Journal of Discrete Algorithms 8 (2010) 214–230

Contents lists available at ScienceDirect

Journal of Discrete Algorithms

www.elsevier.com/locate/jda

Approximation algorithms for shortest descending paths in terrains

Mustaq Ahmed [a,*], Sandip Das [b], Sachin Lodha [c], Anna Lubiw [a,1], Anil Maheshwari [d,1], Sasanka Roy [c]

[a] *David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, N2L 3G1, Canada*
[b] *Indian Statistical Institute, Kolkata, India*
[c] *Tata Consultancy Services Ltd., Pune, India*
[d] *School of Computer Science, Carleton University, Ottawa, ON, K1S 5B6, Canada*

A R T I C L E   I N F O

Article history:
Received 19 December 2008
Accepted 11 May 2009
Available online 14 May 2009

Keywords:
Descending path
Gently descending path
Shortest path
Approximation algorithm
Terrain
Computational geometry

A B S T R A C T

A path from s to t on a polyhedral terrain is *descending* if the height of a point p never increases while we move p along the path from s to t. No efficient algorithm is known to find a *shortest descending path* (*SDP*) from s to t in a polyhedral terrain. We present two approximation algorithms that solve the SDP problem on general terrains. We also introduce a generalization of the shortest descending path problem, called the *shortest gently descending path* (*SGDP*) *problem*, where a path descends, but not too steeply. The additional constraint to disallow a very steep descent makes the paths more realistic in practice. We present two approximation algorithms to solve the SGDP problem on general terrains. All of our algorithms are simple, robust and easy to implement.

© 2009 Elsevier B.V. All rights reserved.

1. Introduction

Finding a shortest path between two points in a geometric domain is one of the most fundamental problems in computational geometry. One extensively-studied version of the problem is to compute a shortest path on a polyhedral terrain; this has many applications in robotics, industrial automation and Geographic Information Systems. Our paper is about a variant of this problem for which no efficient algorithm is known, the *Shortest Descending Path* (*SDP*) *problem*: given a polyhedral terrain, and points s and t on the surface, find a shortest path on the surface from s to t such that, as a point travels along the path, its elevation, or z-coordinate, never increases. We need to compute a shortest descending path, for example, for laying a canal of minimum length from the source of water at the top of a mountain to fields for irrigation purpose, and for skiing down a mountain along a shortest route [5,25]. In this paper we present fully polynomial time approximation schemes to find SDPs in general terrains.[2]

Furthermore, in many applications of SDPs, we want a path that descends, but *not too steeply*. For example, when we ski down a mountain we avoid a too steep descent. In such cases, a very steep segment of a descending path should be replaced by "switchbacks" that go back and forth at a gentler slope, like the hairpin bends on a mountain road (see Fig. 1). The *Shortest Gently Descending Path* (*SGDP*) *problem* combines the SDP problem with the problem of finding shortest *anisotropic* paths, where there are different costs associated with different travel directions in a face. Our constraint that forbids a steep descent is an anisotropic constraint. At first glance it would seem that the entire SGDP problem is a special case of anisotropic paths, but results on anisotropic paths assume (e.g., Sun and Reif [31]) that there is a feasible path

[*] Corresponding author.
E-mail addresses: m6ahmed@uwaterloo.ca (M. Ahmed), sandipdas@isical.ac.in (S. Das), sachin.lodha@tcs.com (S. Lodha), alubiw@uwaterloo.ca (A. Lubiw), anil@scs.carleton.ca (A. Maheshwari), sasanka.roy@tcs.com (S. Roy).
[1] Research supported by NSERC.
[2] Preliminary versions of our SDP algorithms appeared in Ahmed and Lubiw [2] and Roy et al. [26].

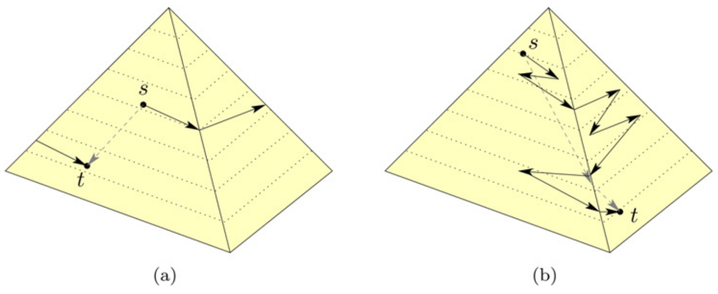

Fig. 1. Descending gently towards a steep direction.

between any two points in a common face, a property that is violated when ascending paths are forbidden. In this paper we combine techniques used for SDPs and for anisotropic paths and present two fully polynomial time approximation schemes to solve the SGDP problem on a general terrain.[3]

1.1. Related work

The SDP problem was introduced by de Berg and van Kreveld [13], who gave an algorithm to preprocess a terrain in $O(n \log n)$ time so that it can be decided in $O(\log n)$ time if there exists a descending path between any pair of vertices. They did not consider the length of the path, and left open the problem of finding the shortest such path. Roy, Das and Nandy [25] solved the SDP problem for two special classes of terrains. For convex (or concave) terrains, they use the *continuous Dijkstra approach* [19] to preprocess the terrain in $O(n^2 \log n)$ time and $O(n^2)$ space so that an SDP of size k can be determined in $O(k + \log n)$ time. For a terrain consisting of edges parallel to one another, they find an SDP in $O(n \log n)$ time by transforming selected faces of the terrain in a way that makes the unfolded SDP a straight line segment. Roy [24] has recently improved this running time to $O(n)$, by replacing a sorting step in the previous algorithm with a divide-and-conquer technique. Ahmed and Lubiw [5] examined the basic properties of SDPs that show the similarities and the dissimilarities between SDPs and shortest paths, and indicated why a shortest path algorithm like the continuous Dijkstra approach cannot be used directly to solve the SDP problem on general terrains. They also gave an $O\left(n^{3.5} \log\left(\frac{1}{\epsilon}\right)\right)$ time algorithm that finds a $(1+\epsilon)$-approximate SDP through a *given* sequence of faces. Their algorithm first formulates the problem as a convex optimization problem, which is then solved using a standard technique in convex programming. In a more recent work Ahmed and Lubiw [3] gave a full characterization of the bend angles of an SDP, which shows that the bend angles along an SDP follow a generalized form of Snell's law of refraction of light. This result implies that computing an exact SDP is not easy even when we know the sequence of faces used by the SDP, due to the numerical issues similar to the ones arising in the Weighted Region Problem [20, Section 8].

Exact algorithms for shortest obstacle-avoiding paths in 2D and shortest paths on 3D polyhedral surfaces rely on two "elementary" approaches: the continuous Dijkstra approach, and the sequence tree approach of Chen and Han [11]. One variant of the continuous Dijkstra approach that has received recent attention utilizes a quad-tree like subdivision of the plane to achieve faster running time [17]. Schreiber and Sharir [28] have extended this variant for a 3D convex surface, which has later been applied to several realistic scenarios [27]. All these algorithms depend on the fact that it is easy to trace a shortest path onward, which is not the case with SDPs in a general terrain.

It was Papadimitriou [22] who first introduced the idea of discretizing space by adding *Steiner points* and approximating a shortest path through the space by a shortest path in the graph of Steiner points. He did this to find a shortest obstacle-avoiding path in 3D—a problem for which computing an exact solution is NP-hard [10]. On polyhedral surfaces, the Steiner point approach has been used in approximation algorithms for many variants of the shortest path problem, particularly those in which the shortest path does not unfold to a straight line segment. One such variant is the Weighted Region Problem [20]. In this problem, a set of constant weights is used to model the difference in costs of travel in different regions on the surface, and the goal is to minimize the weighted length of a path. Mitchell and Papadimitriou [20] gave an approximation algorithm that solves this problem in $O\left(n^8 \log\left(\frac{n}{\epsilon}\right)\right)$ time using the continuous Dijkstra approach. Following their result, several faster approximation schemes [7–9,12,32] have been devised, all using the Steiner point approach. The Steiner points are placed along the edges of the terrain, except that Aleksandrov et al. [9] place them along the bisectors of the face angles. A comparison between these algorithms can be found in [9].

One generalization of the Weighted Region Problem is finding a shortest anisotropic path [23], where the weight assigned to a region depends on the direction of travel. The weights in this problem capture, for example, the effect of gravity and friction on a vehicle moving on a slope. All the papers on this problem used the Steiner point approach [12,18,29,31]. As we mentioned before, these algorithms for anisotropic paths assume that every face f is *totally traversable*, i.e., there is a feasible path from any point to any other point in f. To be precise, Cheng et al. [12] assumed that the (anisotropic) weight associated with a direction of travel is bounded by constants from both above and below, thus any direction of travel is

[3] Preliminary versions of our SGDP algorithms appeared in Ahmed et al. [6].

feasible. (Moreover, the algorithm of Cheng et al. is for a subdivision of the plane, not for a terrain.) The rest of the papers [18,29,31] used the anisotropic weight model of Rowe and Ross [23] which allows switchback paths to "cover" any direction in f. The assumption that every face is totally traversable allows placing Steiner points in a face *independently from all other faces*. Sun and Reif [31, Section V] relaxed this assumption but only in isolated faces. Thus, they can still rely on independent placement of Steiner points in a face. For both the SDP and the SGDP problems, ascending directions are unreachable in *every* face, which necessitates the use of a non-local strategy of placing Steiner points, as we will see in this paper.

To compute a shortest path in a graph of Steiner points in a terrain we use a variant of Dijkstra's algorithm developed by Sun and Reif [30]. Their algorithm, called the Bushwhack algorithm, achieves faster running time by utilizing certain geometric properties of the paths in such a graph. The algorithm has been used in shortest path algorithms for the Weighted Region Problem [9,32] and the Shortest Anisotropic Path Problem [31].

Other approximation techniques for shortest paths on surfaces include variants of the Steiner point approach (e.g., one that places Steiner points away from the surface in order to have a smaller graph [1]) as well as a few completely different approaches (e.g., one using an additive weight Voronoi diagram to approximate distances [16]). We do not explore these techniques in this paper.

1.2. New results

We model each of the SDP and SGDP problems in a terrain as a shortest path problem in a directed graph. The nodes in the graph correspond to Steiner points that are added along the edges of the terrain, with directed edges from higher to lower points in a common face, and edge weights corresponding to the descending distances. We present two algorithms (more precisely, fully polynomial time approximation schemes) each for SDP and SGDP problems; the algorithms are simple, robust and easy to implement. For each problem, the two algorithms differ only in the placement of Steiner points—the first one places Steiner points evenly on the edges, whereas the second one places them in a geometric progression. We measure the performance of these algorithms in terms of the number n of vertices of the terrain, the largest degree d of a vertex, the desired approximation factor ϵ, the length L of the longest edge, the smallest distance h of a vertex from a non-incident edge in the same face, the largest acute angle θ between a non-level edge and a vertical line, and the angle of steepness ψ (see Section 2.1 for details).

