation vectors t and t' in the interior of a cell. We denote by $\mathcal{M}$ the subdivision. The combinatorial structure corresponding to an edge or a vertex of $\mathcal{M}$ is the union of the combinatorial structures of the adjacent cells of the edge or the vertex in $\mathcal{M}$.

Let us consider the different combinatorial structures induced by two triangles $\triangle$ and $\triangle'$ from $\mathcal{T}_f$ and $\mathcal{T}_g$, respectively (Figure2(a)). Let $\mathcal{S}_\triangle = \{t \in \mathbb{R}^2 \mid (\triangle+t) \cap \triangle' \neq \emptyset\}$ (the gray region in Figure 2(b)). For an edge-edge pair (e,e') where e and e' are edges of $\triangle$ and $\triangle'$, respectively, the set of translation vectors t such that $e+t$ and e' intersect forms a parallelogram in $\mathcal{S}_\triangle$ (Figure 2(b)). For a vertex-triangle pair $(\triangle, v)$ where v is a vertex of $\triangle'$, the set of translation vectors t such that $\triangle+t$ and v intersect forms a triangle. Similarly, for a vertex-triangle pair $(v, \triangle')$ where v is a vertex of $\triangle$, the set of translation vectors such that $v+t$ and $\triangle'$ intersect forms a triangle. We say that the parallelogram in $\mathcal{S}_\triangle$ is defined by an edge-edge pair and the triangle in $\mathcal{S}_\triangle$ is defined by a vertex-triangle pair. The overlay of the parallelograms and triangles defined by all edge-edge and vertex-triangle pairs between $\triangle$ and $\triangle'$, respectively, subdivides $\mathcal{S}_\triangle$ into cells such that the interior of each cell corresponds to exactly one combinatorial structure between $\triangle$ and $\triangle'$ (Figure 2(c)).

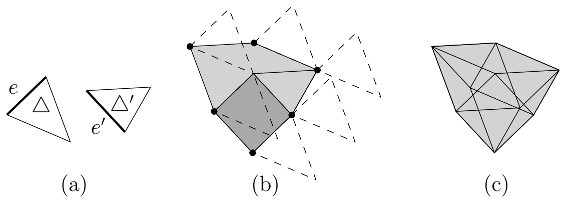

Fig. 2. (a) Two triangles $\triangle$ and $\triangle'$, and two edges e and e' of them. (b) The set of translation vectors t such that $e + t \cap e' \neq \emptyset$ in $\mathcal{S}_\triangle$ (darker parallelogram). (c) The resulting subdivision of $\mathcal{S}_\triangle$.

The subdivision $\mathcal{M}$ induced by $\mathcal{T}_f$ and $\mathcal{T}_g$ is constructed as follows. An edge-edge pair and a vertex-triangle pair between $\mathcal{T}_f$ and $\mathcal{T}_g$ define a parallelogram and a triangle in $\mathcal{S}$, respectively. The subdivision $\mathcal{M}$ is constructed by overlaying the parallelograms and the triangles. In the following lemma, we show that it suffices to overlay the triangles to construct $\mathcal{M}$.

Lemma 2. *The subdivision $\mathcal{M}$ can be constructed by overlaying the triangles in $\mathcal{S}$ defined by all the vertex-triangle pairs between $\mathcal{T}_f$ and $\mathcal{T}_g$.*

Proof. For a parallelogram in $\mathcal{S}$ defined by an edge-edge pair (e, e'), we will show that the edges of the parallelogram are contained in the overlay of the triangles defined by the vertex-triangle pairs.

For a translation vector t in each edge of the parallelogram, a vertex of $e+t$ lies on e', or a vertex of e' lies on $e+t$. Let us consider an edge e_v of the parallelogram which consists of the translation vectors t such that a vertex v of $e+t$ lies on e'. For a triangle $\triangle$ in $\mathcal{T}_g$ that has e' as an edge, the vertex-triangle pair $(v, \triangle)$ defines a triangle in $\mathcal{S}$. By the way to choose v and $\triangle$, a translation vector of e_p is contained in an edge of the triangle defined by the vertex-triangle pair $(v, \triangle)$. Similarly, the remaining edges of the parallelogram are edges of triangles defined by vertex-triangle pairs. $\square$

We propose the simplified way to construct $\mathcal{M}$ as follows. The triangle in $\mathcal{S}$ defined by a vertex of $\mathcal{T}_f$ and a triangle $\triangle$ of $\mathcal{T}_g$ is a translated copy of $\triangle$. For a vertex v of $\mathcal{T}_f$, the set of triangles in $\mathcal{S}$ that are defined by v and the triangles in $\mathcal{T}_g$ forms a translated copy of $\mathcal{T}_g$. Let $\mathcal{T}_g(v)$ be the translated copy of $\mathcal{T}_g$ formed by a vertex v of $\mathcal{T}_f$. The triangle in $\mathcal{S}$ defined by a vertex of $\mathcal{T}_g$ and a triangle $\triangle$ of $\mathcal{T}_f$ is a translated copy of $-\triangle$, where $-\triangle$ is the reflection through the origin of $\triangle$. For a vertex u of $\mathcal{T}_g$, the set of triangles in $\mathcal{S}$ that are defined by u and the triangles in $\mathcal{T}_f$ forms a translated copy of $-\mathcal{T}_f$, where $-\mathcal{T}_f$ is a set of $-\triangle$ for all $\triangle \in \mathcal{T}_f$. Let $\mathcal{T}_f(u)$ be the translated copy of $-\mathcal{T}_f$ formed by a vertex u of $\mathcal{T}_g$. By Lemma 2, $\mathcal{M}$ is the overlay of $\mathcal{T}_f(u)$ and $\mathcal{T}_g(v)$ for all vertices u of $\mathcal{T}_g$ and v of $\mathcal{T}_f$, restricted to $\mathcal{S}$.

2.3 Complexity of the subdivision $\mathcal{M}$

We analyse the number of cells in $\mathcal{M}$ for two terrains. Since $\mathcal{M}$ is a planar subdivision, we can bound the number of cells by bounding the number of vertices of $\mathcal{M}$.