The first algorithm for SDP (or SGDP) is faster in terms of n, ϵ and $\frac{L}{h}$, but it depends on the inclination of the non-level edges. On the other hand, the second algorithm for SDP (or SGDP) does not depend at all on edge inclinations, and hence is better for terrains with almost level edges. It is straightforward to follow a "hybrid" approach that first checks the edge inclinations of the input terrain, and then runs whichever of these two algorithms ensures a better running time for that particular terrain.

Our results on the SDP problem are as follows:

1. Given a vertex s, we place Steiner points evenly along terrain edges, so that after an $O\left(\frac{n^2X}{\epsilon}\log\left(\frac{nX}{\epsilon}\right)\right)$-time preprocessing phase, we can determine a $(1+\epsilon)$-approximate SDP from s to any point v in $O(n)$ time if v is either a vertex of the terrain or a Steiner point, and in $O\left(\frac{nX}{\epsilon}\right)$ time otherwise, where $X = \frac{L}{h\cos\theta}$.
2. The second algorithm places Steiner points in a geometric progression along edges, and modifies the above preprocessing time and the two query times to $O\left(\frac{n^2X'}{\epsilon}\log^2\left(\frac{nX'}{\epsilon}\right)\right)$, $O(n)$, and $O\left(\frac{nX'}{\epsilon}\log\left(\frac{nX'}{\epsilon}\right)\right)$, respectively, where $X' = \frac{L}{h}$.

Our results on the SGDP problem are as follows:

1. Given a vertex s, we place Steiner points evenly along terrain edges, so that after an $O\left(\frac{n^2X}{\epsilon}\log\left(\frac{nX}{\epsilon}\right)\right)$-time preprocessing phase, we can determine a $(1+\epsilon)$-approximate SGDP from s to any point v in $O(nd)$ time if v is either a vertex of the terrain or a Steiner point, and in $O\left(n\left(d+\frac{X}{\epsilon}\right)\right)$ time otherwise, where $X = \frac{L}{h\cos\theta\cos\psi}$.
2. The second algorithm places Steiner points in a geometric progression along edges, and modifies the above preprocessing time and the two query times to $O\left(\frac{n^2X'}{\epsilon}\log^2\left(\frac{nX'}{\epsilon}\right)\right)$, $O(nd)$, and $O\left(nd+\frac{nX'}{\epsilon}\log\left(\frac{nX'}{\epsilon}\right)\right)$, respectively, where $X' = \frac{L}{h\cos\psi}$.

1.3. Organization of the paper

The paper is organized as follows. In Section 2 we define a few terms and discuss the placement of Steiner points. Section 3 gives details of our approximation algorithms for the SDP problem. In Section 4, we establish a few properties of an SGDP and then give our approximation algorithms for the SGDP problem. We conclude in Section 5 with a few open problems.

2. Preliminaries

2.1. Terminology

A terrain is a 2D surface in 3D space with the property that every vertical line intersects it in at most one point [14]. We consider triangulated terrains. For any point p in the terrain, $h(p)$ denotes the height of p, i.e., the z-coordinate of p.

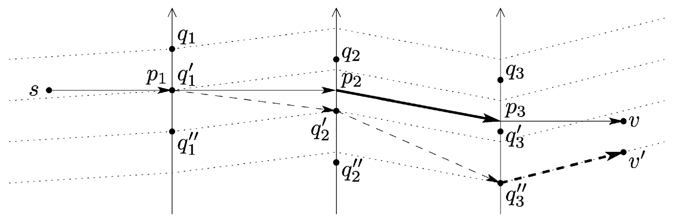

Fig. 2. Problems with independently-placed Steiner points.

We assume without loss of generality that all points of the terrain lie above the plane $z = 0$. An edge or face in 3D is *level* if all points on that edge or face have the same height. We add s as a vertex of the terrain. Let n be the number of vertices in the terrain. By Euler's formula, the terrain has at most $3n$ edges, and at most $2n$ faces.

We reserve the terms "edge" and "vertex" for features of the terrain. We use the term "segment" to denote a line segment of a path, and "node" to denote an endpoint of a segment. We use "node" and "link" to mean the corresponding entities in a graph or a tree. In our figures, an arrow with a solid, dark arrowhead denotes a path segment, and the arrow may be heavy to mark a level segment. In the figures where the direction of the edges are important, we again use arrows to mark the upward direction, but we make the arrowheads V-shaped ("open") in this case to differentiate the edges from the segments. Dotted lines are used to show level lines in a face.

A path P from s to t on the terrain is *descending* if the z-coordinate of a point p never increases while we move p along the path from s to t. We assume that all paths and segments in our discussion are directed. Given an angle $\psi \in [0, \frac{\pi}{2})$, a line segment pq is *steep* if it makes an angle less than ψ with a vertical line. A path P is *gently descending* if P is descending, and no segment of P is steep. A downward direction in a face is called a *critical direction* if the direction makes an angle equal to ψ with a vertical line. (Note that only a *downward* direction can be a critical direction, although both upward and downward directions can be steep.) A gently descending path is called a *critical path* if each of its segments is in a critical direction. A critical path may travel through more than one face; inside a face it will zigzag back and forth. We would like to replace steep descending segments by critical paths. This is sometimes possible, e.g. for a steep segment starting and ending at points interior to a face, but is not possible in general. The details are in Lemma 4.3, which uses the following terms. A vertex v in face f is *locally sharp in f* if v is either the higher endpoint of two steep edges or the lower endpoint of two steep edges of f. A vertex v is *sharp* if it is locally sharp in all its incident faces. Note that a sharp vertex is either the higher endpoint of all the edges incident to it, or the lower endpoint of all such edges. (A sharp vertex of the first type is like a pinnacle from which you cannot descend gently.)

2.2. Placing the Steiner points

Our approximation algorithms work by first discretizing the terrain with many Steiner points along the edges, and then determining a shortest path in a directed graph in which each link connects a pair of vertices or Steiner points in a face of the terrain in the descending (more accurately, in the non-ascending) direction. Although the idea is similar to other Steiner point approaches discussed in Section 1.1, there are two aspects of descending paths that make our approach quite different from previous Steiner point approaches. We will discuss the issue below for the SDP problem. Note that the discussion immediately applies to the SGDP problem.

First, because of the nature of these problems, we have to position the Steiner points quite differently from the Steiner point approaches discussed in Section 1.1. In particular, we cannot place Steiner points in an edge without considering the heights of the Steiner points in other edges. More elaborately, for each Steiner point p in an edge, if there is no Steiner point with height $h(p)$ in other edges of the neighboring faces, it is possible that a descending path from s to v through Steiner points does not exist, or is arbitrarily longer than the SDP. For example, consider the SDP $P = (s, p_1, p_2, p_3, v)$ in Fig. 2, where for each $i \in [1, 3]$, q_i, q_i' and q_i'' are three consecutive Steiner points with $h(q_i) > h(q_i') > h(q_i'')$ such that q_i is the nearest Steiner point above p_i. Note that in this figure the faces have been unfolded onto a plane, and that p_1 and q_1' are the same point. There is no descending path from s to v through the Steiner points: we must cross the first edge at q_1' or lower, then cross the second edge at q_2' or lower, and cross the third edge at q_3'' or lower, which puts us at a height below $h(v)$. Another important observation is that even if a descending path exists, it may not be a good approximation of P. In Fig. 2, for example, if we want to reach a point v' slightly below v, $P' = (s, p_1, q_2', q_3'', v')$ would be a feasible path, but the last intermediate nodes of P and P' are not very close. We can easily extend this example to an SDP P going through many edges such that the "nearest" descending path P' gets further away from P at each step, and at one point, P' starts following a completely different sequence of edges. Clearly, we cannot ensure a good approximation by just making the Steiner points on an edge close to each other.

To guarantee the existence of a descending path through Steiner points that approximates an SDP from s to any vertex, we have to be able to go through the Steiner points in a sequence of faces without "losing height", i.e., along a level path. We achieve this by slicing the terrain with a set of horizontal planes, and then putting Steiner points where the planes intersect the edges. The set of horizontal planes includes one plane through each vertex of the terrain, and other planes in

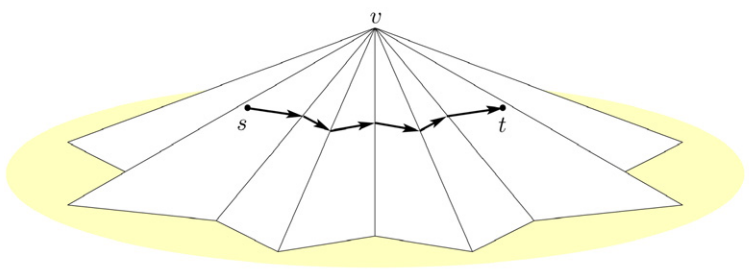

Fig. 3. An SDP that comes close to a vertex $O(n)$ number of times.

between them that are close enough to guarantee a good approximation ratio. Our two algorithms for SDPs differ from each other in the choice of the positions of the horizontal planes.

The second issue is that the previous Steiner point approaches rely on the property that shortest paths in the Weighted Region Problem or in the Shortest Anisotropic Path Problem cannot come very close to a particular vertex more than once. This property does not hold for shortest paths in the SDP problem. In fact, it is possible to construct a terrain where an SDP comes very close to a vertex v as many as $O(n)$ times, moving far away from v after every visit of the vicinity of v. Consider the terrain in Fig. 3 which consists of the triangular faces of a pyramid with a star-shaped base. The points s and t have the same height, so the SDP P from s to t must consist of level segments. Moreover, P consists of $O(n)$ segments in the figure. By moving the convex vertices at the base away from the "center" of the base while keeping them on the same plane, we can make the points of P that are far away from v move even further away from v. Clearly it is possible to make P enter and leave a region close to v as many as $O(n)$ number of times. Because of such a possibility with an SDP, the analysis of our Steiner point approach is completely different from previous approaches.