For two terrains, not necessarily realistic, the subdivision of their translation space has $O(n^2m^2)$ cells; $\mathcal{M}$ is the overlay of $\mathcal{T}_f(u)$ and $\mathcal{T}_g(v)$ for all vertices u of $\mathcal{T}_g$ and v of $\mathcal{T}_f$, restricted to $\mathcal{S}$. So, $\mathcal{M}$ is the overlay of $O(nm)$ edges and has $O(n^2m^2)$ cells. There is an example of two triangulations $\mathcal{T}_f$ and $\mathcal{T}_g$ such that $\mathcal{M}$ has $\Omega(n^2m^2)$ cells (Figure 3), so the bound above is tight. However, the worst

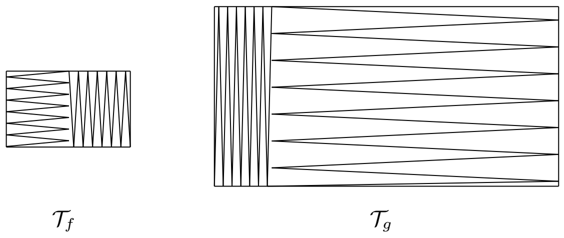

Fig. 3. Two triangulations $\mathcal{T}_f$ and $\mathcal{T}_g$ realizing $\Omega(n^2m^2)$ different combinatorial structures.

case scenario hardly happens in real world applications. We are going to present a better bound for realistic terrains. First of all, we introduce some properties of a realistic terrain.

Lemma 3 ([14]). *Let $f : \mathbb{D}_f \to \mathbb{R}$ be a realistic terrain such that $\mathcal{T}_f$ has n triangles and the side lengths of $\mathbb{D}_f$ are a and $\mathsf{a}r$ for a positive constant r. Then the following conditions hold.*

- *All edges in $\mathcal{T}_f$ have length $\Theta(\frac{\mathsf{a}}{\sqrt{n}})$.*
- *Let R be a rectangle that intersects $\mathcal{T}_f$, of which both side lengths are $\Omega(\frac{\mathsf{a}}{\sqrt{n}})$, and that has total area $\mathcal{R}$. Then R intersects $O(\frac{\mathcal{R}}{\mathsf{a}^2}n)$ triangles of $\mathcal{T}_f$.*

Next, the following lemma describes an upper bound of the number of cells in $\mathcal{M}$ when f and g are realistic terrains.

Lemma 4. *Let $f : \mathbb{D}_f \to \mathbb{R}$ and $g : \mathbb{D}_g \to \mathbb{R}$ be the two realistic terrains such that $\mathcal{T}_f$ (resp. $\mathcal{T}_g$) has m (resp. n) triangles and the side lengths of $\mathbb{D}_f$ (resp. $\mathbb{D}_g$) are a and $\mathsf{a}r$ (resp. a' and $\mathsf{a}'r'$) for a positive constant r (resp. r'). Then, the number of cells in the subdivision $\mathcal{M}$ is $O(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$, and the number of pairs in a combinatorial structure is $O(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$. When edge lengths of $\mathcal{T}_f$ and $\mathcal{T}_g$ are asymptotically same, i.e., $\frac{\mathsf{a}}{\sqrt{m}} = \Theta(\frac{\mathsf{a}'}{\sqrt{n}})$, the number of cells becomes $O(nm^2)$.*

Proof. There are two types of vertices of $\mathcal{M}$: either vertices of $\mathcal{T}_f(u)$ and $\mathcal{T}_g(v)$ or intersections between edges in $\mathcal{T}_f(u)$ and $\mathcal{T}_g(v)$, for all vertices $v \in \mathcal{T}_f$ and $u \in \mathcal{T}_g$. It is easy to count the number of vertices of the first type. The total number of vertices of $\mathcal{T}_f(u)$ and $\mathcal{T}_g(v)$ for all vertices $v \in \mathcal{T}_f$ and $u \in \mathcal{T}_g$ is $O(nm)$. The vertices of the second type can be divided into three subtypes: intersections made by $\mathcal{T}_g(v)$ and $\mathcal{T}_g(v')$, $\mathcal{T}_f(u)$ and $\mathcal{T}_f(u')$, and $\mathcal{T}_f(u)$ and $\mathcal{T}_g(v)$, for all vertices $v, v' \in \mathcal{T}_f$ and $u, u' \in \mathcal{T}_g$.

Intersections made by $\boldsymbol{\mathcal{T}_g(v)}$ and $\boldsymbol{\mathcal{T}_g(v')}$ There are $\binom{m}{2} = O(m^2)$ ways to choose v and v' from the vertices of $\mathcal{T}_f$. By Lemma 3, the length of the longest edge of $\mathcal{T}_g$ is smaller than $\frac{\mathsf{a}'}{\sqrt{n}}i$, for a constant i, so any edge of $\mathcal{T}_g(v)$ can be contained in a square R of side length $\frac{\mathsf{a}'}{\sqrt{n}}i$. Again by Lemma 3, R intersects $O(1)$ triangles in $\mathcal{T}_g(v')$, so an edge of $\mathcal{T}_g(v)$ also intersects $O(1)$ edges in $\mathcal{T}_g(v')$. The number of edges in $\mathcal{T}_g(v)$ is $O(n)$, so $\mathcal{T}_g(v)$ and $\mathcal{T}_g(v')$ have $O(n)$ intersections. Therefore, the total number of intersections of this type is $O(nm^2)$.

Intersections made by $\boldsymbol{\mathcal{T}_f(u)}$ and $\boldsymbol{\mathcal{T}_f(u')}$ There are $\binom{n}{2} = O(n^2)$ ways to choose u and u' from the vertices of $\mathcal{T}_g$. However, not all of the pairs of $\mathcal{T}_f(u)$ and $\mathcal{T}_f(u')$ have an intersection. For $\mathcal{T}_f(u)$ defined by a vertex u in $\mathcal{T}_g$, let us count the number of vertices u' in $\mathcal{T}_g$ such that $\mathcal{T}_f(u)$ intersects $\mathcal{T}_f(u')$. The relative position between $\mathcal{T}_f(u)$ and $\mathcal{T}_f(u')$ is the same to the relative position between u and u'. So, $\mathcal{T}_f(u)$ and $\mathcal{T}_f(u')$ have an intersection if the distance between u and u' is $O(\mathsf{a})$. By Lemma 3, the number of vertices of $\mathcal{T}_g$ contained in a square of side length $O(\mathsf{a})$ centered at u is $O((\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$. The number of intersections between $\mathcal{T}_f(u)$ and $\mathcal{T}_f(u')$ is $O(m)$ by an analogous argument in the first subcase. Therefore, the total number of intersections of this type is $O((\frac{\mathsf{a}}{\mathsf{a}'})^2 n^2 m)$.

Intersections made by $\boldsymbol{\mathcal{T}_f(u)}$ and $\boldsymbol{\mathcal{T}_g(v)}$ There are $O(nm)$ ways to choose u from the vertices of $\mathcal{T}_g$ and v from the vertices of $\mathcal{T}_f$. By Lemma 3, the length of the longest edge of $\mathcal{T}_f$ is smaller than $\frac{\mathsf{a}}{\sqrt{m}}i$ for a constant i, so any edge of $\mathcal{T}_f(u)$ can be contained in a square R of side length $\frac{\mathsf{a}}{\sqrt{m}}i$. Again by Lemma 3, R intersects $O((\frac{\mathsf{a}}{\mathsf{a}'})^2 \frac{n}{m})$ triangles in $\mathcal{T}_g(v)$, so any edge of $\mathcal{T}_f(u)$ intersects $O((\frac{\mathsf{a}}{\mathsf{a}'})^2 \frac{n}{m})$ edges in $\mathcal{T}_g(v)$. The number of edges in $\mathcal{T}_f(u)$ is $O(m)$, so $\mathcal{T}_f(u)$ and $\mathcal{T}_g(v)$ have $O((\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$ intersections. Therefore, the total number of intersections of this type is $O((\frac{\mathsf{a}}{\mathsf{a}'})^2 n^2 m)$.

For any case, the number of intersections is $O(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$. For $t \in \mathcal{S}$, the number of pairs in $\mathcal{C}(t)$ can be shown by the similar argument: for $t \in \mathcal{S}$, the pairs of $\mathcal{C}(t)$ can be computed by overlaying $\mathcal{T}_f + t$ and $\mathcal{T}_g$, so there are $O(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$ pairs. ◻

If f and g are realistic terrains and $\mathbb{D}_g$ is "larger enough" than $\mathbb{D}_f$, we can show that the bound in Lemma 4 is tight by an example. We assume that $\mathbb{D}_f$

is a square of side length a and $\mathcal{T}_f$ is defined on a regular grid which is rotated slightly in counterclockwise orientation (Figure 4(a)). $\mathcal{T}_f$ consists of $\frac{m}{2}$ grid cells and each cell is triangulated by one of the diagonal edges of the cell. Analogously, we assume that $\mathbb{D}_g$ is a square of side length $\mathsf{a}' > 2\mathsf{a}$ and $\mathcal{T}_g$ is defined on a regular grid which is rotated slightly in clockwise orientation (Figure 4(b)). $\mathcal{T}_g$ consists of $\frac{n}{2}$ grid cells and each cell is triangulated by one of its diagonal edges. For simplicity, we consider the squares of the grids, instead of the triangles. It means that we count the number of combinatorial structures between subsets of $\mathcal{T}_f$ and $\mathcal{T}_g$.

First, let us count the number of intersections made by $\mathcal{T}_g(v)$ and $\mathcal{T}_g(v')$ for vertices $v, v' \in \mathcal{T}_f$. We have $\binom{m}{2} = \Omega(m^2)$ pairs of $\mathcal{T}_g(v)$ and $\mathcal{T}_g(v')$ and each pair makes at least $\Omega(n \times (\frac{\mathsf{a}'-\mathsf{a}}{\mathsf{a}'})^2) = \Omega(n)$ intersections because the overlaying area of $\mathcal{T}_g(v)$ and $\mathcal{T}_g(v')$ is at least $(\mathsf{a}'-\mathsf{a})^2$. So, the number of intersections of this type is $\Omega(nm^2)$.

Next, let us count the number of intersections made by $\mathcal{T}_f(u)$ and $\mathcal{T}_f(u')$ for vertices $u, u' \in \mathcal{T}_g$. We assume that $(\frac{\mathsf{a}}{\mathsf{a}'})^2 n$ is larger than a positive constant i. If $(\frac{\mathsf{a}}{\mathsf{a}'})^2 n$ is smaller than a positive constant i, the bound of the number of cells of $\mathcal{M}$ in Lemma 4 becomes $O(nm(m+i)) = O(nm^2)$ and the number of intersections made by $\mathcal{T}_g(v)$ and $\mathcal{T}_g(v')$ for vertices $v, v' \in \mathcal{T}_f$ is already $\Omega(nm^2)$, we are done. By the assumption $\mathsf{a}' \geq 2\mathsf{a}$, there are $\Omega((\frac{\mathsf{a}'-\mathsf{a}}{\mathsf{a}'})^2 n) = \Omega(n)$ vertices $u \in \mathcal{T}_g$ that satisfies the following condition: for a chosen vertex u, there are $\Omega((\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$ vertices u' of $\mathcal{T}_g$ such that $\mathcal{T}_f(u)$ and $\mathcal{T}_f(u')$ make $\Omega(m)$ intersections. Note that $\mathcal{T}_f(u)$ and $\mathcal{T}_f(u')$ make $\Omega(m)$ intersections if the distance between u and u' is smaller than $\mathsf{a}/2$. It means that for the chosen vertex u, the number of intersections made by $\mathcal{T}_f(u)$ is $\Omega((\frac{\mathsf{a}}{\mathsf{a}'})^2 nm)$. Therefore, the number of intersections of this type is $\Omega((\frac{\mathsf{a}}{\mathsf{a}'})^2 n^2 m)$. In total, the number of vertices of $\mathcal{M}$ is $\Omega((\frac{\mathsf{a}}{\mathsf{a}'})^2 n^2 m + nm^2) = \Omega(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$.

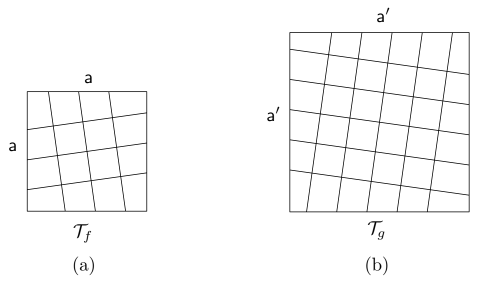

Fig. 4. An example of $\mathcal{T}_f$ and $\mathcal{T}_g$ that induce $\Omega(nm(m+(\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$ different combinatorial structures.

We summarize the results of this section in the following theorem.

Theorem 1. *Let $f : \mathbb{D}_f \to \mathbb{R}$ and $g : \mathbb{D}_g \to \mathbb{R}$ be two realistic terrains such that $\mathcal{T}_f$ (resp. $\mathcal{T}_g$) has m (resp. n) triangles and the side lengths of $\mathbb{D}_f$ (resp.*

$\mathbb{D}_g$) are a and $\mathsf{a}r$ (resp. a' and $\mathsf{a}'r'$) for a positive constant r (resp. r'). The number of different combinatorial structures between $\mathcal{T}_f$ and $\mathcal{T}_g$ is $O(nmA)$, where $A = m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n$. This bound is tight if $\mathsf{a}' > 2\mathsf{a}$.