3. Algorithms for shortest descending paths

In this section we present two algorithms for approximating shortest descending paths—the first one places Steiner points evenly along terrain edges and the second one places them in a geometric progression. Our discussion relies on the following well known [5,25] properties of an SDP:

Lemma 3.1. *Any subpath of an SDP is an SDP.*

Lemma 3.2. *The intersection of an SDP with a face of the terrain is either empty or a line segment.*

3.1. Approximation using uniform Steiner points

In our first algorithm the Steiner points on each edge are evenly spaced. To determine their positions, we first take a set of horizontal planes such that any two consecutive planes are within distance δ of each other, where δ is a small constant that depends on the approximation factor and terrain parameters. We then put a Steiner point at the intersection point of each of these planes with each of the terrain edges. One important observation is that this scheme makes the distance between consecutive Steiner points on an edge dependent on the slope of that edge. For instance, the distance between consecutive Steiner points is more for an almost-level edge than for an almost vertical edge. Since θ is the largest acute angle between a non-level edge and a vertical line, it can be shown that the distance between consecutive Steiner points on a non-level edge is at most $\delta \sec\theta$ (Lemma 3.5). Because of the situation depicted in Fig. 2, we cannot place extra Steiner points *only* on the edges that are almost level. We guarantee a good approximation ratio by choosing δ appropriately. More precisely, we make sure that $\delta \sec\theta$ is small enough for the desired approximation ratio. Note that we can put Steiner points on a level edge without considering heights, since a level edge can never result in the situation depicted in Fig. 2 (because all the points on such an edge have the same height).

Our algorithm runs in two phases. In the preprocessing phase, we place the Steiner points, and then construct a shortest path tree in the corresponding graph. During the query phase, the shortest path tree gives an approximate SDP in a straightforward manner.

3.1.1. Preprocessing phase

Let $\delta = \frac{\epsilon h \cos\theta}{4n}$. We subdivide every non-level edge e of the terrain by placing Steiner points at the points where e intersects each of the following planes: $z = j\delta$ for all positive integers j, and $z = h(x)$ for all vertices x of the terrain. We subdivide every level edge e by putting enough Steiner points so that the length of each part of e is at most $\delta \sec\theta$. Let V be the set of all the vertices and Steiner points in the terrain. We then construct a weighted directed graph $G = (V, E)$ as follows, starting with $E = \emptyset$. For every pair (x, y) of points in V adjacent to a face f of the terrain, we add to E a directed link from x to y if and only if $h(x) \geqslant h(y)$ and xy is either an edge of the terrain or a segment through the interior of f.

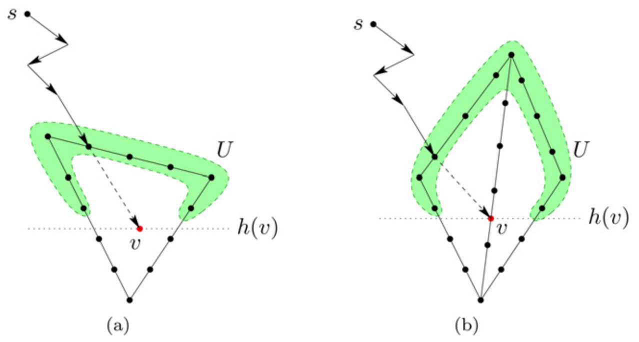

Fig. 4. Finding an SDP from s to an interior point v of (a) a face and (b) an edge.

Note that we do *not* add a link between two points on the same edge unless both of them are vertices. Each link in E is assigned a weight equal to the length of the corresponding line segment in the terrain. Finally we construct a shortest path tree T rooted at s in G using the Bushwhack algorithm.

Note that we are mentioning set E only to make the discussion easy. In practice, we do not construct E explicitly because the neighbors of a node $x \in V$ in the graph are determined *during* the execution of the Bushwhack algorithm.

3.1.2. Query phase

When the query point v is a node of G, we return the path from s to v in T as an approximate SDP. Otherwise, we find the node u among those in V lying in the face(s) containing v such that $h(u) \geqslant h(v)$, and the sum of the length of the path from s to u in T and the length of the segment uv is minimum. We return the corresponding path from s to v as an approximate SDP in this case. To elaborate more on the latter case, let U be the set consisting of the nodes $u \in V$ with the following properties:

(i) u and v lie in a common face, and
(ii) $h(u) \geqslant h(v)$.

It is easy to see that if v is an interior point of a face, then all the nodes in U lie on at most three edges of that face (Fig. 4(a)). Otherwise, v is an interior point of an edge, and there are at most four edges on which the nodes in U can lie (Fig. 4(b)). Since we already know the length of an SDP from s to any $u \in U$, we can find in $|U|$ iterations the node $u \in U$ that minimizes the length of the path constructed by concatenating the segment uv at the end of the path from s to u in T. The corresponding path is returned as an approximate SDP.

3.1.3. Correctness and analysis

For the proof of correctness, we show that an SDP P from s to any point v in the terrain is approximated by a path P' from s to v in the *augmented graph* $G_v(V_v, E_v)$ constructed as follows. We first add v to graph G, and then add to this graph a link uv with weight $|uv|$ for every $u \in V$ such that u and v lie in a common face, and $h(u) \geqslant h(v)$. We construct path P' from P as follows. Note that P' might not be the path returned by our algorithm, but it provides an upper bound on the length of the returned path.

Let $P = (s = p_0, p_1, p_2, \dots, p_k, v = p_{k+1})$ be an SDP from s to v such that p_i and p_{i+1} are two different boundary points of a common face for all $i \in [0, k-1]$, and p_k and p_{k+1} are two points of a common face. For ease of discussion, let e_i be an edge of the terrain through p_i for all $i \in [1, k]$ (e_i can be any edge through p_i if p_i is a vertex). Intuitively, we construct P' by moving each intermediate node of P upward to the nearest Steiner point. More precisely, we define a path $P' = (s = p'_0, p'_1, p'_2, \dots, p'_k, v = p'_{k+1})$ as follows. For each $i \in [1, k]$, let $p'_i = p_i$ if p_i is a vertex of the terrain. Otherwise, let p'_i be the nearest point from p_i in $V \cap e_i$ such that $h(p'_i) \geqslant h(p_i)$. Such a point always exists in V because p_i is an interior point of e_i in this case, and it has two neighbors x and y in $V \cap e_i$ such that $h(x) \geqslant h(p_i) \geqslant h(y)$. Note that each node of P' except possibly the last one is either a vertex or a Steiner point.

Lemma 3.3. *For all $i \in [0, k]$, $h(p'_i) \geqslant h(p'_{i+1})$.*

Proof. We first claim that $h(p'_i) \geqslant h(p_{i+1})$. This claim follows from the facts that $h(p'_i) \geqslant h(p_i)$ by the definition of p'_i, and $h(p_i) \geqslant h(p_{i+1})$ as P is descending. Now consider the following two cases:

Case 1. $p'_{i+1} = p_{i+1}$ or e_{i+1} is a level edge. In this case, $h(p'_{i+1}) = h(p_{i+1})$. It follows from the inequality $h(p'_i) \geqslant h(p_{i+1})$ that $h(p'_i) \geqslant h(p'_{i+1})$.

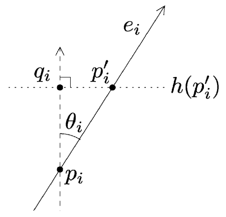

Fig. 5. Bounding $|p_i p'_i|$ when $p_i \neq p'_i$ and e_i is a non-level edge.

Case 2. $p'_{i+1} \neq p_{i+1}$ and e_{i+1} is a non-level edge. In this case, there is either one or no point in e_{i+1} at any particular height. Let p''_{i+1} be the point in e_{i+1} such that $h(p''_{i+1}) = h(p'_i)$, or if no such point exists, let p''_{i+1} be the upper vertex of e_{i+1}. In the latter case, we can infer from the inequality $h(p'_i) \geqslant h(p_{i+1})$ that $h(p'_i) > h(p''_{i+1})$. Therefore we have $h(p'_i) \geqslant h(p''_{i+1})$ in both cases. Since $p''_{i+1} \in V \cap e_{i+1}$, the definition of p'_{i+1} implies that $h(p''_{i+1}) \geqslant h(p'_{i+1})$. So, $h(p'_i) \geqslant h(p'_{i+1})$.

Therefore, $h(p'_i) \geqslant h(p'_{i+1})$ for all $i \in [0, k]$. □

Lemma 3.4. *Path P' exists in G_v.*

Proof. It is sufficient to prove that $p'_i p'_{i+1} \in E_v$ for all $i \in [0, k-1]$, because by definition both p'_i and p'_{i+1} are in V_v, and $(p'_k, v) \in E_v$.

For all $i \in [0, k-1]$, p'_i and p'_{i+1} are boundary points of a common face by definition, and $h(p'_i) \geqslant h(p'_{i+1})$ by Lemma 3.3. It then follows from the construction that $p'_i p'_{i+1} \notin E$ only in the case that both of p'_i and p'_{i+1} lie on a common edge, and at most one of them is a vertex. We show as follows that this is impossible. When both p_i and p_{i+1} are vertices of the terrain, both p'_i and p'_{i+1} are vertices. When at least one of p_i and p_{i+1} is an interior point of an edge, they cannot lie on a common edge (Lemma 3.2); therefore, both of p'_i and p'_{i+1} cannot lie on a common edge unless both of p'_i and p'_{i+1} are vertices. So, it is impossible that both p'_i and p'_{i+1} lie on a common edge, and at most one of them is a vertex. Therefore, $p'_i p'_{i+1} \in E \subset E_v$. □

Lemma 3.5. *For all $i \in [1, k]$, $|p_i p'_i| \leqslant \frac{\epsilon h}{4n}$.*

Proof. It is sufficient to show that $|p_i p'_i| \leqslant \delta \sec\theta$ for all i, because $\delta \sec\theta = \frac{\epsilon h}{4n}$.

When $p_i = p'_i$, $|p_i p'_i| = 0 < \delta \sec\theta$. When $p_i \neq p'_i$, and e_i is a level edge, $|p_i p'_i| \leqslant \delta \sec\theta$ by construction. We will now focus on the case $p_i \neq p'_i$ and e_i is a non-level edge.

Consider the vertical plane containing the edge e_i. Construct a line vertically upward from p_i to the point q_i where $h(q_i) = h(p'_i)$ (Fig. 5). Let θ_i be the angle $\angle q_i p_i p'_i$. Since $h(q_i) = h(p'_i) > h(p_i)$, θ_i is an acute angle, and hence $\theta \geqslant \theta_i$, which implies: $\cos\theta \leqslant \cos\theta_i = \frac{|q_i p_i|}{|p_i p'_i|}$, and therefore, $|p_i p'_i| \leqslant |q_i p_i| \sec\theta$. As $q_i p_i$ is a vertical line segment, $|q_i p_i| = h(q_i) - h(p_i) = h(p'_i) - h(p_i) \leqslant \delta$ by construction, and therefore, $|p_i p'_i| \leqslant |q_i p_i| \sec\theta \leqslant \delta \sec\theta$. □

Lemma 3.6. *The algorithm returns a $(1+\epsilon)$-SDP.*

Proof. Let P and P' be respectively an SDP and a path in G_v as described above. Our algorithm finds a shortest path P'' in G_v, which provides a descending path of the same length. Since P' is a path in G_v (Lemma 3.4), the length of P'' is at most the length of P', and therefore it is sufficient to prove that the length of P' is at most $(1+\epsilon)$ times the length of P.