3 Geometric Matching Algorithms under L_∞ metric

In this section, we first show how to compute $\mathcal{M}$ and the combinatorial structures corresponding to the interiors of the cells. We find an optimal translation vector $t^* \in c$ for each cell c in $\mathcal{M}$ by considering the combinatorial structure corresponding to the interior of c. For two translation vectors t in the interior of c and t' on the boundary of c, every vertex of $\mathcal{O}(f, g, t')$ is an intersection induced by an edge-edge pair or a vertex-triangle pair in $\mathcal{C}(t)$, so it is enough to consider the combinatorial structure corresponding to the interior of c. From now on, the term "combinatorial structure of a cell" means $\mathcal{C}(t)$ for a translation vector t in the interior of the cell.

We observe that the combinatorial structures of the adjacent cells of $\mathcal{M}$ have only $O(1)$ different edge-edge or vertex-triangle pairs, so they can be computed efficiently. We present a deterministic algorithm and a randomized algorithm to compute an optimal translation vector.

3.1 Construction of $\mathcal{M}$

The subdivision $\mathcal{M}$ is constructed by overlaying $\mathcal{T}_f(u)$ and $\mathcal{T}_g(v)$ for all vertices $v \in \mathcal{T}_f$ and $u \in \mathcal{T}_g$ restricted to $\mathcal{S}$ as explained in Section 2.2. The total number of the edges to overlay is $O(nm)$. The overlay of N edges can be constructed in $O(N \log N + K)$ time using $O(N + K)$ space [2], where K is the number of intersections. In our case, $N = O(nm)$ and $K = O(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$ by Theorem 1, so $\mathcal{M}$ can be constructed in $O(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n + \log n))$ time using $O(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$ space.

However, it is not necessary to maintain the whole subdivision $\mathcal{M}$; after finding an optimal translation vector among the translation vectors in a cell of $\mathcal{M}$, the cell is not necessary any more. So we only maintain a small part of $\mathcal{M}$ to reduce the space. We divide $\mathcal{S}$ into a regular grid of length ℓ and then compute cells of the subdivision $\mathcal{M}$ that intersect each of the grid cells, one by one.

For a grid cell c, every triangle of $\mathcal{T}_f(u)$ and $\mathcal{T}_g(v)$ which intersect c can be obtained by maintaining regular grids of length ℓ for $\mathcal{T}_f$ and $\mathcal{T}_g$. From $\mathcal{T}_f(u)$ for vertices $u \in \mathcal{T}_g$, the set of triangles intersecting the grid cell c can be computed in $O((\frac{\ell}{\mathsf{a}'})^2 nm)$ time: there are $O((\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$ copies that have triangles intersecting c and each of such copies has $O((\frac{\ell}{\mathsf{a}})^2 m)$ triangles intersecting c by Lemma 3. Similarly, from $\mathcal{T}_g(v)$ for vertices $v \in \mathcal{T}_f$, the set of triangles intersecting the grid cell c can be computed in $O((\frac{\ell}{\mathsf{a}'})^2 nm)$ time: there are $O(m)$ copies and each of such copies has $O((\frac{\ell}{\mathsf{a}'})^2 n)$ triangles intersecting c by Lemma 3.

Each grid cell is a square of side length ℓ, so the number of cells of $\mathcal{M}$ that intersect a grid cell is $O((\frac{\ell}{\mathsf{a}'})^2 nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$. This bound can be proved by the similar argument in Lemma 4. There are $O((\frac{\ell}{\mathsf{a}'})^2 nm)$ edges to overlay and $O((\frac{\ell}{\mathsf{a}'})^2 nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$ intersections, so these cells of $\mathcal{M}$ for a grid cell can be computed in $O((\frac{\ell}{\mathsf{a}'})^2 nm \log((\frac{\ell}{\mathsf{a}'})^2 nm) + (\frac{\ell}{\mathsf{a}'})^2 nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$ time using $O(n + m + (\frac{\ell}{\mathsf{a}'})^2 nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$ space [2].

If we use a grid with too small ℓ then the number of grid cells that intersect a cell of $\mathcal{M}$ can be arbitrarily large, so it increases time to solve the problem. To avoid that, we set ℓ to be longer than the asymptotic edge lengths of $\mathcal{T}_f$ and $\mathcal{T}_g$, $\Theta(\frac{\mathsf{a}}{\sqrt{m}} + \frac{\mathsf{a}'}{\sqrt{n}})$. With this ℓ, the number of grid cells that intersect a cell of $\mathcal{M}$ is $O(1)$, and the time and space complexities to compute $\mathcal{M}$ cell by cell of the grid become $O(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$ and $O(n + (m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)^2)$, respectively.

Lemma 5. *For two realistic terrains, $\mathcal{M}$ can be reported cell by cell of a grid of length $\ell = \Theta(\frac{\mathsf{a}}{\sqrt{m}} + \frac{\mathsf{a}'}{\sqrt{n}})$ in $O(nmA)$ time using $O(n + A^2)$ space, where $A = m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n$.*

After constructing the cells of $\mathcal{M}$ in a grid cell, we compute the corresponding combinatorial structure of each cell of $\mathcal{M}$. The straightforward way is computing the combinatorial structure of each cell separately, but this is inefficient because the combinatorial structures of the adjacent cells of $\mathcal{M}$ are similar as in the following.

Let c_i and c_j be two adjacent cells of $\mathcal{M}$ with a common edge e, and $\mathcal{C}_i$ and $\mathcal{C}_j$ be the corresponding combinatorial structures, respectively. Without loss of generality, we can say that e is a part of an edge of a triangle in $\mathcal{S}$ defined by a vertex v of $\mathcal{T}_f$ and a triangle $\triangle$ of $\mathcal{T}_g$. Let t be a translation vector in the interior of e. In the overlay of $\mathcal{T}_f + t$ and $\mathcal{T}_g$, $v + t$ lies on the interior of an edge of $\triangle$ and v is the unique vertex that lies on an edge of the overlay of $\mathcal{T}_f + t$ and $\mathcal{T}_g$ by the construction. It means that $\mathcal{C}_i$ and $\mathcal{C}_j$ have at most $O(1)$ different pairs because of the k-low-density assumption (Definition 2(a)) which implies that the degree of each vertex of $\mathcal{T}_f$ and $\mathcal{T}_g$ is at most k.