When $k = 0$, implying that P does not cross an edge of the terrain, we have $P = (s, v) = P'$, which proves the lemma trivially. When $k > 0$, the length of P' is equal to:

$$\begin{aligned}
\sum_{i=0}^{k} |p'_i p'_{i+1}| &\leqslant \sum_{i=0}^{k} \left(|p'_i p_i| + |p_i p_{i+1}| + |p_{i+1} p'_{i+1}|\right) \quad \text{(from triangle inequality)} \\
&= \sum_{i=0}^{k} |p_i p_{i+1}| + 2\sum_{i=1}^{k} |p_i p'_i| \quad (\text{since } p_0 = p'_0 \text{ and } p_{k+1} = p'_{k+1}) \\
&\leqslant \sum_{i=0}^{k} |p_i p_{i+1}| + \frac{\epsilon h k}{2n} \quad \text{(Lemma 3.5)} \\
&< \sum_{i=0}^{k} |p_i p_{i+1}| + \epsilon h,
\end{aligned}$$

because it follows from Lemma 3.2 that k is less than $2n$, the number of faces in the terrain. Now, since $k > 0$, p_1 lies on the edge opposite to p_0 in the face containing both p_0 and p_1, and therefore, $h \leqslant |p_0 p_1| \leqslant \sum_{i=0}^{k} |p_i p_{i+1}|$. So, $\sum_{i=0}^{k} |p'_i p'_{i+1}| < (1+\epsilon)\sum_{i=0}^{k} |p_i p_{i+1}|$, and therefore, the length of P' is at most $(1+\epsilon)$ times the length of P. □

Lemma 3.7. *Let $X = \left(\frac{L}{h}\right)\sec\theta$. Graph G has less than $\frac{15n^2X}{\epsilon}$ nodes and $O\left(\frac{n^3X^2}{\epsilon^2}\right)$ links. Moreover, it has less than $\frac{5nX}{\epsilon}$ nodes along any edge of the terrain.*

Proof. We will first prove the last part of the lemma. For each edge e of the terrain, the number of Steiner points corresponding to the planes $z = j\delta$ is at most $\frac{L}{\delta} - 1$, and the number of Steiner points corresponding to the planes $z = h(x)$ is at most $n-2$. So,

$$|V \cap e| \leqslant \left(\frac{L}{\delta} - 1\right) + (n-2) + 2 < \frac{L}{\delta} + n \leqslant 5n\left(\frac{L}{h}\right)\left(\frac{1}{\epsilon}\right)\sec\theta = \frac{5nX}{\epsilon}$$

because $\delta = \frac{\epsilon h \cos\theta}{4n}$ and $\left(\frac{L}{h}\right)\left(\frac{1}{\epsilon}\right)\sec\theta \geqslant 1$.

We will now compute $|V|$ and $|E|$. Let $c = \frac{5nX}{\epsilon}$ for ease of discussion. As the number of edges is at most $3n$, we have: $|V| < 3nc = \frac{15n^2X}{\epsilon}$.

For each face f of the terrain, there are less than $3c$ points in $V \cap f$, and each such point has less than $2c$ neighbors in f (more precisely, in the induced subgraph $G[V \cap f]$). So, the number of directed links in E contributed by f is less than $6c^2$, and this bound is tight for a level face. Because there are at most $2n$ faces, $|E| < 12nc^2 = O\left(\frac{n^3X^2}{\epsilon^2}\right)$. □

Theorem 1. *Let $X = \left(\frac{L}{h}\right)\sec\theta$. Given a vertex s, and a constant $\epsilon \in (0,1]$, we can discretize the terrain with $\frac{15n^2X}{\epsilon}$ Steiner points so that after a preprocessing phase that takes $O\left(\frac{n^2X}{\epsilon}\log\left(\frac{nX}{\epsilon}\right)\right)$ time for a given vertex s, we can determine a $(1+\epsilon)$-approximate SDP from s to any point v in:*

(i) *$O(n)$ time if v is a vertex of the terrain or a Steiner point, and*
(ii) *$O\left(\frac{nX}{\epsilon}\right)$ time otherwise.*

Proof. The approximation factor follows from Lemma 3.6.

As we have mentioned before, we do not construct E explicitly because the neighbors of a node $x \in V$ in the graph are determined during the execution of the Bushwhack algorithm. As a result, the (implicit) construction of G takes $O(|V|)$ time. It follows from the running time of the Bushwhack algorithm that the preprocessing time of our algorithm is $O(|V|\log|V|) = O\left(\frac{n^2X}{\epsilon}\log\left(\frac{nX}{\epsilon}\right)\right)$ by Lemma 3.7.

During the query phase, if v is a vertex of the terrain or a Steiner point, the approximate path is in the tree T. Because the tree has height $O(n)$, it takes $O(n)$ time to trace the path. Otherwise, v is an interior point of a face or an edge of the terrain. The last intermediate node u on the path to v is a vertex or a Steiner point that lies on the boundary of a face containing v. If v is interior to a face [an edge], there are 3 [respectively 4] edges of the terrain on which u can lie. Thus there are $O\left(\frac{nX}{\epsilon}\right)$ choices for u by Lemma 3.7, and we try all of them to find the best approximate path, which takes $O\left(\frac{nX}{\epsilon}\right) + O(n) = O\left(\frac{nX}{\epsilon}\right)$ time. □

Corollary 1. *If the answer to a query is the length of an SDP (rather than the SDP itself), the query times for cases* (i) *and* (ii) *of Theorem 1 become $O(1)$ and $O\left(\frac{nX}{\epsilon}\right)$ respectively.*

Note that the space requirement of our algorithm is $O(|V|) = O\left(\frac{n^2X}{\epsilon}\right)$ since we are not storing E explicitly. Also note that using Dijkstra's algorithm with a Fibonacci heap [15] instead of the Bushwhack algorithm yields an even simpler algorithm with a preprocessing time of $O(|V|\log|V| + |E|) = O\left(n^3\left(\frac{X}{\epsilon}\right)^2\right)$.

3.2. Approximation using Steiner points in geometric progression

Unlike our first algorithm where the Steiner points on each edge are evenly spaced, our second algorithm places them non-uniformly along the edges. The Steiner points we use here are of two kinds. We first place Steiner points in "geometric progression" along the edges, as done by Aleksandrov et al. [7]. We call these points *primary Steiner points*. Then we place more Steiner points, called *isohypse Steiner points*, to guarantee that for every descending path in the terrain there exists a descending path through the Steiner points. Although the number of Steiner points used in this technique is more than in our first algorithm, the running time of the resulting algorithm no longer depends on the slope of the edges.

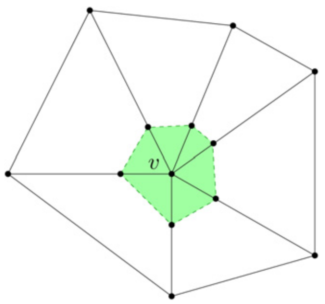

Fig. 6. Vicinity of a vertex.

3.2.1. *Preprocessing phase*

The primary Steiner points are placed in such a way that for each vertex v of an edge e, there is a set of primary Steiner points whose distances from v form a geometric progression. Although the distance between a pair of consecutive Steiner points on e increases as we move away from v, we can still guarantee a good approximation ratio. This is because intuitively the length of a segment connecting two edges adjacent to v increases as we move the segment away from v—see Lemma 3.10 for a more precise statement. One observation is that if we want to maintain the geometric progression of the distances for the Steiner points very close to v, we would need infinitely many Steiner points near v. To avoid this problem, we do not put any primary Steiner points in a small region near v.

Before going into further details, we will define a few constants for ease of discussion. Let $\delta_1 = \frac{\epsilon h}{6n}$, and $\delta_2 = \frac{\epsilon h}{6L}$. The constant δ_1 will define a region near v where we do not put any primary Steiner points, while δ_2 will determine the distances between consecutive primary Steiner points outside that region.

Definition 1 *(Vicinity of a vertex).* In a face f incident to a vertex v, let p_1 and p_2 be two points lying on two different edges of f at v such that $|vp_1| = |vp_2| = \delta_1$. Clearly, $\triangle vp_1p_2$ is an isosceles triangle. The *vicinity of* v is defined to be the union of all such isosceles triangles around v (Fig. 6).

Note that the vicinities of any two vertices v_1 and v_2 are mutually disjoint because $\delta_1 < \frac{h}{2} < \frac{|v_1v_2|}{2}$.

In the preprocessing phase, we determine the positions of the Steiner points as follows. First, on every edge $e = v_1v_2$ we place primary Steiner points at points $p \in e$ such that $|pq| = \delta_1(1+\delta_2)^i$ for $q \in \{v_1, v_2\}$ and $i \in \{0, 1, 2, \ldots\}$. Then we add up to $3n$ isohypse Steiner points for each primary Steiner point and for each vertex, as follows. For every non-level edge e, and every point p that is either a primary Steiner point or a vertex, we place an isohypse Steiner point at the point where e intersects the horizontal plane through p (i.e., the plane $z = h(p)$).

After placing the Steiner points, we construct a weighted directed graph $G = (V, E)$ and then construct a shortest path tree T rooted at s in G in the same way as in our first algorithm (Section 3.1.1).

3.2.2. *Query phase*

The queries are handled in exactly the same manner as in Section 3.1.2.

3.2.3. *Correctness and analysis*

For the proof of correctness, we follow the same approach used in Section 3.1.3: we show that an SDP P from s to any point v in the terrain is approximated by a path P' from s to v in the augmented graph $G_v(V_v, E_v)$. We construct path P' by moving each intermediate node of P upward to the nearest Steiner point. The correctness of our algorithm follows because the path returned by our algorithm is not longer than P'.