Observation 1 *The combinatorial structures of two adjacent cells of $\mathcal{M}$ have $k = O(1)$ different pairs.*

Note that the sequence of cells can be obtained by a standard DFS(depth first search) scheme.

3.2 A deterministic geometric matching algorithm

We first consider a deterministic algorithm to compute an optimal translation vector of a cell in $\mathcal{M}$. Computing an optimal translation vector of a cell can be reduced to a linear programming in $\mathbb{R}^4$ as follows. We use w to represent the fourth coordinate of $\mathbb{R}^4$.

For an edge-edge pair (e, e') of a combinatorial structure of a cell c in $\mathcal{M}$, let $(0, 0, v(t, h))$ be a vertical translation vector with respect to $t = (t_x, t_y) \in \mathbb{R}^2$ and $h \in \mathbb{R}$ such that $f[e] + (t_x, t_y, h) + (0, 0, v(t, h))$ and $g[e']$ are contained in a common plane in $\mathbb{R}^3$. For a translation vector $t \in c$, let p be the intersection point of $e + t$ and e'. Then, $|v(t, h)| = |(f(p) + h) - g(p + t)|$. The set of points

$(t_x, t_y, h, |v(t,h)|)$ in $\mathbb{R}^4$ is the upper envelope of two hyperplanes: two sets of points $(t_x, t_y, h, v(t,h))$ and $(t_x, t_y, h, -v(t,h))$ for $t = (t_x, t_y) \in \mathbb{R}^2$ and $h \in \mathbb{R}$. For a vertex-triangle pair of a combinatorial structure, we can construct the upper envelope of two hyperplanes in $\mathbb{R}^4$ analogously.

For a cell c in $\mathcal{M}$ and a translation vector $t \in c$, we construct a set of hyperplanes in $\mathbb{R}^4$ corresponding to the pairs in a combinatorial structure $\mathcal{C}(t)$ as described. The problem of finding an optimal translation vector t for c reduces to the linear programming problem of finding a point q of the smallest w-coordinate in the upper envelope of the hyperplanes in $\mathbb{R}^4$ for $(q_x, q_y) \in c$, where q_x and q_y are the x- and y-coordinates of q, respectively. Note that the restriction $(q_x, q_y) \in c$ can be described by a set of linear inequality constraints. The translation vector (q_x, q_y) is an optimal translation vector for c.

Matoušek and Schwarzkopf [12] gave a dynamic data structure supporting a linear programming in $\mathbb{R}^4$. With N hyperplanes, the data structure can be constructed in $O(N^{4/3+\delta})$ deterministic time and space for any fixed $\delta > 0$. Also, both update time (insertions and deletions of hyperplanes) and query time are $O(N^{1/3+\delta})$ amortized time.

The overall strategy is as follows. We subdivide $\mathcal{S}$ by a regular grid of length $\ell = \Theta(\frac{\mathsf{a}}{\sqrt{m}} + \frac{\mathsf{a}'}{\sqrt{n}})$ and treat the cells one by one. For a grid cell, we compute the corresponding part of $\mathcal{M}$ and a sequence of adjacent cells in $\mathcal{M}$ as described in Section 3.1. Next, we build the data structure [12] with hyperplanes in $\mathbb{R}^4$ for the combinatorial structure of the first cell of the sequence. We find the lowest point in the upper envelope of the hyperplanes. For the rest cells in the sequence, we update the data structure for the corresponding combinatorial structure, and then find the lowest point. Note that the number of updates for a cell is $O(1)$ by Observation 1 if we follow the sequence of adjacent cells. We repeat this until the end of the sequence. By Lemma 4, the number of edge-edge and vertex-triangle pairs in a combinatorial structure is $O(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$ and the number of different combinatorial structure is $O(nm(m+(\frac{\mathsf{a}}{\mathsf{a}'})^2 n))$. The following theorem summarizes the overall time and space complexity.

Theorem 2. *Let $f : \mathbb{D}_f \to \mathbb{R}$ and $g : \mathbb{D}_g \to \mathbb{R}$ be the two realistic terrains such that $\mathcal{T}_f$ (resp. $\mathcal{T}_g$) has m (resp. n) triangles and the side lengths of $\mathbb{D}_f$ (resp. $\mathbb{D}_g$) are a and $\mathsf{a}r$ (resp. a' and $\mathsf{a}'r'$) for a positive constant r (resp. r'). We can compute an optimal translation vector t^* such that $\mathbb{D}_f + t^* \subset \mathbb{D}_g$ and the distance under L_∞ metric between f and $g\restriction_{\mathbb{D}^*}$ is minimized in $O(nmA^{4/3+\delta})$ time with $O(n + A^2)$ space for any fixed $\delta > 0$ and $A = m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n$.*

3.3 A randomized geometric matching algorithm

In this section, we propose a randomized algorithm for the reduced problem described in Section 3.2. We first present a decision algorithm to decide the existence of a point of the upper envelope of hyperplanes in $\mathbb{R}^4$ such that the w-coordinates of the point is smaller than an input value. Next, we propose a randomized approach to compute an optimal translation vector.

As described in Section 3.2, we construct the set of hyperplanes in $\mathbb{R}^4$ corresponding to the combinatorial structure of a cell c of $\mathcal{M}$ and the set of hyperplanes corresponding to the boundary edges of c. A decision version of finding the lowest point in the upper envelope of hyperplanes can be stated as follows: given $\delta > 0$, is there a point in the upper envelope of hyperplanes whose w-coordinate is smaller than δ?

We check whether there is such a point in the upper envelope by introducing a new hyperplane $H_\delta : w = \delta$. If the upper envelope has such a point then the intersections of H_δ and each 'upper half-space' of the hyperplanes have a non-empty common intersection in H_δ. This problem can be reduced to a linear programming in $\mathbb{R}^3$.

Eppstein [6] gave a semi-online data structure for a linear programming in $\mathbb{R}^3$. The data structure supports update and query in $O(\log^2 N \log^2 \log N)$ time if one knows the updating sequence of length N in advance. Again, we subdivide $\mathcal{S}$ by a regular grid of length $\ell = \Theta(\frac{\mathsf{a}}{\sqrt{m}} + \frac{\mathsf{a}'}{\sqrt{n}})$ and treat cell by cell of the grid. For a grid cell, we compute the corresponding part of $\mathcal{M}$ and a sequence of adjacent cells in $\mathcal{M}$ as described in Section 3.1. We can construct the update sequence of half-spaces for a grid cell and the length of the update sequence is $O((\frac{\ell}{\mathsf{a}'})^2 nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)) = O((m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)^2)$.