Let $P = (s = p_0, p_1, p_2, \ldots, p_k, v = p_{k+1})$ be an SDP from s to v such that p_i and p_{i+1} are two different boundary points of a common face for all $i \in [0, k-1]$, and p_k and p_{k+1} are two points of a common face. Let e_i be an edge of the terrain through p_i for all $i \in [1, k]$; e_i can be any edge through p_i if p_i is a vertex. Now define path $P' = (s = p'_0, p'_1, p'_2, \ldots, p'_k, v = p'_{k+1})$ as follows: for each $i \in [1, k]$, let $p'_i = p_i$ if p_i is a vertex of the terrain; otherwise, let p'_i be the nearest point from p_i in $V \cap e_i$ such that $h(p'_i) \geqslant h(p_i)$.

Lemma 3.8. *For all $i \in [0, k]$, $h(p'_i) \geqslant h(p'_{i+1})$.*

Proof. The proof is exactly the same as in Lemma 3.3. □

Lemma 3.9. *Path P' exists in G_v.*

Proof. The proof is exactly the same as in Lemma 3.4. □

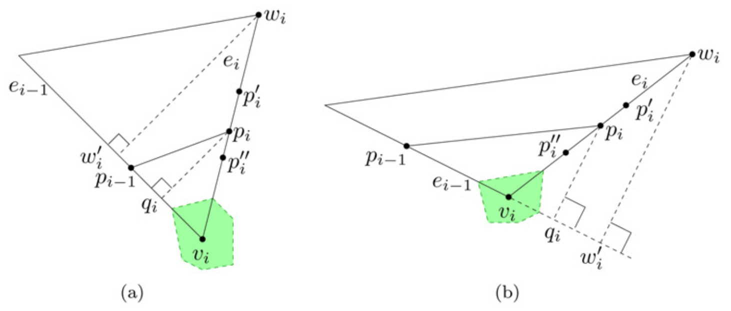

Fig. 7. Bounding $|p_i p'_i|$ when the face angle at v_i is (a) acute and (b) obtuse.

Lemma 3.10. *For all $i \in [1, k]$ such that p_i is not inside a vertex vicinity, $|p_i p'_i| < \frac{\epsilon}{6}|p_{i-1} p_i|$.*

Proof. If p_i coincides with p'_i, the lemma follows trivially as $|p_i p'_i| = 0$. We will now focus on the case when these two points do not coincide. Since p_i is not inside a vertex vicinity, there is another Steiner point p''_i in e_i such that p'_i and p''_i lie on the opposite sides of p_i. Let v_i be the common vertex of e_{i-1} and e_i, w_i be the other vertex of e_i, and q_i and w'_i be two points in e_{i-1} such that $p_i q_i \perp e_{i-1}$ and $w_i w'_i \perp e_{i-1}$. Fig. 7 depicts these vertices and points, for both the cases that the face angle at v_i is (a) acute and (b) obtuse.

We will first show that $|p'_i p''_i| \leqslant \delta_2 |v_i p_i|$. When $|v_i p''_i| < |v_i p'_i|$, we have $|v_i p'_i| \leqslant (1 + \delta_2)|v_i p''_i|$ by construction, and therefore, $|p'_i p''_i| = |v_i p'_i| - |v_i p''_i| \leqslant \delta_2 |v_i p''_i| \leqslant \delta_2 |v_i p_i|$. On the other hand, when $|v_i p''_i| > |v_i p'_i|$, then we have $|v_i p''_i| \leqslant (1 + \delta_2)|v_i p'_i|$, which similarly implies that $|p'_i p''_i| \leqslant \delta_2 |v_i p_i|$.

Therefore, $|p_i p'_i| < |p'_i p''_i| \leqslant \delta_2 |v_i p_i| = \delta_2 |q_i p_i| \cdot \frac{|v_i p_i|}{|q_i p_i|} = \delta_2 |q_i p_i| \cdot \frac{|v_i w_i|}{|w'_i w_i|}$. The lemma then follows from the inequalities $|q_i p_i| \leqslant |p_{i-1} p_i|$, $|v_i w_i| \leqslant L$, and $|w'_i w_i| \geqslant h$, as well as the definition of δ_2. $\square$

Lemma 3.11. *For all $i \in [1, k]$ such that p_i is on or inside a vertex vicinity, $|p_i p'_i| \leqslant \frac{\epsilon h}{6n}$.*

Proof. If p_i is a vertex, the lemma follows trivially since $p'_i = p_i$ in this case. If p_i is not a vertex, let e_i be the edge containing p_i, and v_i be the vertex whose vicinity contains p_i. It is not hard to see that v_i is a vertex of e_i because δ_1 is strictly less than h. Let q_i be the primary Steiner point on e_i which lies at distance δ_1 from v_i. Clearly p_i lies in line segment $v_i q_i$. Now p'_i cannot be outside line segment $v_i q_i$ because otherwise we would have chosen either v_i or q_i as p'_i. As a result, p'_i also lies in line segment $v_i q_i$. Therefore, $|p_i p'_i| \leqslant |v_i q_i| = \delta_1 = \frac{\epsilon h}{6n}$. $\square$

Lemma 3.12. *The algorithm returns a $(1 + \epsilon)$-SDP.*

Proof. As in the proof of Lemma 3.6, it is sufficient to prove that the length of P' is at most $(1 + \epsilon)$ times the length of P, where P and P' be respectively an SDP and a path in G_v as described above.

When $k = 0$, we have $P = (s, v) = P'$, which proves the lemma trivially. Otherwise, the length of P' is equal to:

$$\begin{aligned}
\sum_{i=0}^{k} |p'_i p'_{i+1}| &\leqslant \sum_{i=0}^{k} |p_i p_{i+1}| + 2\sum_{i=1}^{k} |p_i p'_i| \quad \text{(as in the proof of Lemma 3.6)} \\
&< \sum_{i=0}^{k} |p_i p_{i+1}| + 2\sum_{i=1}^{k} \left(\frac{\epsilon}{6}|p_{i-1} p_i| + \frac{\epsilon h}{6n} \right) \quad \text{(by Lemmas 3.10 and 3.11)} \\
&\leqslant \left(1 + \frac{\epsilon}{3}\right) \sum_{i=0}^{k} |p_i p_{i+1}| + \frac{\epsilon h k}{3n} \\
&< \left(1 + \frac{\epsilon}{3}\right) \sum_{i=0}^{k} |p_i p_{i+1}| + \frac{2\epsilon h}{3},
\end{aligned}$$

since $k < 2n$. Now, because $k > 0$, we have $h \leqslant |p_0 p_1| \leqslant \sum_{i=0}^{k} |p_i p_{i+1}|$. So, $\sum_{i=0}^{k} |p'_i p'_{i+1}| < (1 + \epsilon) \sum_{i=0}^{k} |p_i p_{i+1}|$, and therefore, the length of P' is at most $(1 + \epsilon)$ times the length of P. $\square$

Observation 3.1. *For any real number $x \in (0, 1]$, $\log(1 + x) > \frac{x \log e}{2}$.*

Lemma 3.13. *Graph G has less than $\frac{153n^2L}{\epsilon h}\log(\frac{6nL}{\epsilon h})$ nodes and $O(\frac{n^3L^2}{\epsilon^2h^2}\log^2(\frac{nL}{\epsilon h}))$ links. Moreover, it has less than $\frac{51nL}{\epsilon h}\log(\frac{6nL}{\epsilon h})$ nodes along any edge of the terrain.*

Proof. We will first compute an upper bound on the number of primary Steiner points, which will then be used to prove the lemma.

Let n_e be the number of primary Steiner points on edge e. It is straightforward to see that n_e is at most $2j$, where j is the largest integer satisfying the inequality $\delta_1(1+\delta_2)^j < L$. Therefore,

$$n_e \leqslant 2j < \frac{2\log(\frac{L}{\delta_1})}{\log(1+\delta_2)} < \frac{2}{\frac{\delta_2\log e}{2}}\log\left(\frac{L}{\delta_1}\right) < \frac{16.64L}{\epsilon h}\log\left(\frac{6nL}{\epsilon h}\right).$$

Since there are at most $3n$ edges in the terrain, the total number of primary Steiner points is at most $3nn_e$, which is less than $\frac{50nL}{\epsilon h}\log(\frac{6nL}{\epsilon h})$.

We will now prove the last part of the lemma. For each point p that is either a primary Steiner points and a vertex, there is at most one node in $V\cap e$ for any edge e. This is obvious when p lies on e. On the other hand, if p does not lie on e, there is at most one isohypse Steiner point on e that corresponds to p. Using the above bound on the number of primary Steiner points, we have:

$$|V\cap e| < \frac{50nL}{\epsilon h}\log\left(\frac{6nL}{\epsilon h}\right)+n < \frac{51nL}{\epsilon h}\log\left(\frac{6nL}{\epsilon h}\right).$$

We will now compute $|V|$ and $|E|$. Let $c=\frac{51nL}{\epsilon h}\log(\frac{6nL}{\epsilon h})$ for ease of discussion. Because the number of edges is at most $3n$, we have: $|V| < 3nc = \frac{153n^2L}{\epsilon h}\log(\frac{6nL}{\epsilon h})$. Using the same argument we used in the proof of Lemma 3.7, we can say that the number of directed links in E contributed by each face of the terrain is less than $6c^2$. Because there are at most $2n$ faces, $|E| < 12nc^2 = O(\frac{n^3L^2}{\epsilon^2h^2}\log^2(\frac{nL}{\epsilon h}))$. □

Theorem 2. *Given a vertex s, and a constant $\epsilon\in(0,1]$, we can discretize the terrain with at most $\frac{150n^2L}{\epsilon h}\log(\frac{6nL}{\epsilon h})$ Steiner points so that after a preprocessing phase that takes $O(\frac{n^2L}{\epsilon h}\log^2(\frac{nL}{\epsilon h}))$ time for a given vertex s, we can determine a $(1+\epsilon)$-approximate SDP from s to any point v in:*

(i) *$O(n)$ time if v is a vertex of the terrain or a Steiner point, and*
(ii) *$O(\frac{nL}{\epsilon h}\log(\frac{nL}{\epsilon h}))$ time otherwise.*

Proof. The proof is the same as in Theorem 1 except that we use Lemmas 3.12 and 3.13 instead of Lemmas 3.6 and 3.7, respectively. □

Corollary 2. *If the answer to a query is the length of an SDP, the query times for cases* (i) *and* (ii) *of Theorem 2 become $O(1)$ and $O(\frac{nL}{\epsilon h}\log(\frac{nL}{\epsilon h}))$ respectively.*

As in the case of our first algorithm, we can use Dijkstra's algorithm with a Fibonacci heap [15] instead of the Bushwhack algorithm to have an even simpler algorithm with a preprocessing time of $O(\frac{n^3L^2}{\epsilon^2h^2}\log^2(\frac{nL}{\epsilon h}))$.

4. Algorithms for shortest gently descending paths

In this section we present two algorithms for approximating shortest gently descending paths. First we start with some properties of these paths.

4.1. Properties of SGDPs

Observe that as in the case of shortest descending paths (Lemma 3.1), the following holds for shortest gently descending paths.

Lemma 4.1. *Any subpath of an SGDP is an SGDP.*

Recall the definition of critical paths in Section 2.1. Next we show that any critical path in the terrain is an SGDP.

Lemma 4.2. *Any critical path from a point a to a point b in the terrain is an SGDP of length $(h(a)-h(b))\sec\psi$.*

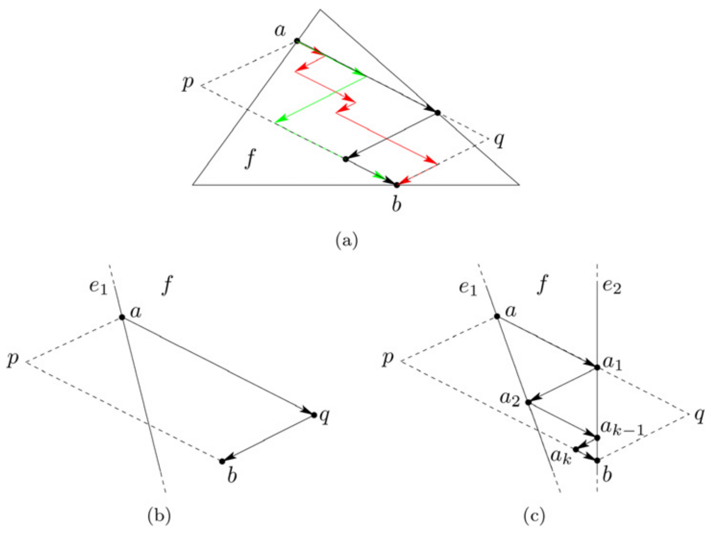

Fig. 8. (a) Three of the infinitely many critical paths from a to b when ab is a steep segment and neither a nor b is a locally sharp vertex in f. A critical path when parallelogram $apbq$ contains (b) one and (c) two edges of f.

Proof. Any critical path P is a gently descending path. Since each segment of P makes an angle ψ with a vertical line, the length of P is $(h(a) - h(b))\sec\psi$. Ignoring the terrain, *any* gently descending path from a to any point at height $h(b)$ has length at least $(h(a) - h(b))\sec\psi$. So, P is an SGDP. □

We will introduce a notation to simplify our expressions involving the length of an SGDP. For any two points p and q in a common face f, let $\|pq\| = |(h(p) - h(q))|\sec\psi$ when line pq is steep, and $\|pq\| = |pq|$, otherwise. Thus $\|pq\|$ is the length of an SGDP from p to q if one exists.

Observation 4.1. *For any three points p, q and r in a face f:* (i) $\|pq\| = \|qp\|$, (ii) $\|pr\| \leqslant \|pq\| + \|qr\|$, *and* (iii) $|pq| \leqslant \|pq\| \leqslant |pq|\sec\psi$.

Proof. The proof is obvious except for the second inequality of case (iii), which follows from the inequalities $\sec\psi \geqslant 1$ and $|pq| \geqslant |h(p) - h(q)|$. □

Like other Steiner point approaches, our graph of Steiner points has a directed link (of appropriate cost) between two Steiner points a and b in a common face f such that the link represents an SGDP from a to b. However, unlike other well-studied shortest paths in terrains where the shortest path represented by the link lies completely in f, the SGDP in our case may go through many other faces. For example, the critical path in Fig. 1(a) is an SGDP by Lemma 4.2, and it goes through four faces even though both s and t lie on a common face. It is not straightforward to determine if such an SGDP exists at all—we need this information during the construction of the graph. Moreover, in case an SGDP exists, we want to know the number of faces used by the path because our algorithm has to return the path in the terrain that corresponds to the shortest path in the graph. Lemma 4.3 below handles these issues:

Lemma 4.3. *Let a and b be two points in a face f with $h(a) \geqslant h(b)$.*

(i) *If at least one of a and b is a sharp vertex, no gently descending path exists from a to b.*
(ii) *If neither a nor b is a locally sharp vertex in f, there exists an SGDP from a to b lying completely in f. Moreover, the SGDP is a critical path if ab is steep.*
(iii) *Otherwise, there exists an SGDP from a to b that uses at most $d + 1$ faces, and is a critical path.*

Proof.

(i) If a [or b] is a sharp vertex, the segment ap [respectively pb] is steep for any point p in any face incident to a [respectively b]. So, no gently descending path exists from a to b.
(ii) If ab is not a steep segment, this is the SGDP. Otherwise, there are many critical paths in f from a to b, as shown Fig. 8(a). To be precise, we trace one such path as follows. Let $apbq$ be the parallelogram defined by the critical directions at a and b in the plane containing f (Fig. 8). We follow a critical direction in f from a until we intersect either pb or qb or an edge of f. Let a_1 be the intersection point. Clearly, a_1 is not a locally sharp vertex in f, and therefore,

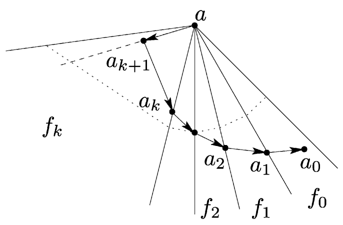

Fig. 9. A critical path from a vertex a that is locally sharp in $f = f_0$, but not in f_k.

if a_1 is on neither pb nor qb, we can again follow a critical direction from a_1 until we intersect either pb or qb or an edge of f. Let a_2 be this intersection point. We repeat this step until we reach a point a_k on either pb or qb. Thus we get the critical path $(a, a_1, a_2, \dots, a_k, b)$, which is an SGDP by Lemma 4.2. This completes the proof provided that this critical path always reaches either pb or qb.

When at most one edge of f intersects the interior of the parallelogram, our critical path reaches either p or q by following an edge of the parallelogram at a (Fig. 8(b)). When two edges e_1 and e_2 of f intersect the interior of the parallelogram (Fig. 8(c)), the point $e_1 \cap e_2$ is neither a nor b nor an interior point of f. This is because f is a triangle, and at most one edge of f at a [and b] intersects the interior of the parallelogram (since neither a nor b is locally sharp in f). So, there is a "gap" between e_1 and e_2 in the parallelogram, which gives us a positive constant c such that $|a_i a_{i+1}| > c$ for all $i \in [1, k-2]$. We therefore reach a_k in a finite number of steps (in at most $\frac{\|ab\|}{c} + 2$ steps to be precise).

(iii) In this case, at least one of a and b is a locally sharp vertex in f, and therefore, ab is a steep line segment. If a is not a locally sharp vertex in f, let $a' = a$. Otherwise, let a' be an interior point of line segment ab such that no vertex of the terrain lies strictly in between the planes $z = h(a)$ and $z = h(a')$. We similarly define a point b', and make sure that $h(a') > h(b')$. Clearly $h(a) > h(a') > h(b') > h(b)$, and $a'b'$ is a steep line segment. By case (ii) of the lemma there exists a critical path from a' to b' in f. We claim that there exists a critical path from a to a' through at most $\lfloor \frac{d}{2} \rfloor + 1$ faces when $a \neq a'$, and that there exists a critical path from b' to b through at most $\lfloor \frac{d}{2} \rfloor + 1$ faces when $b \neq b'$. Because face f is used by all these three subpaths, the whole path uses at most $d+1$ faces. The proof then follows from Lemma 4.2. We will now prove the first claim. The proof for the second claim is similar, and hence omitted.

Since a is not a sharp vertex, it must have some incident face in which it is not locally sharp. Let $(f = f_0, f_1, f_2, \dots, f_k)$ be (one of) the shortest sequence of faces around vertex a such that a is not a locally sharp vertex in f_k (Fig. 9). Clearly, $k \leqslant \lfloor \frac{d}{2} \rfloor + 1$. We will build as follows a critical path backward from a'. Let $a_0 = a'$. For each $i \in [0, k-1]$ in this order, since a is a locally sharp vertex in f_i, there is a point $a_{i+1} \in f_i \cap f_{i+1}$ such that $a_{i+1}a_i$ is a critical direction in f_i. The final point a_k lies on the edge between f_k and f_{k-1} which is a steep edge. Because a is not a locally sharp vertex in f_k, and no other vertex of f_k lies above the plane $z = h(a_k)$, there exists an interior point $a_{k+1} \in f_k$ such that both aa_{k+1} and $a_{k+1}a_k$ are critical directions. Clearly the path $(a, a_{k+1}, a_k, \dots, a_0 = a')$ is a critical path through at most $\lfloor \frac{d}{2} \rfloor + 1$ faces. $\square$

We would like the property of an SGDP that the path would visit a face at most once, because our method of approximating a path introduces some error each time the path crosses an edge. But unlike SDPs, this property does *not* hold for an SGDP. For example, the two SGDPs in Fig. 1 visit a face twice. In fact, there is no bound on the number of times a face may be visited—the SGDP in Fig. 1(a) can be made to spiral around the pyramid more and more times by making angle ψ closer and closer to $\frac{\pi}{2}$. The good news is that we can "convert" any non-ideal SGDP into an ideal one—we will prove this claim in the following lemma:

Lemma 4.4. *If there is a gently descending path from s to t, there exists an ideal SGDP from s to t.*

Proof. Let P_0 be a gently descending path from s to t. We first show that P_0 can be converted to an ideal path without increasing its length. Then we prove that the set of all ideal s–t paths is closed, which implies that the minimum length can be attained (not just approached). The lemma then follows immediately.

For the first claim, let f be a face whose interior intersects P_0, and let a_f [and b_f] be the infimum [respectively supremum] of the points in P_0 that lie in the *interior* of f (considering the natural ordering of the points in P_0, i.e., from s to t). Because both P_0 and f are closed sets, both a_f and b_f are points in P_0 on the boundary of f. It follows that neither a_f nor b_f is a locally sharp vertex in f by definition. So, Lemma 4.3(ii) implies that there is an SGDP from a_f to b_f that lies completely in f. Replacing the part of P_0 from a_f to b_f with an SGDP in f for all f yields an ideal s–t path P_1 which is not longer than P_0.