Lemma 6. *For given $\delta > 0$, we can solve the decision problem for each cell of $\mathcal{M}$ in $O(\log^2 A \log^2 \log A)$ amortized time where $A = m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n$. Therefore, we can find translation vectors t such that $d_\infty(f, g\restriction_{\mathbb{D}_f + t}) \leq \delta$ in $O(nmA \log^2 A \log^2 \log A)$ time.*

Lemma 6 means that for a given threshold value, we can find subdomains $\mathbb{D}$ of $\mathbb{D}_g$ such that g defined on $\mathbb{D}$ is "similar enough" to f. Next, we present how to use the solutions of the decision problems to find an optimal translation vector. First we randomly choose one cell c of $\mathcal{M}$ and compute the minimum distance δ between f and g for the combinatorial structure of c by solving a linear programming [11]. With this δ, we solve the decision problem for each cells of $\mathcal{M}$ and find cells which realize the distance smaller than δ. We can expect that only constant fraction of the number of the cells realize the distance smaller than δ. We repeat this procedure recursively with a new distance δ' computed from one of the cells decided as 'yes' from the previous recursive step until we find a cell which realizes the minimum distance. The expected number of recursion is $O(\log(\#\text{cells})) = O(\log(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n))) = O(\log n)$. We summarize this section in the following theorem.

Theorem 3. *Let $f : \mathbb{D}_f \to \mathbb{R}$ and $g : \mathbb{D}_g \to \mathbb{R}$ be the two realistic terrains such that $\mathcal{T}_f$ (resp. $\mathcal{T}_g$) has m (resp. n) triangles and the side lengths of $\mathbb{D}_f$ (resp. $\mathbb{D}_g$) are a and $\mathsf{a}r$ (resp. a' and $\mathsf{a}'r'$) for a positive constant r (resp. r'). We can compute an optimal translation vector t^* such that $\mathbb{D}_f + t^* \subset \mathbb{D}_g$ and the distance under L_∞ metric between f and $g\restriction_{\mathbb{D}^*}$ is minimized in $O(nmA \log n \log^2 A \log^2 \log A)$ expected time with $O(n + A^2)$ space where $A = m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n$.*

4 Geometric Matching Algorithm under L_1 metric

In this section, we solve the terrain matching problem under L_1 metric. To compute the distance between two terrains under L_1 metric, we need to compute the volume function between two terrains. The volume function of two terrains is a sum of volume functions between pairs of triangles, so we need to compute the volume functions between pairs of triangles.

Lemma 1 shows that the distance between two terrains under L_∞ metric is decided by an edge-edge pair or a vertex-triangle pair. Let t be a translation vector in $\mathcal{S}$ and $\mathcal{C}(t)$ be the corresponding combinatorial structure. From the edge-edge and vertex-triangle pairs of $\mathcal{C}(t)$, we can determine intersecting pairs of triangles $(\triangle, \triangle')$ where $\triangle \in \mathcal{T}_f$ and $\triangle' \in \mathcal{T}_g$. Note that only these pairs of triangles can have non-zero volume between $f[\triangle]$ and $g[\triangle']$. Therefore, we can use $\mathcal{M}$ and the corresponding combinatorial structures to get different pairs of triangles that can have non-zero volume between them. Lemma 4 shows that there are $O(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$ edge-edge and vertex-triangle pairs in a combinatorial structure. It means that there are $O(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$ triangle pairs that realize non-zero volume between them for a combinatorial structure.

However, for a fixed two-dimensional translation vector t, the volume function between a pair of triangles is not described as a single formula. The example below shows that for a fixed set of edge-edge and vertex-triangle pairs, there are many different formulae for the volume function between a pair of triangles. To

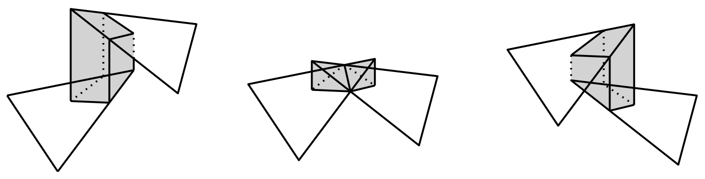

Fig. 5. Three different formulae for the volume function between two triangles.

get a single formula for the volume function, we need to consider the amount of vertical translation h of f along with the translation vector $t = (t_x, t_y) \in \mathcal{S}$. If we concatenate h to the translation vector $t = (t_x, t_y)$ as a third coordinate, then a vertical prism $\{(t_x, t_y, h) | h \in \mathbb{R}\}$ in $\mathbb{R}^3$ over each cell of $\mathcal{M}$ can be seen as all possible vertical translations for the pairs of triangles corresponding to the combinatorial structure of the cell. We subdivide the vertical prisms such that the volume function between each pair of triangles is described as a single formula within a cell of the subdivision.

Each pair of triangles $(\triangle, \triangle')$ divides the vertical prism into three parts. Each of the three parts corresponds to the relative position of two triangles $f[\triangle]$ and $g[\triangle']$: $f[\triangle]$ lies above $g[\triangle']$, $f[\triangle]$ lies below $g[\triangle']$, and $f[\triangle]$ intersects $g[\triangle']$. These three parts in the prism is defined by two planes in $\mathbb{R}^3$. Therefore, the desired subdivision of a vertical prism is a subdivision of $O(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)$ planes in $\mathbb{R}^3$ restricted to the vertical prism. The number of cells in a vertical prism is $O((m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)^3)$ and it can be computed in the same time [5].

For the three-dimensional translation vectors (t_x, t_y, h) in each cell of the vertical prism, the volume function between a pair of triangles is determined as a single formula which is a constant degree polynomial of t_x, t_y, and h, and the sum of the formulae is also a constant-degree polynomial of t_x, t_y, and h. For each cell of the vertical prism, a three-dimensional translation vector that realize the minimum volume can be found by using numerical methods in $O(|C|)$ time, where $|C|$ is the complexity of the cell. Also, for two adjacent cells of the vertical prism, only $O(1)$ pairs of triangles have different formulae. So the formula of the volume function of an adjacent cell can be derived in $O(1)$ time. There are $O(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)^4)$ cells and their total complexity is also $O(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)^4)$. Therefore, the translation vector $t \in \mathcal{S}$ and the amount of vertical translation h that minimize the L_1 distance between f and $g\restriction_{\mathbb{D}_f + t}$ can be computed in $O(nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)^4)$ time.