For the second claim we need to show that the infimum of the set of lengths of ideal gently descending paths from s to t is attained by an ideal path. Each ideal path corresponds to a sequence of faces with no repetition of faces. There are finitely many such face sequences. Thus it suffices to prove the claim for a single face sequence. This follows by induction on the length of the face sequence, based on the fact that from one face to the next face the path crosses an edge, which is a closed set. □

4.2. Approximation using uniform Steiner points

4.2.1. Algorithm

In our first approximation algorithm, we preprocess the terrain by adding uniformly spaced Steiner points as follows. We take a set of level planes such that the distance between any two consecutive planes is at most $\delta = \frac{\epsilon h}{4n} \cos\theta \cos\psi$. We make sure that there is a level plane through every vertex. We then put Steiner points at all the intersection points of these planes with the non-level edges of the terrain. On the level edges, we add enough Steiner points so that consecutive points are at most $\delta \sec\theta$ units apart. We then construct a weighted directed graph $G(V, E)$ in which each node in V represents either a Steiner point or a vertex, and there is a directed link $pq \in E$ if and only if all of the following are true:

(i) p and q lie in a common face,
(ii) $h(p) \geqslant h(q)$, and
(iii) neither p nor q is a sharp vertex.

By Lemma 4.3, there is an SGDP from p to q. The weight of link pq is the length of such an SGDP, i.e., $\|pq\|$. In the final step of our preprocessing we compute a shortest path tree T rooted at s using the Bushwhack algorithm. The Bushwhack algorithm works for any distance metric that satisfies the following property: if e and e' are two edges from the same face, and a and b are two points on edge e, then edge e' can be divided into two sub-intervals A and B, where points in A have a shorter path from a than from b and points in B have a shorter path from b than from a. This property holds for our distance metric (even though an SGDP connecting two points in face f may leave f). The proof follows from Sun and Reif [31, Lemmas 3 and 4], where they showed that this property holds for anisotropic paths.

To answer a query, we simply return the path from s to query point v in T if $v \in V$ and v is not a sharp vertex. If v is a sharp vertex, we return nothing since there is no SGDP from s to v. Otherwise, $v \notin V$. In this case, we find the node u among those in V lying in the face(s) containing v such that $h(u) \geqslant h(v)$, and the sum of $\|uv\|$ and the length of the path from s to u in T is minimum. Finally we return the corresponding path from s to v as an approximate SGDP.

There is an important issue regarding the path returned by our algorithm. It is a path in the graph augmented by vertex v. To obtain an actual path on the terrain we must replace each link ab in the path by an SGDP of the same length, which is possible by the definition of the links of the graph. Such an SGDP is not unique if ab is steep (Fig. 8(a)), but it is easy to compute one such path. In the case where neither a nor b is locally sharp in their common face f, we can even compute an SGDP from a to b in f with the minimum possible number of bends (i.e. with no more bends than any other SGDP from a to b in f). Note, however, that the problem of minimizing the total number of bends in an SGDP is NP-hard because the hardness proof in Ahmed and Lubiw [4] applies directly to SGDPs.

4.2.2. Correctness and analysis

The proof of correctness follows an approach similar to the one in Section 3.1.3: we show that an ideal SGDP P from s to any point v in the terrain is approximated by a path P' from s to v in the augmented graph $G_v(V_v, E_v)$ constructed as follows. We first add v to graph G, and then add to this graph a link uv with weight $\|uv\|$ for every $u \in V$ such that u and v lie in a common face, and $h(u) \geqslant h(v)$.

Let $\sigma_P = (s = p_0, p_1, p_2, \ldots, p_k, v = p_{k+1})$ be an ordered subsequence of the nodes in P such that (a) for each $i \in [0, k]$, the segments in-between p_i and p_{i+1} lie in a common face f_i, and (b) for each $i \in [0, k-1]$, the segment exiting from p_{i+1} does *not* lie in f_i. Note that p_i and p_{i+1} are two different boundary points of face f_i for all $i \in [0, k-1]$, and p_k and p_{k+1} are two different points of face f_k (p_{k+1} can be an interior point of f_k). For all $i \in [0, k]$, the part of P between p_i and p_{i+1} remains in f_i. Let e_i be an edge of the terrain through p_i for all $i \in [1, k]$ (e_i can be any edge through p_i if p_i is a vertex).

We construct a node sequence $P' = (s = p'_0, p'_1, p'_2, \ldots, p'_k, v = p'_{k+1})$ as follows: for each $i \in [1, k]$, let $p'_i = p_i$ if p_i is a vertex of the terrain; otherwise, let p'_i be the nearest point from p_i in $V \cap e_i$ such that $h(p'_i) \geqslant h(p_i)$. We will first prove in Lemma 4.7 that this node sequence defines a path in G_v.

Lemma 4.5. *For all $i \in [0, k]$, $h(p'_i) \geqslant h(p'_{i+1})$.*

Proof. We have $h(p'_i) \geqslant h(p_i)$ and $h(p_i) \geqslant h(p_{i+1})$ as in the proof of Lemma 3.3, even though both p_i and p'_i are defined differently here. This lemma then follows from the rest of the proof of Lemma 3.3. □

Lemma 4.6. *For all $i \in [0, k+1]$, p'_i is not a sharp vertex.*

Proof. None of $p'_0 = s$ and $p'_{k+1} = v$ are sharp vertices because both the segments sp_1 and p_kv are gently descending. For each $i \in [1, k]$, if p'_i is a sharp vertex, then p'_i is either the unique topmost vertex or the unique bottommost vertex in all incident faces. Therefore, either $h(p'_{i-1}) < h(p'_i) > h(p'_{i+1})$, or $h(p'_{i-1}) > h(p'_i) < h(p'_{i+1})$. Both of these are impossible by Lemma 4.5. So p'_i is a not sharp vertex. $\square$

Lemma 4.7. *Node sequence P' defines a path in G_v.*

Proof. It is sufficient to show that $p'_i p'_{i+1} \in E_v$ for all $i \in [0, k]$. For $i \in [0, k-1]$, since p'_i and p'_{i+1} are boundary points of face f_i by definition, $h(p'_i) \geqslant h(p'_{i+1})$ by Lemma 4.5, and neither p'_i nor p'_{i+1} is a sharp vertex by Lemma 4.6, it follows from the construction that $p'_i p'_{i+1} \in E \subseteq E_v$. The case $i = k$ is similar except that p'_k and $v = p'_{k+1}$ are points (i.e., not boundary points) in face f_k, and that $p'_k v \in E_v$. $\square$

Lemma 4.8. *For all $i \in [1, k]$, $\|p_i p'_i\| \leqslant \frac{\epsilon h}{4n}$.*

Proof. Using the current definition of δ in the second part of the proof of Lemma 3.5, it follows in a straightforward manner that $|p_i p'_i| \leqslant \delta \sec\theta$ for all i. Observation 4.1(iii) then implies:

$$\|p_i p'_i\| \leqslant |p_i p'_i| \sec\psi \leqslant \delta \sec\theta \sec\psi = \frac{\epsilon h}{4n},$$

which completes the proof. $\square$

Lemma 4.9. *The algorithm returns a $(1+\epsilon)$-SGDP.*

Proof. Let P and P' be respectively an ideal SGDP and a node sequence in G_v as described above. Our algorithm finds a shortest path P'' in G_v, which provides a gently descending path of the same length. Since P' is a path in G_v (Lemma 4.7), the length of P'' is at most the length of P', and therefore it is sufficient to prove that the length of P' is at most $(1+\epsilon)$ times the length of P.

When $k = 0$, both P and P' have length $\|sv\|$, which proves the lemma trivially. When $k > 0$, the length of P' is equal to:

$$\begin{aligned}
\sum_{i=0}^{k} \|p'_i p'_{i+1}\| &\leqslant \sum_{i=0}^{k} \bigl(\|p'_i p_i\| + \|p_i p_{i+1}\| + \|p_{i+1} p'_{i+1}\|\bigr) \quad \text{(Observation 4.1(ii))} \\
&= \sum_{i=0}^{k} \|p_i p_{i+1}\| + 2\sum_{i=1}^{k} \|p_i p'_i\| \quad \text{(Observation 4.1(i))} \\
&\leqslant \sum_{i=0}^{k} \|p_i p_{i+1}\| + \frac{\epsilon h k}{2n} \quad \text{(Lemma 4.8)} \\
&< \sum_{i=0}^{k} \|p_i p_{i+1}\| + \epsilon h,
\end{aligned}$$

since $k < 2n$. Now, because $k > 0$, we have:

$$h \leqslant |p_0 p_1| \leqslant \sum_{i=0}^{k} |p_i p_{i+1}| \leqslant \sum_{i=0}^{k} \|p_i p_{i+1}\|$$

by Observation 4.1(iii). Therefore, $\sum_{i=0}^{k} \|p'_i p'_{i+1}\| < (1+\epsilon) \sum_{i=0}^{k} \|p_i p_{i+1}\|$. So, the length of P' is at most $(1+\epsilon)$ times the length of P. $\square$

Lemma 4.10. *Let $X = \frac{L}{h} \sec\theta \sec\psi$. Graph G has $O\bigl(\frac{n^2 X}{\epsilon}\bigr)$ nodes in total, and $O\bigl(\frac{nX}{\epsilon}\bigr)$ nodes along any edge of the terrain.*

Proof. The proof is the same as that of Lemma 3.7, except that we use δ and X defined here. $\square$

Theorem 3. *Let $X = \frac{L}{h} \sec\theta \sec\psi$. Given a vertex s, we can preprocess the terrain in $O\bigl(\frac{n^2 X}{\epsilon} \log\bigl(\frac{nX}{\epsilon}\bigr)\bigr)$ time after which we can determine a $(1+\epsilon)$-approximate SGDP from s to any query point v in: (i) $O(nd)$ time if v is a vertex or a Steiner point, and (ii) $O\bigl(n\bigl(d + \frac{X}{\epsilon}\bigr)\bigr)$ time otherwise.*

Proof. The approximation factor follows from Lemma 4.9.

The preprocessing time of our algorithm is the same as the running time of the Bushwhack algorithm, which is $O(|V| \log |V|) = O\bigl(\frac{n^2 X}{\epsilon} \log\bigl(\frac{nX}{\epsilon}\bigr)\bigr)$ by Lemma 4.10.

During the query phase, if v is a vertex or a Steiner point, the approximate path is in the tree T. Because the tree has height $O(n)$, it takes $O(n)$ time to trace the path in the tree. Tracing the corresponding path in the terrain takes $O(nd)$ time by Lemma 4.3. The total query time is thus $O(nd)$ in this case. If v is neither a vertex nor a Steiner point, v is an interior point of a face or an edge of the terrain. The last intermediate node u on the path to v is a vertex or a Steiner point that lies on the boundary of a face containing v. If v is interior to a face [an edge], there are 3 [respectively 4] edges of the terrain on which u can lie. Thus there are $O(\frac{nX}{\epsilon})$ choices for u by Lemma 4.10, and we try all of them to find the shortest approximate distance from s to v. Finally tracing the corresponding path in the terrain takes $O(nd)$ time by Lemma 4.3. The total query time in this case is $O(\frac{nX}{\epsilon}) + O(nd) = O(n(d + \frac{X}{\epsilon}))$. □

Corollary 3. *If the answer to a query is the length of an SGDP (rather than the SGDP itself), the query times for cases* (i) *and* (ii) *of Theorem* 3 *become* $O(1)$ *and* $O(\frac{nX}{\epsilon})$, *respectively.*

4.3. Approximation using non-uniform Steiner points

Our second approximation algorithm differs from the first one only in the way Steiner points are placed. We follow the same scheme as in Section 3.2, except that the parameters δ_1 and δ_2 take into account the steepness. We now place Steiner points in two phases. First, on every edge $e = v_1v_2$ we place Steiner points at points $p \in e$ such that $|pq| = \delta_1(1+\delta_2)^i$ for $q \in \{v_1, v_2\}$ and $i \in \{0, 1, 2, \ldots\}$, where $\delta_1 = \frac{\epsilon h}{6n}\cos\psi$ and $\delta_2 = \frac{\epsilon h}{6L}\cos\psi$. In the second phase we slice the terrain with a level plane through every Phase 1 Steiner point and every vertex, and add Steiner points at the points where these planes intersect the terrain. All the lemmas and their proofs in Section 3.2 can be adapted to this scenario by considering the characteristics of an SGDP. We summarize the result in the following theorem.

Theorem 4. *Let* $X' = \frac{L}{h}\sec\psi$. *Given a vertex s, we can preprocess the terrain in* $O(\frac{n^2X'}{\epsilon}\log^2(\frac{nX'}{\epsilon}))$ *time after which we can determine a* $(1+\epsilon)$*-approximate SGDP from s to any point v in*: (i) $O(nd)$ *time if v is a vertex or a Steiner point, and* (ii) $O(nd + \frac{nX'}{\epsilon}\log(\frac{nX'}{\epsilon}))$ *time, otherwise.*

Proof. We can prove this theorem in the same way as we proved Theorem 2, but utilizing the properties of SGDPs as we did in Theorem 3. The proof is a mere exercise and as such does not lead to any new insights, and thus omitted. □

Corollary 4. *If the answer to a query is the length of an SGDP, the query times for cases* (i) *and* (ii) *of Theorem* 4 *become* $O(1)$ *and* $O(\frac{nX'}{\epsilon}\log(\frac{nX'}{\epsilon}))$, *respectively.*

5. Conclusion

It may appear that the running time can be improved by using the technique by Aleksandrov et al. [9] who place Steiner points along the bisectors of the face angles. Although the technique improves all previous results on the Weighted Region Problem, it cannot be used for the SDP problem very easily. The main problem is that it is not clear how to prove the existence of a *feasible* path that approximates an SDP.

It will be interesting to see whether the technique used in Cheng et al. [12] can be applied to absorb the geometric parameters in the complexities of our algorithms.

When query point v is neither a vertex of the terrain nor a Steiner point, the query phase can be made faster by using a point location data structure on the underlying weighted Voronoi diagram defined by the Steiner points on the boundary of each face.

Moet et al. [21] showed that the visibility map, shortest path map and Voronoi diagram can be computed more efficiently in a terrain in which certain geometric parameters have bounded values. They call such terrains *realistic terrains*, although they mention that their assumptions on the bounds on those parameters are *not* based on the terrains encountered in practice. It is interesting to know if practical terrains follow their assumptions. If this is the case, then two of our algorithms, more precisely those using non-uniform Steiner points, become independent of geometry in such terrains because realistic terrains satisfy $\frac{L}{h} = O(1)$.

References

[1] Pankaj K. Agarwal, Sariel Har-Peled, Meetesh Karia, Computing approximate shortest paths on convex polytopes, in: Proceedings of the 16th Annual Symposium on Computational Geometry, ACM, New York, NY, USA, 2000, pp. 270–279.

[2] Mustaq Ahmed, Anna Lubiw, An approximation algorithm for shortest descending paths, CoRR 0705.1364v1 [cs.CG], May 2007 (Tech. Report CS-2007-14, University of Waterloo).

[3] Mustaq Ahmed, Anna Lubiw, Properties of shortest descending paths, in: The 17th Fall Workshop on Computational and Combinatorial Geometry (FWCG), Hawthorne, New York, November 2007 (Extended abstract).

[4] Mustaq Ahmed, Anna Lubiw, Shortest anisotropic paths with few bends is NP-complete, in: The 18th Fall Workshop on Computational Geometry (FWCG): Abstracts, October 2008, pp. 28–29.

[5] Mustaq Ahmed, Anna Lubiw, Shortest descending paths through given faces, Comput. Geom. Theory Appl. 42 (5) (July 2009) 464–470, Special Issue on the Canadian Conference on Computational Geometry (CCCG 2005 and CCCG 2006).

[6] Mustaq Ahmed, Anna Lubiw, Anil Maheshwari, Shortest gently descending paths, in: Proceedings of the Third Annual Workshop on Algorithms and Computation (WALCOM), in: Lecture Notes in Computer Science, vol. 5431, Springer-Verlag, February 2009, pp. 59–70.

[7] Lyudmil Aleksandrov, Mark Lanthier, Anil Maheshwari, Jörg-Rüdiger Sack, An ϵ-approximation algorithm for weighted shortest paths on polyhedral surfaces, in: Proceedings of the Sixth Scandinavian Workshop on Algorithm Theory, in: Lecture Notes in Computer Science, vol. 1432, Springer-Verlag, Berlin, Germany, 1998, pp. 11–22.

[8] Lyudmil Aleksandrov, Anil Maheshwari, Jörg-Rüdiger Sack, Approximation algorithms for geometric shortest path problems, in: Proceedings of the 32nd Annual ACM Symposium on Theory of Computing, ACM Press, New York, NY, USA, 2000, pp. 286–295.

[9] Lyudmil Aleksandrov, Anil Maheshwari, Jörg-Rüdiger Sack, Determining approximate shortest paths on weighted polyhedral surfaces, J. ACM 52 (1) (2005) 25–53.

[10] John F. Canny, John H. Reif, New lower bound techniques for robot motion planning problems, in: Proceedings of the 28th Annual Symposium on Foundations of Computer Science, 1987, pp. 49–60.

[11] Jindong Chen, Yijie Han, Shortest paths on a polyhedron. I. Computing shortest paths, Internat. J. Comput. Geom. Appl. 6 (2) (1996) 127–144.

[12] Siu-Wing Cheng, Hyeon-Suk Na, Antoine Vigneron, Yajun Wang, Approximate shortest paths in anisotropic regions, in: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2007, pp. 766–774.

[13] Mark de Berg, Marc J. van Kreveld, Trekking in the Alps without freezing or getting tired, Algorithmica 18 (3) (1997) 306–323.

[14] Mark de Berg, Marc J. van Kreveld, Mark Overmars, Otfried Cheong, Computational Geometry: Algorithms and Applications, second ed., Springer-Verlag, Berlin, Germany, 2000.

[15] Michael L. Fredman, Robert E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, J. ACM 34 (3) (1987) 596–615.

[16] Sariel Har-Peled, Constructing approximate shortest path maps in three dimensions, SIAM J. Comput. 28 (4) (1999) 1182–1197.

[17] John Hershberger, Subhash Suri, An optimal algorithm for Euclidean shortest paths in the plane, SIAM J. Comput. 28 (6) (1999) 2215–2256.

[18] Mark Lanthier, Anil Maheshwari, Jörg-Rüdiger Sack, Shortest anisotropic paths on terrains, in: Proceedings of the 26th International Colloquium on Automata, Languages and Programming, Springer-Verlag, London, UK, 1999, pp. 524–533.

[19] Joseph S.B. Mitchell, David M. Mount, Christos H. Papadimitriou, The discrete geodesic problem, SIAM J. Comput. 16 (4) (1987) 647–668.

[20] Joseph S.B. Mitchell, Christos H. Papadimitriou, The weighted region problem: Finding shortest paths through a weighted planar subdivision, J. ACM 38 (1) (1991) 18–73.

[21] Esther Moet, Marc van Kreveld, A. Frank van der Stappen, On realistic terrains, in: Proceedings of the 22nd Annual Symposium on Computational Geometry, ACM Press, New York, NY, USA, 2006, pp. 177–186.

[22] Christos H. Papadimitriou, An algorithm for shortest-path motion in three dimensions, Inform. Process. Lett. 20 (1985) 259–263.

[23] Neil C. Rowe, Ron S. Ross, Optimal grid-free path planning across arbitrarily-contoured terrain with anisotropic friction and gravity effects, IEEE Trans. Robot. Autom. 6 (5) (1990) 540–553.

[24] Sasanka Roy, Algorithms for some geometric facility location and path planning problems, PhD thesis, Indian Statistical Institute, Kolkata, India, 2008.

[25] Sasanka Roy, Sandip Das, Subhas C. Nandy, Shortest monotone descent path problem in polyhedral terrain, Comput. Geom. Theory Appl. 37 (2) (2007) 115–133.

[26] Sasanka Roy, Sachin Lodha, Sandip Das, Anil Maheshwari, Approximate shortest descent path on a terrain, in: Proceedings of the 19th Canadian Conference on Computational Geometry, August 2007, pp. 189–192.

[27] Yevgeny Schreiber, Shortest paths on realistic polyhedra, in: Proceedings of the 23rd Annual Symposium on Computational Geometry, ACM, New York, NY, USA, 2007, pp. 74–83.

[28] Yevgeny Schreiber, Micha Sharir, An optimal-time algorithm for shortest paths on a convex polytope in three dimensions, in: Proceedings of the 22nd Annual Symposium on Computational Geometry, ACM, New York, NY, USA, 2006, pp. 30–39.

[29] Zheng Sun, Tian-Ming Bu, On discretization methods for approximating optimal paths in regions with direction-dependent costs, Inform. Process. Lett. 97 (4) (2006) 146–152.

[30] Zheng Sun, John H. Reif, BUSHWHACK: An approximation algorithm for minimal paths through pseudo-Euclidean spaces, in: Proceedings of the 12th International Symposium on Algorithms and Computation, Springer-Verlag, London, UK, 2001, pp. 160–171.

[31] Zheng Sun, John H. Reif, On finding energy-minimizing paths on terrains, IEEE Trans. Robotics 21 (1) (2005) 102–114.

[32] Zheng Sun, John H. Reif, On finding approximate optimal paths in weighted regions, J. Algorithms 58 (1) (2006) 1–32.