To reduce the space complexity, we apply the same approach used in Section 3.1. For $\ell = \Theta(\frac{\mathsf{a}}{\sqrt{m}} + \frac{\mathsf{a}'}{\sqrt{n}})$, we subdivide $\mathcal{S}$ by a regular grid of length ℓ and treat the subdivision $\mathcal{M}$ cell by cell of the grid. For a grid cell, we compute corresponding part of $\mathcal{M}$. For the cells of $\mathcal{M}$ in the grid cell, we maintain only one vertical prism. So, the space complexity is bounded by $O(n + (\frac{\ell}{\mathsf{a}'})^2 nm(m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n) + (m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)^3) = O(n + (m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n)^3)$. We summarize this subsection in the following Theorem.

Theorem 4. *Let $f : \mathbb{D}_f \to \mathbb{R}$ and $g : \mathbb{D}_g \to \mathbb{R}$ be the two realistic terrains such that $\mathcal{T}_f$ (resp. $\mathcal{T}_g$) has m (resp. n) triangles and the side lengths of $\mathbb{D}_f$ (resp. $\mathbb{D}_g$) are a and $\mathsf{a}c$ (resp. a' and $\mathsf{a}'c'$) for a positive constant c (resp. c'). We can compute an optimal translation vector t^* such that $\mathbb{D}_f + t^* \subset \mathbb{D}_g$ and the distance under L_1 metric between f and $g\restriction_{\mathbb{D}^*}$ is minimized in $O(nmA^4)$ time using $O(n + A^3)$ space, where $A = m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n$.*

5 Experiment

Computing an optimal translation vector of a cell can be reduced to the linear programming mentioned in Section 3.2. In our implementation of the geometric matching algorithm under L_∞ metric, we maintain combinatorial structures and compute an optimal translation vector for each cell separately. We used an algorithm to solve the linear programming that runs in expected linear time to the number of the planes [11][1]. While solving the linear programming in a cell, we stop the procedure when the solution becomes larger than the best solution of cells that already passed. The worst-case time complexity of this implementation is expected $O(nmA^2)$ as the number of constraints of each linear programming is $O(A)$, where $A = m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n$.

[1] A dynamic data structure [12][6] is not used due to technical difficulties.

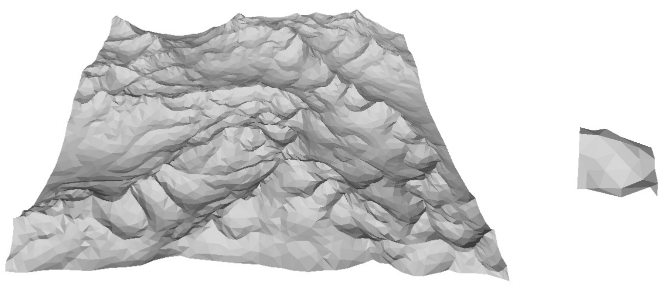

Fig. 6. The input domain terrain and the input patch terrain ($n = 8000, m = 40$).

Experimental settings

The inputs of our experiments are a relatively large terrain and a small patch terrain. The number of triangles of the domain terrain is n, and that of the patch terrain is m. In our experiment, synthetic terrains are used to be inputs. The terrains are first generated using a ridged multifractal terrain model [4] with standard parameters, and simplified by quadric based edge collapse decimation [8] to get the specific numbers of triangles. The ridged multifractal terrain model is known to well describe a natural environment contains plains, foothills, and mountains, all in one construction as in Figure 6.

Table 1. Related measurements of domain terrains

Parameter, Measurement	Value
The number of triangles	2000 4000 6000 8000
Minimum edge-length (a)	2.00 1.13 1.03 0.87
Maximum edge-length (b)	27.93 23.77 18.22 14.81
Size of the domain	256.12×256.27

Table 2. Related measurements of patch terrains

Parameter, Measurement	Value
The number of triangles	10 20 30 40 60
Minimum edge-length (a)	6.94 2.97 3.25 2.55 2.17
Maximum edge-length (b)	20.24 17.90 15.88 13.62 11.94
Size of the domain	26.86×25.93

The experiments are done for each pair of five different patch terrains and four different domain terrains with different number of triangles. We measure those terrains to get values that mainly related to the realistic assumptions. The values are shown in Table 1 and Table 2. The domain terrains have the same domain and so the patch terrains: the domain terrains have the rectangular domain of size 256.12×256.27, and the patch terrains have the rectangular domain of size 26.86×25.93. Also, the lengths of the edges tend to decrease as the number of triangles increases in a terrain. It means that the number of triangles represents the density of the triangulation.

We carry out our experiments with Intel Core i5-4590 CPU at 3.30GHz and 8GB DDR3 memory, and the algorithms are coded in C++.

Experimental results

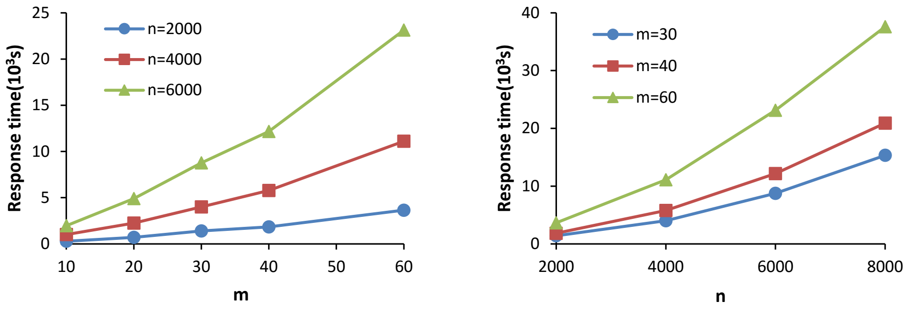

Fig. 7. Effect of triangulation density on response time.

We show the response times of our algorithm over varying m and n in Figure 7. The response times do not increase rapidly even the worst case time complexity of this implementation is expected $O(nmA^2)$ time, where $A = m + (\frac{\mathsf{a}}{\mathsf{a}'})^2 n$. We observe that the results show trends similar to the number of faces of $\mathcal{M}$ in Figure 8

The asymptotic time to construct $\mathcal{M}$ is less than that of the rest as described in Section 3. The ratio of constructing $\mathcal{M}$ decreases when m and n increases as shown in Figure 9.

6 Conclusion

In this paper, we study the problem of finding a part of a big terrain which is the most similar to a given small terrain. We propose a subdivision which can be used to find a candidate set. Also we analyse its complexity when two terrains satisfy the realistic assumption. Based on the subdivision, we suggest

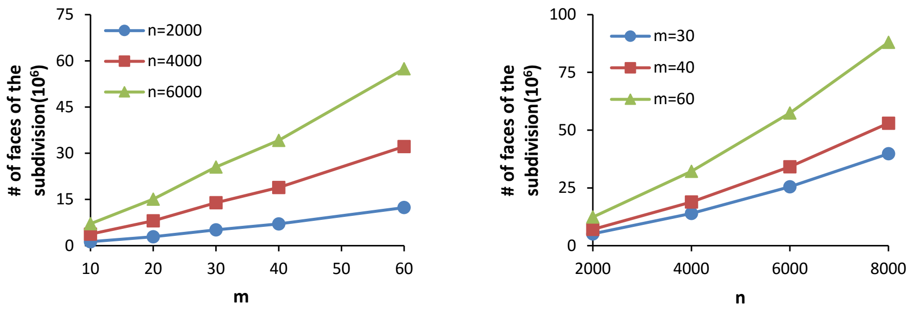

Fig. 8. Effect of triangulation density on the complexity of $\mathcal{M}$.

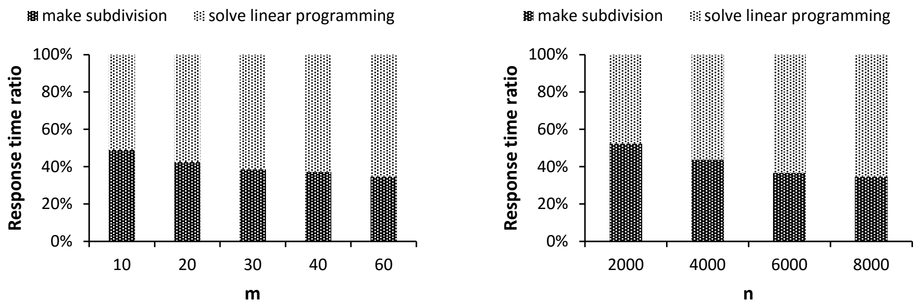

Fig. 9. Break down of the response time ratio.

two algorithms to solve the terrain matching problem under L_∞ metric. One can choose one of the algorithms depending on the size of two terrains. Also, we extend each cell of $\mathcal{M}$ to a vertical prism and subdivide it to get a fixed formula of the volume between two terrains. With this extended subdivision, we solve the terrain matching problem under L_1 metric.

For the further researches, we can consider the same problem with the generalization of the distance measures to the L_p metric for $p > 1$ or the same problem allowing rotation of terrains around z-axis. We expect that our construction of the subdivision $\mathcal{M}$ and its extension can be a good starting point for the future works.

References

1. Agarwal, P.K., Aronov, B., van Kreveld, M., Löffler, M., Silveira, R.I.: Computing correlation between piecewise-linear functions. SIAM Journal on Computing 42(5), 1867–1887 (2013)
2. Balaban, I.J.: An optimal algorithm for finding segments intersections. In: Proceedings of the 11th Symposium on Computational Geometry. pp. 211–219 (1995)

3. Carr, J.R., Sobek, J.S.: Digital scene matching area correlator (DSMAC). In: Proceedings on the Society of Photo-Optical Instrumentation Engineers. vol. 0238, pp. 36–41 (1980)
4. Ebert, D.S., Musgrave, F.K., Peachey, D., Perlin, K., Worley, S.: Texturing and Modeling: A Procedural Approach. Morgan Kaufmann Publishers Inc. (2002)
5. Edelsbrunner, H., O'Rourke, J., Seidel, R.: Constructing arrangements of lines and hyperplanes with applications. SIAM Journal on Computing 15(2), 341–363 (1986)
6. Eppstein, D.: Dynamic three-dimensional linear programming. ORSA Journal on Computing 4(4), 360–368 (1992)
7. Ernst, M.D., Flinchbaugh, B.E.: Image/map correspondence using curve matching. In: Proceedings on AAAI Symposium on Robot Navigation. pp. 15–18 (1989)
8. Garland, M., Heckbert, P.S.: Surface simplification using quadric error metrics. In: Proceedings of the 24th annual conference on Computer graphics and interactive techniques. pp. 209–216. ACM Press/Addison-Wesley Publishing Co. (1997)
9. Golden, J.P.: Terrain contour matching (TERCOM): A cruise missile guidance aid. In: Proceedings on the Society of Photo-Optical Instrumentation Engineers. vol. 0238, pp. 10–18 (1980)
10. Johnson, A.E., Ansar, A., Matthies, L.H., Trawny, N., Mourikis, A.I., Roumeliotis, S.I.: A general approach to terrain relative navigation for planetary landing (2007)
11. Matoušek, J., Sharir, M., Welzl, E.: A subexponential bound for linear programming. Algorithmica 16(4-5), 498–516 (1996)
12. Matoušek, J., Schwarzkopf, O.: Linear optimization queries. In: Proceedings of the Eighth Annual Symposium on Computational Geometry. pp. 16–25. SCG '92, ACM (1992)
13. Medioni, G., Nevatia, R.: Matching images using linear features. Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-6(6), 675–685 (1984)
14. Moet, E., van Kreveld, M., van der Stappen, A.F.: On realistic terrains. Computational Geometry 41(1??), 48 – 67 (2008)
15. Moroz, G., Aronov, B.: Computing the distance between piecewise-linear bivariate functions. In: Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 288–293. SODA '12, SIAM (2012)
16. Newman, P., Durrant-Whyte, H.: Using sonar in terrain-aided underwater navigation. In: Proceedings on IEEE International Conference on Robotics and Automation. vol. 1, pp. 440–445 (1998)
17. Olson, C., Matthies, L.: Maximum likelihood rover localization by matching range maps. In: Proceedings on IEEE International Conference on Robotics and Automation. vol. 1, pp. 272–277 (1998)
18. Rodriquez, J.J., Aggarwal, J.K.: Matching aerial images to 3-d terrain maps. IEEE Transactions on Pattern Analysis and Machine Intelligence 12(12), 1138–1149 (1990)
19. Sistiaga, M., Opderbecke, J., Aldon, M., Rigaud, V.: Map based underwater navigation using a multibeam echosounder. In: Proceedings on OCEANS. vol. 2, pp. 747–751 (1998)
20. Williams, S., Dissanayake, G., Durrant-Whyte, H.: Towards terrain-aided navigation for underwater robotics. Advanced Robotics 15(5), 533–549 (2001)